AN ABSTRACT OF THE THESIS OF

Connelly S. Barnes for the degree of Honors Baccalaureate of Science in Computational Physics and Mathematics presented on June 2nd, 2006. Title: ThermoSolver: An Integrated Educational Thermodynamics Software Program.

Abstract approved: ____
Mentor's Name

ThermoSolver is an educational thermodynamics software program designed to be both easy to use and useful in that it permits the user to make nontrivial chemical engineering thermodynamic calculations. The software program accompanies the textbook *Engineering and Chemical Thermodynamics* by Milo Koretsky, and is available for free download from the Web. ThermoSolver has a built-in database of chemical properties, and contains a number of sub-programs which perform calculations such as finding saturation pressures and fugacities, solving of equations of state, solving vapor-liquid equilibrium problems, and determining chemical reaction equilibria. The software has been integrated into the problems and examples in the textbook by Koretsky; it compares favorably with similar educational software, and has been used in several chemical engineering classes taught at Oregon State University and elsewhere.

Key Words: Chemical, engineering, thermodynamics, software, education

Corresponding e-mail address: barnesc@lifetime.oregonstate.edu

ThermoSolver: An Integrated Educational Thermodynamics Software Program

by

Connelly S. Barnes

A PROJECT

submitted to

Oregon State University

University Honors College

in partial fulfillment of
the requirements for the
degree of

Honors Baccalaureate of Science in Computational Physics and Mathematics

Presented June 2, 2006
Commencement June 2006

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THERMOSOLVER: AN INTEGRATED EDUCATIONAL THERMODYNAMICS SOFTWARE PROGRAM

I. INTRODUCTION

One of the challenges in teaching thermodynamics is that equations are often difficult to solve in closed form. With pencil and paper, students can solve simple thermodynamic problems, but as problems become more complicated, the resulting systems of equations often must be solved by numerical methods. For small, well-defined classes of problems, an instructor can typically develop a spreadsheet, a worksheet in a Computer Algebra System (CAS), or a script in a programming language which solves the problem. Students can then experiment with the worksheet or source code to solve thermodynamic problems within a fixed, narrow problem domain. However, if the instructor wishes to allow the student to explore a wider parameter space (e.g. use a large number of species, compare equations of state, activity coefficient models, correction factors, and so forth), then it is undesirable to have a large number of worksheets, and thus it is often preferable to create a code library of prewritten software routines, or develop a stand-alone software program which can perform thermodynamic calculations automatically.

Our goal is to allow students to "explore" thermodynamics – to make nontrivial thermodynamic calculations, and easily compare different thermodynamic equations and models, without being encumbered by programming every solution. Specifically, the objectives of this thesis are:

1. To create a thermodynamics program with an easy to use, intuitive graphical user interface.

2. To research and develop the numerical algorithms needed to solve the thermodynamic model equations.

We also wish to integrate the software with the textbook *Engineering and Chemical Thermodynamics* by Koretsky so that it is a natural extension of the learning curriculum[2]. ThermoSolver is a software program that we have engineered specifically to meet these goals.

ThermoSolver is a graphical Windows program available for free from the Web[1]. The program is composed of a number of mini-solvers which can be selected from a graphical menu. By using point-and-click navigation, students may compute fugacity and saturation pressure for species, solve equations of state, fit excess Gibbs energy models and investigate VLE systems, make bubble-point and dew-point calculations, and solve for chemical reaction equilibria. The components of the software system are designed to be didactic, and typically allow the student to compare various thermodynamic correction factors, equations of state, fugacity rules, and so forth.

To make ThermoSolver easy to use, we have incorporated a database of 353 chemical species into the software, and added plotting utilities, options for importing and exporting spreadsheet data, and online HTML documentation. We have integrated ThermoSolver with end of chapter exercises in the textbook by Koretsky[2].

We have found ThermoSolver to be widely useful, even beyond its intended purpose. Due to the integrated database, and the thermodynamic property calculators for single species, ThermoSolver is useful as a data source, and as a quick and practical tool for finding state properties of a given species. At Oregon State University, ThermoSolver has been used by students in chemical engineering thermodynamics courses, a graduate reactors course, and a chemical plant design course, as taught by faculty Milo Koretsky and Sho Kimura. In the plant design course, ThermoSolver was not specifically named as a tool that students should use, and yet 10 out of 39 students stated that they have used ThermoSolver at least once in the class. While we do not track usage statistics of other Universities, a quick Web search reveals that ThermoSolver has been used in classes at the University of Notre Dame in Indiana and the University of Colorado at Boulder.

II. COMPARISON WITH EXISTING THERMODYNAMICS SOFTWARE

In this chapter, we compare ThermoSolver with similar existing software: the textbook software provided with Elliott and Lira's *Introductory Chemical Engineering Thermodynamics*[3], the software with Sandler's *Chemical and Engineering Thermodynamics*[4], the software with Kyle's *Chemical and Process Thermodynamics*[5], and the Engineering Equation Solver by F-Chart Software, as bundled with *Thermodynamics: An Engineering Approach* by Çengel and Boles[6]. Each of these textbooks except for the last is used for chemical engineering. The textbook by Çengel and Boles is a general engineering text with a much larger market share. The best selling textbook in chemical engineering thermodynamics is *Chemical Engineering Thermodynamics* by Smith, Van Ness, and Abbott, which does not include any software[7].

We summarize the results of the comparison in Table 1. We include in our comparison only spreadsheets, computer algebra system worksheets, and programs designed to run on a personal computer. From our comparison, we specifically exclude programs for handheld calculators and spreadsheets which contain only examples (i.e. spreadsheets lacking general-purpose algorithms). Some captions in the table require some clarification: for the EOS solver, we use "solve P, v, T" to indicate that the program is capable of solving for pressure P, molar volume v, or temperature T given the other two state parameters; "saturation pressure" indicates that the program can calculate the saturation pressure at a given T, and "saturation temperature" indicates that the program can calculate saturation temperature at a given P. We use VL as an abbreviation for

Vapor-Liquid, LL as an abbreviation for Liquid-Liquid, SG as an abbreviation for Solid-Gas, and the suffix E when used stands for equilibrium. Finally, the category "param fitting, isobaric" indicates that the program is capable of solving for the parameters in an excess Gibbs energy (g^E) model so that the model matches experimental data.

Table 1 does not comprehensively cover every feature of the relevant programs. Instead, it purposely omits data so that one may see the general picture. The general conclusions are as follows: ThermoSolver does well in fitting and plotting solutions for binary VLE, although it omits two important activity coefficient models (UNIFAC and UNIQUAC); ThermoSolver is useful for finding saturation pressures and temperatures, fugacity coefficients and solving equations of state, however it does not provide steam tables or solve for the departure functions of enthalpy or entropy; ThermoSolver solves for single and multiple reaction equilibria but omits other useful equilibrium calculations such as LL equilibrium, SG equilibrium, VL flash, and LL flash. The software packages provided by Elliott and Lira, Sandler, and Kyle all use Microsoft Excel spreadsheets or Mathcad worksheets to implement a significant amount of functionality. In constrast, ThermoSolver does not use worksheets at all.

The Engineering Equation Solver allows the user to enter and solve arbitrary equations; moreover, these equations can call useful thermodynamic functions such as `pressure()`, `volume()`, `entropy()`, or `enthalpy()`. A large number of single species thermodynamic functions are built into the program, so that departure functions, steam properties, and

Table 1: Comparison of ThermoSolver and similar software packages.

	EOS Solver		Other Species Properties				Fugacity Calculations	
	Solve *v*	Solve *P*, *v*, *T*	Saturation pressure	Saturation temp.	Departure H and S	Steam properties	Single species	Mixture
ThermoSolver	✓	✓	✓	✓			✓	✓
Elliott and Lira	✓		✓		✓	✓	✓	
Sandler	✓				✓		✓	✓
Kyle	✓				******			
EES	✓	✓	✓	✓	✓	✓	✓	

	Binary VLE					Multicomponent VLE		
	Param fitting, isobaric	Param fitting, isothermal	Calculate activity coeffs	No. activity coeff. models	Plotting	No. activity coeff. models	Residue curves	
ThermoSolver	✓	✓	✓	5	✓	1****		
Elliott and Lira			✓	5		2	✓	
Sandler			✓	1		1		
Kyle	✓	✓	✓	4				
EES								

	Bubble point / dew point				Other Eq. Calculations			
	Bubble point	Dew point	Activity coeffs.	Fugacity corrections	VL flash	LL flash	LLE	SGE
ThermoSolver	✓	✓	✓	✓				
Elliott and Lira	✓	✓	✓	✓	✓	✓	✓	
Sandler	✓	✓	✓	✓	✓			
Kyle								✓
EES								

	Reaction Equilibria		Database		Program Type(s)		
	Single rxn.	Multiple rxn.	Integrated database	Species in database	Worksheets	Standalone programs	Modern graphical program
ThermoSolver	✓	✓	✓	353		✓	✓
Elliott and Lira	✓	**	*	0, 255*	✓	✓	
Sandler	✓		*	0, 99*	✓	✓	***
Kyle							
EES			✓	87		✓	✓

* Depends on which solver is used. In the software provided by Elliott and Lira and Sandler, most solvers do not provide a database of physical property data.

** A program is provided which will calculate mole fractions given "superficial mole fractions" and chemical reaction equilibrium constants.

*** Two programs are available as Visual Basic executables: the PR solver and the reaction equilibrium solver. The majority of Sandler's programs are DOS programs or Mathcad worksheets.

**** The multicomponent Wilson model is used in the bubble-point and dew-point solvers.

***** Kyle's software calculates only the enthalpy departure.

fugacity coefficients can be trivially calculated. However, the program does not include any general-purpose functions for working with multicomponent systems. The program does include a large number of examples, yet these examples are hard-coded so that it is not immediately obvious how to add another species to an equilibrium constant solver, or experiment with gas-phase correction factors. The same techniques used to make the Engineering Equation Solver could be extended so that multi-component systems can be solved by mere function calls – this might be a useful direction that future developers of educational software could explore. Still, the approach of typing in equations manually in a domain-specific language is somewhat intimidating for new users, so the point-and-click graphical interfaces as found in ThermoSolver have merit as well.

Although the table does not show it, the single most conspicuous difference between ThermoSolver and the programs by Kyle, Elliott and Lira, and Sandler is that ThermoSolver is remarkably more easy to use. That is, ThermoSolver has an integrated database which is used in all parts of the program, so users do not need to look up physical properties, and the use of a point-and-click interface integrated into a main menu allows users to easily "explore" the program. In contrast, the barrier to using the programs by Kyle, Elliott and Lira, and Sandler is rather high: in the DOS programs, one must typically read an appendix to decide which program should be run, then once the program is run one must read a page of instructions, carefully press the right keys, deal with loading or saving of data, and look up physical constants and enter them. With spreadsheets, similar steps are necessary: one must find which spreadsheet should be opened, decipher the meaning of the cells and look up and enter physical constants.

III. COMPUTATIONAL FEATURES OF THERMOSOLVER

The software program can be installed from the enclosed CD-ROM, or downloaded from the Web. The main menu for the program is shown in Figure 1. From here, a number of sub-programs can be selected: the Species Database contains properties for many of the built-in species; the Saturation Pressure Calculator finds saturation pressures and temperatures; the Equation of State Solver solves for one of the state properties P, v, T given the other two; the Fugacity Coefficient Solver solves for pure and mixed fugacities for systems of species; the Models for Excess Gibbs Energy (g^E) utility fits activity coefficient models to experimental data; the Bubble Point and Dew Point solver finds bubble points and dew points using various fugacity and activity coefficient corrections; the Chemical Reaction Equilibria solver finds single reaction equilibrium constants and multiple reaction equilibria.

Documentation is provided with the program, including usage instructions and full listings of the equations and numerical algorithms used.

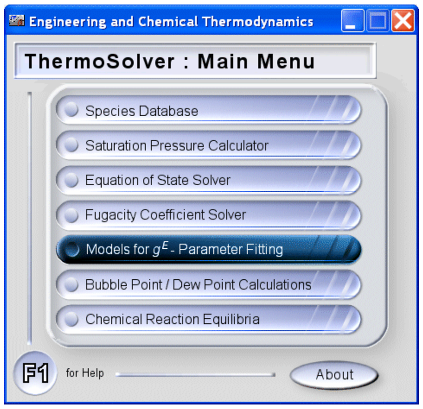

Engineering and Chemical Thermodynamics
ThermoSolver : Main Menu
Species Database
Saturation Pressure Calculator
Equation of State Solver
Fugacity Coefficient Solver
Models for g^E - Parameter Fitting
Bubble Point / Dew Point Calculations
Chemical Reaction Equilibria
F1
for Help
About

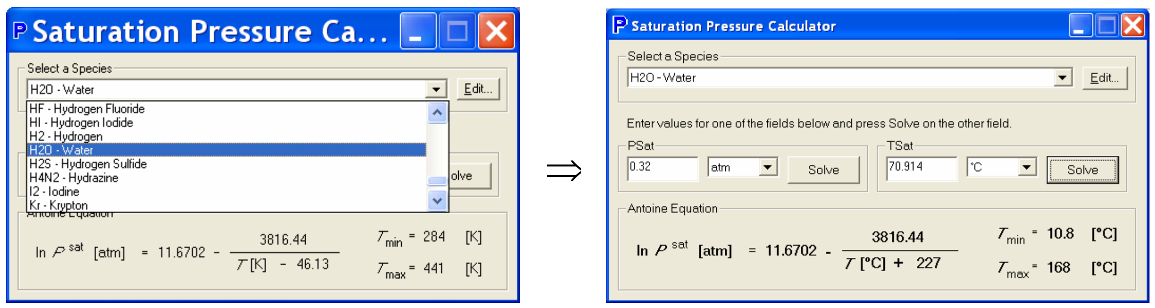
Figure 1: ThermoSolver main menu.

Species Database

The species database contains all thermodynamic constants for species used by the software. There are 353 species in the database by default. Each database entry stores information for a single species: constants stored include critical temperature, critical pressure, acentric factor, Antoine constants, and heat capacity constants. Not every species has every field filled out: some fields are left blank, and thus a species which omits a given constant will not be available in solvers which depend on that constant. The user may edit the database through a separate window: species may be added, removed, or constants for an existing species may be modified. The data in the species database are from various reference texts[8] [9] [10] [11] [12].

Saturation Pressure Calculator

The Saturation Pressure Calculator uses the Antoine equation to solve for the saturation pressure or saturation temperature for a single species. The species of interest can be selected from a drop-down menu; only species in the database which have Antoine coefficients may be selected. There are 338 such species available by default. If one wishes to solve for P^{sat} (saturation pressure), then the species temperature can be entered and the Solve button pressed next to the pressure box. Likewise if one wishes to solve for T^{sat} (saturation temperature), then the Solve button can be pressed next to the temperature box. An example use of the solver is shown in Figure 2. The form of the Antoine equation is given in the solver window to reinforce with the student how the calculation is being executed.


Saturation Pressure Ca...

Select a Species: H2O - Water

HF - Hydrogen Fluoride
HI - Hydrogen Iodide
H2 - Hydrogen
H2O - Water
H2S - Hydrogen Sulfide
H4N2 - Hydrazine
I2 - Iodine
Kr - Krypton

Antoine Equation

$\ln P^{sat}$ [atm] $= 11.6702 - \dfrac{3816.44}{T\,[K] - 46.13}$ $T_{min} = 284$ [K] $T_{max} = 441$ [K]

$\Rightarrow$

Saturation Pressure Calculator

Select a Species: H2O - Water

Enter values for one of the fields below and press Solve on the other field.

PSat: 0.32 atm — Solve

TSat: 70.914 °C — Solve

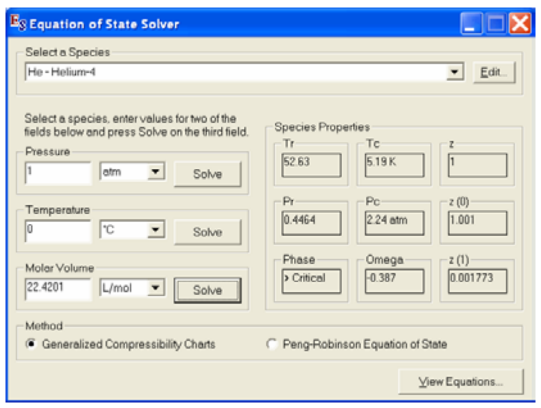
Antoine Equation

$\ln P^{sat}$ [atm] $= 11.6702 - \dfrac{3816.44}{T\,[°C] + 227}$ $T_{min} = 10.8$ [°C] $T_{max} = 168$ [°C]

Figure 2: The Saturation Pressure Calculator sub-program.

Equation of State Solver

The Equation of State Solver can be used to find one of the state properties *P, v*, *T* given the other two, for a single species, using the Lee-Kesler or Peng-Robinson equations of state [13] [14]. The window for the Equation of State solver is shown in Figure 3. The species should be selected from the drop-down menu; only species in the database that have Antoine constants, critical temperature, critical pressure, and acentric factor may be selected. There are 338 of such species available by default. Two of the three state properties should be entered, and the Solve button can be clicked to solve for the third. On the right-hand side, properties are shown for the species: the reduced temperature and


Equation of State Solver

Select a Species: He - Helium-4

Select a species, enter values for two of the fields below and press Solve on the third field.

Pressure: 1 atm — Solve

Temperature: 0 °C — Solve

Molar Volume: 22.4201 L/mol — Solve

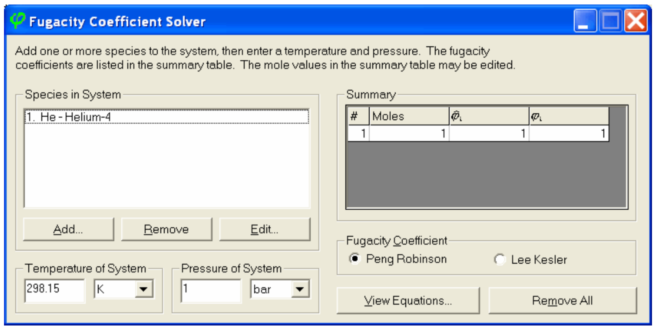
Species Properties

Tr: 52.63	Tc: 5.19 K	z: 1
Pr: 0.4464	Pc: 2.24 atm	z (0): 1.001
Phase: > Critical	Omega: -0.387	z (1): 0.001773

Method: [✓ Generalized Compressibility Charts] Peng-Robinson Equation of State

View Equations...

Figure 3: The Equation of State Solver.



Add one or more species to the system, then enter a temperature and pressure. The fugacity coefficients are listed in the summary table. The mole values in the summary table may be edited.

Species in System: 1. He - Helium-4

Summary

#	Moles	$\hat{\varphi}_i$	φ_i
1	1	1	1

Fugacity Coefficient: [✓ Peng Robinson] Lee Kesler

Temperature of System: 298.15 K

Pressure of System: 1 bar

Figure 4: The Fugacity Coefficient Solver

pressure, the critical temperature and pressure, the phase, the acentric factor ω, and the compressibility z. By default, the Lee-Kesler equation of state is used; this is indicated as a radio button titled "Generalized Compressibility Charts" (the textbook uses the Lee-Kesler equation of state for the compressibility charts in the appendix). The Peng-Robinson equation of state may also be selected.

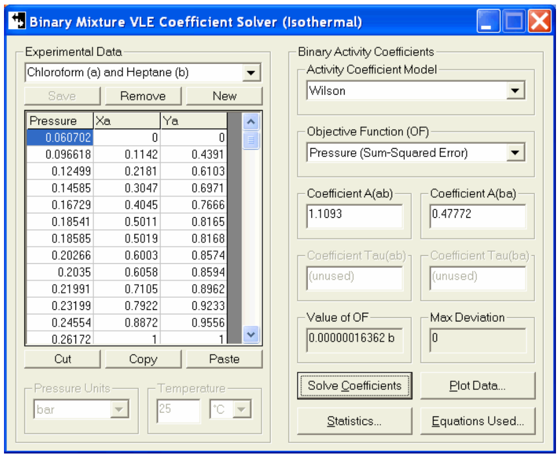
If the species is in the saturated vapor-liquid phase (i.e. both the liquid phase and the vapor phase exist), then the program displays a message box indicating the problem, with the liquid and vapor molar volumes and the liquid mole fraction. The View Equations button allows students to see how these calculations are executed.

Fugacity Coefficient Solver

The Fugacity Coefficient Solver solves for the pure fugacity coefficient ϕ_i or the fugacity coefficient $\hat{\phi}_i$ of each species in a mixture. The solver window is shown in Figure 4. The solver uses either the Peng-Robinson or Lee-Kesler equation of state[13] [14]. On the left side of the window, one can use the Add button to specify the number of moles of each species in the system, and the pressure and temperature can be chosen. On the right hand side of the window, the fugacity coefficients are displayed. If the Peng-Robinson equation of state is chosen, then pure and mixed fugacity coefficients are shown, but if the Lee-Kesler equation of state is chosen, then only pure fugacity coefficients are calculated.

Models for Excess Gibbs Energy (g^E) – Parameter Fitting

When the Models for g^E sub-program is selected, ThermoSolver asks the user whether isothermal or isobaric data will be analyzed. Once this choice has been made, the solver window for vapor-liquid equilibrium (VLE) data is shown. This window is shown in Figure 5. From the left side of the window, one can choose a pre-existing data set, or click New to create a new empty data set. On the right side of the window, the activity coefficient model can be specified: the 2-suffix Margules, 3-suffix Margules, Wilson, van Laar, and NRTL models are supported[2]. The objective function specifies which quantity will be minimized when fitting the model parameters to the data. Three objective



Experimental Data: Chloroform (a) and Heptane (b)

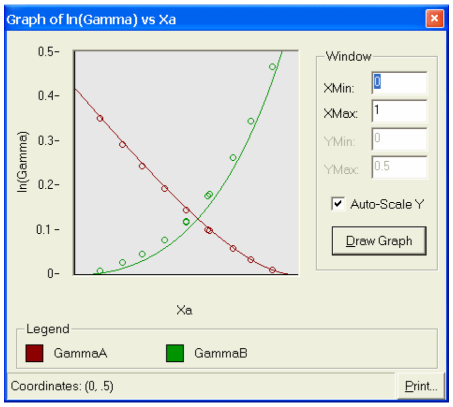
Pressure	Xa	Ya
0.060702	0	0
0.096618	0.1142	0.4391
0.12499	0.2181	0.6103
0.14585	0.3047	0.6971
0.16729	0.4045	0.7666
0.18541	0.5011	0.8165
0.18585	0.5019	0.8168
0.20266	0.6003	0.8574
0.2035	0.6058	0.8594
0.21991	0.7105	0.8962
0.23199	0.7922	0.9233
0.24554	0.8872	0.9556
0.26172	1	1

Pressure Units: bar
Temperature: 25 °C

Binary Activity Coefficients
Activity Coefficient Model: Wilson
Objective Function (OF): Pressure (Sum-Squared Error)
Coefficient A(ab): 1.1093
Coefficient A(ba): 0.47772
Coefficient Tau(ab): (unused)
Coefficient Tau(ba): (unused)
Value of OF: 0.00000016362 b
Max Deviation: 0

Figure 5: The solver window for VLE parameters.

functions are available: sum-squared error in pressure, sum-squared error in g^E, and sum-squared relative error in the activity coefficients γ_a and γ_b. Once the g^E model and objective function have been chosen, the Solve button can be clicked to find the model parameters which minimize the objective function. The Plot button can be used to view the curve fit by the model versus the experimental data points: plots can be made for $\ln\gamma_a$ and $\ln\gamma_b$ vs x_a, P vs x_a, P vs y_a, T vs x_a, T vs y_a, and y_a vs x_a., where x_i is the liquid mole fraction of species i and y_i is the vapor mole fraction of species i. See Figure 6 for an example plot. By default, only the current g^E model is plotted, but multiple g^E models can also be compared. The created plots can also be printed. The Statistics button reports a number of properties about the fit, such as the value of the three objective functions, and the mean and max deviations in pressure, γ_a, γ_b, g^E, and y_a. Again, the Equations Used button allows the student to identify the key thermodynamic relations.



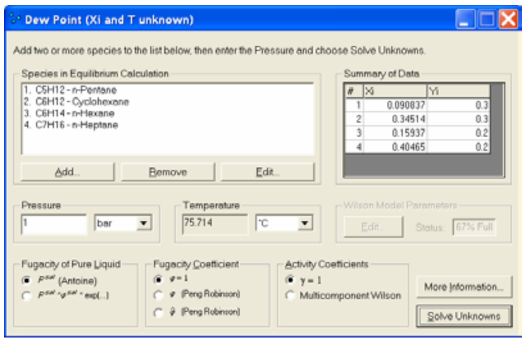
XMin: 0
XMax: 1
YMin: 0
YMax: 0.5
[✓ Auto-Scale Y]
Legend: 🟥 GammaA 🟩 GammaB
Coordinates: (0, .5)

Figure 6: An example plot for the system in Figure 5.

Bubble Point and Dew Point Calculations

ThermoSolver can make two bubble point calculations: it can solve for (y_i, P) given (x_i, T), or it can solve for (y_i, T) given (x_i, P). Likewise, the program can make two dew point calculations: it can solve for (x_i, P) given (y_i, T), or it can solve for (x_i, T) given (y_i, P). When the Bubble Point and Dew Point solver is first opened, a grid displaying these four options is shown. Once the appropriate option has been chosen, the Bubble Point or Dew Point solver is opened. The Dew Point solver is shown in Figure 7. Within the Dew Point solver, the Add button can be used to add one or more species with specified vapor mole fractions y_i. The known temperature or pressure can be entered, then the Solve button may be used to solve for the unknown temperature or pressure, as well as the liquid mole fractions x_i. At the bottom of the window, various choices are available: the pure liquid fugacity can be determined either from the Antoine P^{sat}, or with the Poynting correction included (assuming an incompressible liquid)[2]:

$$f_i^l = P_i^{\text{sat}} \varphi_i^{\text{sat}} \exp\left[\frac{v_i^{l,\text{sat}} (P - P_i^{\text{sat}})}{RT} \right] \tag{3.1}$$



Add two or more species to the list below, then enter the Pressure and choose Solve Unknowns.

Species in Equilibrium Calculation

1. C5H12 - n-Pentane
2. C6H12 - Cyclohexane
3. C6H14 - n-Hexane
4. C7H16 - n-Heptane

Summary of Data

#	Xi	Yi
1	0.090837	0.3
2	0.34514	0.3
3	0.15937	0.2
4	0.40465	0.2

Pressure: 1 bar

Temperature: 75.714 °C

Fugacity of Pure Liquid: [✓ P^{sat} (Antoine)] $P^{sat} \cdot \varphi^{sat} \cdot \exp(...)$

Fugacity Coefficient: [✓ $\varphi = 1$] φ (Peng Robinson) $\hat{\varphi}$ (Peng Robinson)

Activity Coefficients: [✓ $\gamma = 1$] Multicomponent Wilson

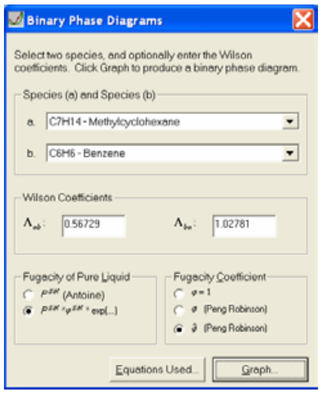
Figure 7: The Dew Point solver.

The fugacity coefficient used in the bubble point solver can either be unity, pure species ϕ_i from the Peng-Robinson equation of state, or the fugacity coefficient $\hat{\phi}_i$ in a mixture from the Peng-Robinson equation of state. The activity coefficients can either be unity or determined from the multicomponent Wilson equation. If the Wilson equation is chosen and there are n species in the system, then typically n^2 Wilson parameters must be entered, although Wilson parameters are "known" by the software for 473 binary systems, so in some cases the grid of n^2 Wilson parameters may be partially filled in.

If the More Information button is selected, then a table containing either $(x_i, y_i, \hat{\varphi}_i, f_i^l, P_i^{\text{sat}})$ or $(x_i, y_i, \varphi_i, f_i^l, P_i^{\text{sat}})$ is shown, depending on the chosen corrections. The data can be copied and pasted into a spreadsheet program such as Microsoft Excel or OpenOffice Calc.

Binary Phase Diagrams

Binary phase diagrams can be produced by selecting Bubble Point and Dew Point Calculations from the main menu and then clicking the Binary Phase Diagrams button. The binary phase diagrams window is depicted in Figure 8. The phase diagram solver uses the same equations and correction factors as the Bubble Point and Dew Point solver, however the system must contain exactly two species. If the Graph button is clicked, then one can plot $\ln\gamma_a$ and $\ln\gamma_b$ vs x_a, P vs x_a or y_a (with constant temperature), T vs x_a or y_a (constant pressure), and y_a vs x_a (constant temperature or constant pressure). In each case, the constant temperature or constant pressure is specified. Again, the created plots can be printed.


Binary Phase Diagrams

Select two species, and optionally enter the Wilson coefficients. Click Graph to produce a binary phase diagram.

Species (a) and Species (b)

a. C7H14 - Methylcyclohexane

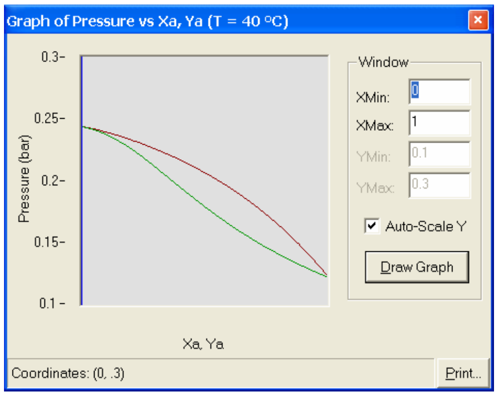
b. C6H6 - Benzene

Wilson Coefficients

Λ_{ab}: 0.56729 Λ_{ba}: 1.02781

Fugacity of Pure Liquid: p^{sat} (Antoine) [✓ $p^{sat} \cdot \varphi^{sat} \cdot \exp(\ldots)$]

Fugacity Coefficient: $\varphi = 1$ / φ (Peng Robinson) [✓ $\hat{\varphi}$ (Peng Robinson)]


Graph of Pressure vs Xa, Ya (T = 40 °C)

Window

XMin: 0

XMax: 1

YMin: 0.1

YMax: 0.3

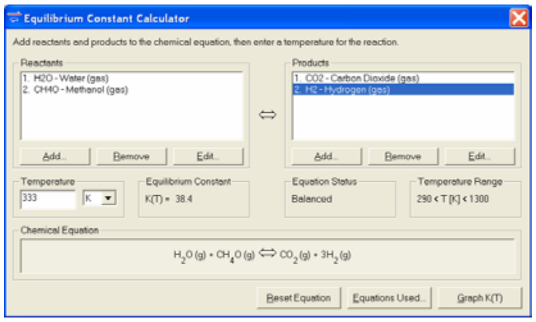
[✓ Auto-Scale Y]

Coordinates: (0, .3)

Figure 8: (Left) The Binary Phase Diagrams sub-program. (Right) A plot of P vs x_a (red) and P vs y_a (green) at 40 °C.

Equilibrium Constant (K_T) Solver

The Equilibrium Constant Solver determines the equilibrium constant K_T at a single temperature, or plots K_T over a range of temperatures, for a single chemical reaction. The Equilibrium Constant Solver window is shown in Figure 9. The mole values of the reactants and products can be specified by clicking the Add buttons on the left and right hand sides of the window, respectively. The "equation status" frame will read "unbalanced" until the user has balanced the equation. The temperature can be changed and the equilibrium constant K_T will be displayed. If the Graph button is clicked, then a plot of $\log_{10} K_T$ vs $1000/T$ [K] is produced.


Equilibrium Constant Calculator

Add reactants and products to the chemical equation, then enter a temperature for the reaction.

Reactants

1. H2O - Water (gas)
2. CH4O - Methanol (gas)

Products

1. CO2 - Carbon Dioxide (gas)
2. H2 - Hydrogen (gas)

Temperature: 333 K

Equilibrium Constant: K(T) = 38.4

Equation Status: Balanced

Temperature Range: 290 < T [K] < 1300

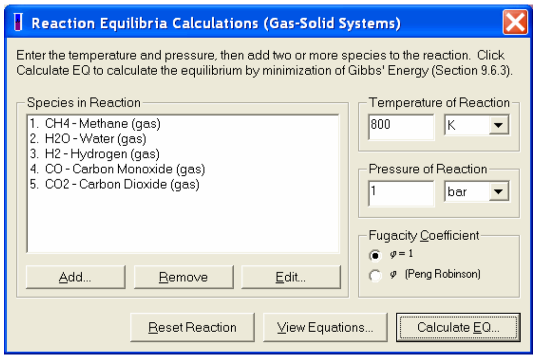
Chemical Equation

$H_2O\,(g) + CH_4O\,(g) \Leftrightarrow CO_2\,(g) + 3H_2\,(g)$

Figure 9: Equilibrium Constant Solver.

Multiple Chemical Reaction Equilibria

The Multiple Chemical Reaction Equilibria solver finds the number of moles and gas mole fraction y_i of each species at equilibrium via a minimization of g^E technique described in Section 9.7 of Koretsky[3]. The solver window is shown in Figure 10. The solver works for gas-solid systems. Any number of species may be included in the system. The initial number of moles must be specified for each species, and if it not available from the database, then the Gibbs' energy of formation must also be specified. An option allows the fugacity coefficients to either be set to unity or to the Peng-Robinson pure species ϕ_i.


Reaction Equilibria Calculations (Gas-Solid Systems)

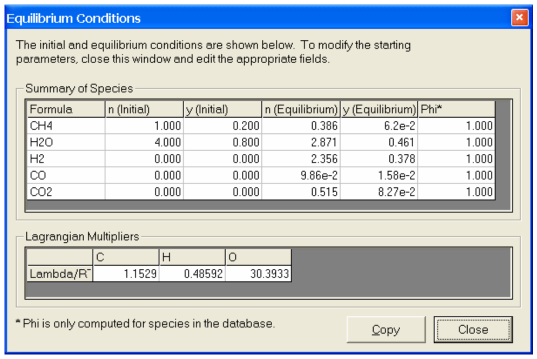
Enter the temperature and pressure, then add two or more species to the reaction. Click Calculate EQ to calculate the equilibrium by minimization of Gibbs' Energy (Section 9.6.3).

Species in Reaction
1. CH4 - Methane (gas)
2. H2O - Water (gas)
3. H2 - Hydrogen (gas)
4. CO - Carbon Monoxide (gas)
5. CO2 - Carbon Dioxide (gas)

Temperature of Reaction: 800 K

Pressure of Reaction: 1 bar

Fugacity Coefficient: [✓ $\varphi = 1$] φ (Peng Robinson)


Equilibrium Conditions

The initial and equilibrium conditions are shown below. To modify the starting parameters, close this window and edit the appropriate fields.

Summary of Species

Formula	n (Initial)	y (Initial)	n (Equilibrium)	y (Equilibrium)	Phi*
CH4	1.000	0.200	0.386	6.2e-2	1.000
H2O	4.000	0.800	2.871	0.461	1.000
H2	0.000	0.000	2.356	0.378	1.000
CO	0.000	0.000	9.86e-2	1.58e-2	1.000
CO2	0.000	0.000	0.515	8.27e-2	1.000

Lagrangian Multipliers

	C	H	O
Lambda/RT	1.1529	0.48592	30.3933

* Phi is only computed for species in the database.

Figure 10: (Left) Multiple Chemical Reaction Equilibria solver. (Right) Equilibrium conditions for the specified system.

IV. SOFTWARE DESIGN: NUMERICAL METHODS AND ALGORITHMS

This chapter describes the numerical algorithms used "under the hood" to perform ThermoSolver's calculations. We give the thermodynamic equations and a description of the algorithms used for each solver. In some cases, further detail may be available from the program's built-in help system.

Multidimensional Root Finder

Throughout the program, we often wish to compute a root of a vector-valued function $f : \mathbf{R}^n \mapsto \mathbf{R}^n$. Our standard method for solving this problem is to use a modified Newton's method. With Newton's method, one starts with a guess value $\mathbf{x}_0 \in \mathbf{R}^n$, and iterates

$$\mathbf{x}_{i+1} = \mathbf{x}_i - (Df|_{\mathbf{x}_i})^{-1} f(\mathbf{x}_i) \tag{4.1}$$

until $f(\mathbf{x}_i)$ is sufficiently small. Here *Df* is the Jacobian matrix, evaluated at $\mathbf{x}_i$:

$$Df = \begin{bmatrix} \dfrac{\partial f_1}{\partial x_1} & \cdots & \dfrac{\partial f_1}{\partial x_n} \\ \vdots & & \vdots \\ \dfrac{\partial f_n}{\partial x_1} & \cdots & \dfrac{\partial f_n}{\partial x_n} \end{bmatrix} \tag{4.2}$$

The product $(Df|_{\mathbf{x}_i})^{-1} f(\mathbf{x}_i)$ is evaluated via Gaussian elimination. Newton's method is derived from the first-order Taylor expansion of *f*, so in regions where *f* is not approximately linear and "near" a root, the Newton iterations tend to behave erratically or diverge to infinity. To prevent this behavior, we modify the standard Newton iteration to a simplified version of the "globally convergent" Newton-like iteration in Press et al[15].

For simplicity, in the following discussion we will shorten $f(\mathbf{x}_i)$ to *f* and $Df|_{\mathbf{x}_i}$ to *Df*, with the assumption that *f* is always evaluated at $\mathbf{x}_i$. Observe that in each iteration the Newton step $\boldsymbol{\delta} = -(Df)^{-1} f$ is a descent direction for the function $\|f\|^2 = (f \cdot f)$. That is, for $\|f\|^2 \neq 0$, the gradient $\nabla\|f\|^2$ evaluated in the direction $\boldsymbol{\delta}$ is

$$\left[\nabla\|f\|^2\right]\boldsymbol{\delta} = \left[2f^T(Df)\right]\left[-(Df)^{-1}f\right] = -2(f \cdot f) = -2\|f\|^2 < 0 . \qquad (4.3)$$

Thus if we use the modified Newton's step $\mathbf{x}_{i+1} = \mathbf{x}_i + \lambda\boldsymbol{\delta}$, then for some sufficiently small $\lambda \in [0,1]$, we are guaranteed that $\|f\|^2$ will decrease. Our modified Newton's algorithm is then: compute the Newton step $\boldsymbol{\delta}$, try $\lambda = 1,\ 1/2,\ 1/4,\ 1/8,\ \ldots$ until $\|f\|^2$ evaluated at $\mathbf{x}_{i+1}$ is less than or equal to $\|f\|^2$ evaluated at $\mathbf{x}_i$, then accept the new position $\mathbf{x}_{i+1}$ and repeat this iterative process until $\|f(\mathbf{x}_i)\|$ is below some error tolerance. Press et al note that such a method may fail to converge even to a minimum of $\|f\|^2$, but in our problems we found that this simple scheme converges reliably[15].

Numerical Methods Used in Equation of State Solver

The equation of state calculator solves two equations of state: the Peng-Robinson equation[14] and the Lee-Kesler equation[13]. The Peng-Robinson equation of state is:

$$\begin{aligned}
&P = \frac{RT}{v-b} - \frac{a\alpha}{v^2 + 2bv - b^2}, \\
&a = 0.45724\frac{(RT_c)^2}{P_c}, \qquad b = 0.07780\frac{RT_c}{P_c}, \\
&\alpha = \left[1 + (0.37464 + 1.54226\omega - 0.26992\omega^2)(1 - \sqrt{T/T_c})\right]^2
\end{aligned} \qquad (4.4)$$

Here P_c and T_c are the critical temperature and pressure, and ω is the acentric factor. These values can be measured experimentally, and are presumed to be known for the given species. The Peng-Robinson equation is straightforward to solve for one of *P*, *v*, or *T* given the other two variables, because the equations can be solved analytically. To determine the phase, we use a technique which we developed from the paper by Lee and Kesler, which is described later on (see equation (4.6)).

The Lee-Kesler equation of state relates *P*, *v*, *T* with the following equations:

$$P_r = \frac{P}{P_c}, \qquad T_r = \frac{T}{T_c}$$

$$B = b_1 - b_2 T_r^{-1} - b_3 T_r^{-2} - b_4 T_r^{-3}, \quad C = c_1 - c_2 T_r^{-1} + c_3 T_r^{-3}, \quad D = d_1 + d_2 T_r^{-1}$$

$$z^{(i)} = \frac{P_r v^*}{T_r} = 1 + \frac{B}{v^*} + \frac{C}{\left(v^*\right)^2} + \frac{D}{\left(v^*\right)^5} + \frac{c_4^*}{T_r^3 \left(v^*\right)^2}\left(\beta + \frac{\gamma}{\left(v^*\right)^2}\right)\exp\left[\frac{-\gamma}{(v^*)^2}\right] \qquad (4.5)$$

The equations are solved in the following manner: first *P* and *T* are chosen. Next P_r and T_r can be found, and the variables $B, C, D, c_4, \beta,$ and γ can be evaluated from the table:

	Simple (*i* = 0)	Correction (*i* = 1)
b_1	0.1181193	0.2026579
b_2	0.265728	0.331511
b_3	0.154790	0.027655
b_4	0.030323	0.203488
c_1	0.0236744	0.0313385
c_2	0.0186984	0.0503618
c_3	0	0.06901
c_4	0.042724	0.041577
d_1	1.55488×10^{-5}	4.8736×10^{-5}
d_2	6.23689×10^{-5}	7.40336×10^{-6}
β	0.65392	1.226
γ	0.060167	0.03754

Now if $i = 0$, we use the constants in the left column, and numerically solve for $v^{*(0)}$ in equation (4.5). If $i = 1$, we use the constants in the right column, and solve for $v^{*(1)}$ in equation (4.5). Once $v^{*(0)}$ and $v^{*(1)}$ have been determined, we evaluate $z^{(0)}$ and $z^{(1)}$, which are used to find the molar volume v:

$$z = z^{(0)} + \frac{\omega\left[z^{(1)} - z^{(0)}\right]}{0.3978}$$
$$v = \frac{zRT}{P}$$

The original paper by Lee and Kesler states these equations, but does not specify which root to choose when there are multiple roots. We have adopted the following conventions: if the species is in liquid phase, then choose the smallest positive root $v^{*(0)}$; if the species is in gas phase, then choose the largest positive root $v^{*(0)}$ (there is only one root if the species is in the supercritical phase). Regardless of the phase, for $v^{*(1)}$ we choose the positive root which minimizes $\left|z^{(1)} - z^{(0)}\right|$. To determine the phase, we used the Wagner equation of Reid et al[8] with an additional constant and 12th-order term:

$$\log_{10} P_r^{\text{sat}} = a_1 + T_r^{-1}(a_2\tau + a_3\tau^{1.5} + a_4\tau^3 + a_5\tau^6 + a_6\tau^{12}) \tag{4.6}$$

Minimizing the squared error in P_r against the data in Lee and Kesler's paper, we obtain:

	$P_r^{\text{sat},(0)}$	$P_r^{\text{sat},(1)}$
a_1	0.000475	-0.001037
a_2	-2.625398	-2.055769
a_3	0.636566	0.158476
a_4	-0.644317	-4.040313
a_5	0.129073	0.385900
a_6	-0.195596	-1.860181

We then have $\log_{10} P_r^{\text{sat}} = \log_{10} P_r^{\text{sat}(0)} + \omega \log_{10} P_r^{\text{sat}(1)}$. When solving for *v* given *P* and *T*, we use our modified Newton root finder with the initial guess values

$$\begin{aligned} v^{*(0)} &= \frac{T_r}{P_r} \cdot \begin{cases} 10^{-7}, & P_r > P_r^{sat} \text{ and } T_\text{r} < 1 \\ 1, & \text{otherwise} \end{cases} \\ v^{*(1)} &= \left\{ \frac{T_r}{2P_r}, \frac{2T_r}{P_r} \right\} \end{aligned} \tag{4.7}$$

That is, there are two initial values for $v^{*(1)}$ and the appropriate converged root is chosen according to the criteria given earlier. If the phase is determined by only the $P_r^{\text{sat},(0)}$ term, then the compressibility charts in Lee and Kesler's paper are reproduced exactly by the above method. Because Lee and Kesler use only the $P_r^{\text{sat},(0)}$ term in determining the phase, the compressibility charts in the paper will not always give the correct phase. In our software, the $P_r^{\text{sat},(0)}$ term alone is used, but we intend to change this in the future to use the Antoine equation. When solving for *P* given *T* and *v*, we define a new function

$$f(P) = \frac{P_r v}{T_r} - z(P,T) \tag{4.8}$$

Where $z(P,T)$ is the converged root from the previously described root finder which solves for *v*. We then use the modified Newton root finder on *f* to find the pressure *P* which solves the state equation. When solving for *T* given *P* and *v*, we use an identical approach, except that we now vary *T* when computing *f*.

Numerical Methods Used for Parameter Fitting of g^E Models

The parameter fitting solver fits activity coefficient models to experimental data for binary vapor-liquid equilibrium systems. We denote the two species in the binary system as species *a* and species *b*. The solver requires a list of experimental data points of the form (P, x_a, y_a) (isothermal, with specified *T*), or (T, x_a, y_a) (isobaric, with specified *P*). The activity coefficient γ_i is used in the relation,

$$x_i \gamma_i P_i^{\text{sat}} = y_i P \tag{4.9}$$

Where *i* denotes either species *a* or species b, x_i is the liquid mole fraction, y_i is the vapor mole fraction, *P* is the pressure, and P_i^{sat} is the saturation pressure of species *i*. Note that the activity coefficient γ_i and P_i^{sat} implicitly depend on temperature. In theory, if the experimental data had no errors, then equation (4.9) would be satisfied exactly at every point in the data set. In practice, the experimental data contain variation, so we use a mathematical model for γ_i which has from 1 to 3 adjustable parameters, and we modify the parameters in the model so that equation (4.9) is approximately satisfied at every data point. The activity coefficient models supported by ThermoSolver are shown in Table 2.

To specify how we wish to "approximately" solve equation (4.9), we use an objective function. The objective function identifies which quantity will be minimized when fitting the model parameters to the data. Three objective functions are available: sum-squared error in pressure, sum-squared error in excess Gibbs' energy (g^E), and sum-squared

	g^E	$RT\ln\gamma_a$	$RT\ln\gamma_b$
2 suffix Margules	Ax_ax_b	Ax_b^2	Ax_a^2
3 suffix Margules	$x_ax_b\left[A+B(x_a-x_b)\right]$	$(A+3B)x_b^2-4Bx_b^3$	$(A-3B)x_a^2+4Bx_a^3$
van Laar	$x_ax_b\left(\dfrac{AB}{Ax_a+Bx_b}\right)$	$A\left(\dfrac{Bx_b}{Ax_a+Bx_b}\right)^2$	$B\left(\dfrac{Ax_a}{Ax_a+Bx_b}\right)^2$
Wilson	$-RT\begin{bmatrix} x_a\ln(x_a+\Lambda_{ab}x_b) \\ +x_b\ln(x_b+\Lambda_{ba}x_a)\end{bmatrix}$	$-RT\begin{bmatrix}\ln(x_a+\Lambda_{ab}x_b)+ \\ x_b\left(\dfrac{\Lambda_{ba}}{x_b+\Lambda_{ba}x_a}-\dfrac{\Lambda_{ab}}{x_a+\Lambda_{ab}x_b}\right)\end{bmatrix}$	$-RT\begin{bmatrix}\ln(x_b+\Lambda_{ba}x_a)+ \\ x_a\left(\dfrac{\Lambda_{ab}}{x_a+\Lambda_{ab}x_b}-\dfrac{\Lambda_{ba}}{x_b+\Lambda_{ba}x_a}\right)\end{bmatrix}$
NRTL	$RTx_ax_b\left(\dfrac{\tau_{ba}G_{ba}}{x_a+x_bG_{ba}}+\dfrac{\tau_{ab}G_{ab}}{x_b+x_aG_{ab}}\right)$	$RTx_b^2\left[\dfrac{\tau_{ba}G_{ba}^2}{(x_a+x_bG_{ba})^2}+\dfrac{\tau_{ab}G_{ab}}{(x_b+x_aG_{ab})^2}\right]$	$RTx_a^2\left[\dfrac{\tau_{ba}G_{ba}}{(x_a+x_bG_{ba})^2}+\dfrac{\tau_{ab}G_{ab}^2}{(x_b+x_aG_{ab})^2}\right]$

Table 2: Activity coefficient models, reproduced with permission from Koretsky[2]. The NRTL model depends on three parameters, as G_{ab} and G_{ba} are not independent but instead are given in terms of a single parameter α_{ab}: $G_{ab}=\exp(-\alpha_{ab}\tau_{ab})$, $G_{ba}=\exp(-\alpha_{ba}\tau_{ba})$. The Wilson model's parameters must be positive.

relative error in the activity coefficients γ_a and γ_b:

$$OF_P=\sum(P_{\text{exp}}-P_{\text{calc}})^2 \tag{4.10}$$

$$OF_{g^E}=\sum(g^E_{\text{exp}}-g^E_{\text{calc}})^2 \tag{4.11}$$

$$OF_\gamma=\sum\left[\left(\frac{\gamma_{a,\text{exp}}-\gamma_{a,\text{calc}}}{\gamma_{a,\text{exp}}}\right)^2+\left(\frac{\gamma_{b,\text{exp}}-\gamma_{b,\text{calc}}}{\gamma_{b,\text{exp}}}\right)^2\right] \tag{4.12}$$

The summation runs through all points in the list of experimentally measured data points, the subscript "exp" indicates the experimentally measured value at the given data point, and "calc" indicates the calculated value from the model. The variables with subscript "exp" can be precomputed from the experimental data: at each data point, equation (4.9) can be solved for $\gamma_{i,\text{exp}}$, and the remaining experimental values are given by:

$$g^E_{\exp} = RT(x_a \ln \gamma_a + x_b \ln \gamma_b) \tag{4.13}$$

$$P_{\exp} = x_a \gamma_a P^{\text{sat}}_a + x_b \gamma_b P^{\text{sat}}_b \tag{4.14}$$

In the special case where x_i and y_i are zero, the equation $\gamma_i P_i^{\text{sat}} = P$ is used instead. The variables with subscript "calc" must be recomputed with every change to the parameters in the activity coefficient model: g^E_{calc} and $\gamma^E_{i,\text{calc}}$ can be computed directly from Table 2, while P_{calc} can be computed from equation (4.14). Thus given any particular data set and the parameters of the activity coefficient model, we can evaluate each of the three objective functions.

To minimize the objective function, we use the Nelder-Mead simplex method[16], with starting points of $A = RT$ for the two-suffix Margules model, $(A, B) = (RT, RT)$ for the three-suffix Margules and van Laar models, $(\Lambda_{ab}, \Lambda_{ba}) = (1,1)$ for the Wilson model, and $(\alpha_{ab}, \tau_{ab}, \tau_{ba}) = \pm(0.1, 0.1, 0.1)$ for the NRTL model (where the $\pm$ indicates that two starting locations are chosen, and the converged parameters are compared by objective function to determine the best fit). When solving a minimization problem in *n* variables, the Nelder-Mead method requires *n* + 1 starting vertices, so for the first vertex we use the starting point $\mathbf{x}_0 \in \mathbf{R}^n$, and for the remaining vertices we use $\mathbf{x}_i = \mathbf{x}_0 + 10\mathbf{e}_i x_{0i}$, where $\mathbf{e}_i$ is the *i*th canonical basis vector. Typically 1000 iterations are sufficient to produce a good fit. For our data sets it takes under one second to solve for the parameters on a machine with a 1 GHz processor.

Algorithms Used in Bubble Point and Dew Point Calculations

In a bubble point calculation, we are given (x_i, P) or (x_i, T) and we wish to solve for the variables (y_i, T) or (y_i, P), respectively, in the system of equations

$$y_i \hat{\phi}_i P = x_i \gamma_i f_i^l \tag{4.15}$$
$$\sum y_i = 1 \tag{4.16}$$

In a dew point calculation, we are given (y_i, P) or (y_i, T) and we solve for (x_i, T) or (x_i, P), respectively, in the system of equations

$$y_i \hat{\phi}_i P = x_i \gamma_i f_i^l \tag{4.17}$$
$$\sum x_i = 1 \tag{4.18}$$

ThermoSolver allows the user to make trade-offs between simplicity and thermodynamic correctness. The user can choose as the pure liquid fugacity f_i^l either the low-pressure and low-temperature approximation $f_i^l = P_i^{\text{sat}}$ or else the Poynting correction for an incompressible fluid can be included:

$$f_i^l = P_i^{\text{sat}} \phi_i^{\text{sat}} \exp\left[\frac{v_i^{l,\text{sat}}(P - P_i^{\text{sat}})}{RT}\right] \tag{4.19}$$

Here P_i^{sat} is evaluated from the Antoine equation, ϕ_i^{sat} is evaluated from the pure species fugacity coefficient in the Peng-Robinson equation of state at the current temperature and the critical pressure, and $v_i^{l,sat}$ is evaluated from the Rackett equation[2]:

$$v_i^l = \frac{RT_{c,i}\left(0.29056 - 0.08775\omega_i\right)^{1+\tau_i^{0.28571}}}{P_i}, \tau_i = 1 - \frac{T}{T_{c,i}} \tag{4.20}$$

Thus only the critical pressure and temperature, acentric factor, and Antoine constants are needed to evaluate the pure liquid fugacity f_i^l.

The user can choose to have a fugacity coefficient $\hat{\phi}_i$ of unity, the pure species ϕ_i from the Peng-Robinson equation of state, or the fugacity coefficient $\hat{\phi}_i$ of species *i* in a mixture from the Peng-Robinson equation of state[13]. The Peng-Robinson pure and mixed fugacity coefficients can be derived from the Peng-Robinson equation of state, using calculus and approximations for interaction terms, as given by Koretsky[2]. The full expressions for the Peng-Robinson fugacity coefficients are given in the HTML documentation for ThermoSolver[1].

Finally, the user can choose to use either an activity coefficient of one, or the multicomponent Wilson equation[2]:

$$\gamma_k = 1 - \ln\left(\sum_{j=1}^{m} x_j \Lambda_{kj}\right) - \sum_{i=1}^{m} \frac{x_i \Lambda_{ik}}{\ln\left(\sum_{j=1}^{m} x_j \Lambda_{ij}\right)} \tag{4.21}$$

The parameters in the multicomponent Wilson equation are determined by binary VLE parameter fitting: if *i* and *j* are two species in the system, then the Wilson components should be found for the binary system *i* – *j*, which then become the multicomponent Wilson parameters Λ_{ij} and Λ_{ji}. Wilson parameters are temperature-specific, so when we wish to convert Wilson parameters from one temperature to another, we must find the temperature-independent parameters ζ_{ij} [2]:

$$\Lambda_{ij} = \frac{v_j^l}{v_i^l} \exp\left[\frac{-\zeta_{ij}}{RT}\right] \tag{4.22}$$

Thus for each pair of Wilson parameters Λ_{ij} and Λ_{ji}, the user must also enter the temperature at which the activity coefficient model was fit.

Once the correction factors have been specified, the bubble point or dew point must be solved. There are two bubble point problems and two dew point problems, so four algorithms are needed to solve all cases. We use straightforward modifications of the algorithms given in Smith, Van Ness and Abbott[7].

The following equations are used by the four algorithms:

$$P = \sum_i \frac{y_i \hat{\phi}_i}{\gamma_i f_i^l} \tag{4.23}$$

$$P = \sum_i \frac{x_i \gamma_i f_i^l}{\hat{\phi}_i} \tag{4.24}$$

$$P_i^{\text{sat,corrected}} = \frac{y_i \hat{\phi}_i P}{x_i \gamma_i f_i^l} P_i^{\text{sat,Antoine}} \tag{4.25}$$

$$T = \frac{B_i}{A_i - \ln P_i^{\text{sat,corrected}}} - C_i \tag{4.26}$$

The term $P_i^{\text{sat,Antoine}}$ indicates the saturation pressure for species *i* computed from the Antoine equation, and equation (4.26) is simply the inverse of the Antoine equation for species *i*.

Algorithm for dew point calculation of (x_i, P):

```
Initialize $\hat{\phi}_i$, $\gamma_i$, and $f_i^l$ to unity.
Evaluate $P$ from equation (4.23).
Evaluate $f_i^l$ from the Antoine equation or equation (4.19).
Evaluate $P$ from equation (4.23).
Do
    Evaluate $\hat{\phi}_i$ from pure or mixed Peng Robinson EOS, or set to unity.
    Evaluate $f_i^l$ from the Antoine equation or equation (4.19).
    Do
        Solve equation (4.15) for $x_i$.
        Multiply each $x_i$ by $1/\sum x_i$, so that $\sum x_i = 1$.
        Evaluate $\gamma_i$ from equation (4.21), or set to unity.
    Loop until change in params from top of "Do" loop is $<$ threshold.
    Evaluate $P$ from equation (4.23).
Loop until change in parameters from top of "Do" loop is $<$ threshold.
```

For the threshold value, we require that each of the changes in x_i and y_i be less than ε, and we require that the relative "error" in P and T as computed against the previous value P_{old} or T_{old} be less than ε. We chose $\varepsilon = 10^{-7}$, and we also terminate the loops if a maximum number of iterations is exceeded.

Algorithm for bubble point calculation of (y_i, P):

```
Initialize $\hat{\phi}_i$, $\gamma_i$, and $f_i^l$ to unity.
Evaluate $P$ from equation (4.24).
Evaluate $f_i^l$ from the Antoine equation or equation (4.19).
Evaluate $P$ from equation (4.24).
Evaluate $\gamma_i$ from equation (4.21), or set to unity.
Evaluate $P$ from equation (4.24).
Do
    Solve equation (4.15) for $y_i$.
    Evaluate $\hat{\phi}_i$ from pure or mixed Peng Robinson EOS, or set to unity.
    Evaluate $f_i^l$ from the Antoine equation or equation (4.19).
    Evaluate $\gamma_i$ from equation (4.21), or set to unity.
    Evaluate $P$ from equation (4.24).
Loop until change in parameters from top of "Do" loop is $<$ threshold.
```

Algorithm for bubble point calculation of (y_i, T):

```
Initialize $\hat{\phi}_i$ to unity.
Initialize $T = \sum x_i T_i^{\text{sat}}$ , where $T_i^{\text{sat}}$ is calculated from Antoine equation.
Evaluate $f_i^l$ from the Antoine equation or equation (4.19).
Evaluate $\gamma_i$ from equation (4.21), or set to unity.
Do
    Evaluate $f_i^l$ from the Antoine equation or equation (4.19).
    Solve equation (4.15) for $y_i$.
    Multiply each $y_i$ by $1/\sum y_i$ , so that $\sum y_i = 1$.
    Evaluate $\hat{\phi}_i$ from pure or mixed Peng Robinson EOS, or set to unity.
    Evaluate $\gamma_i$ from equation (4.21), or set to unity.
    Evaluate $P_1^{\text{sat,corrected}}$ in eq. (4.25), then evaluate $T$ in eq (4.26).
Loop until change in parameters from top of "Do" loop is $<$ threshold.
```

Algorithm for dew point calculation of (x_i, T):

```
Initialize $\hat{\phi}_i$ and $\gamma_i$ to unity.
Initialize $T = \sum y_i T_i^{\text{sat}}$ , where $T_i^{\text{sat}}$ is calculated from Antoine equation.
Evaluate $f_i^l$ from the Antoine equation or equation (4.19).
Solve equation (4.15) for $x_i$.
Evaluate $P_1^{\text{sat,corrected}}$ from eqn (4.25), then evaluate $T$ from eqn (4.26).
Evaluate $f_i^l$ from the Antoine equation or equation (4.19).
Evaluate $\hat{\phi}_i$ from pure or mixed Peng Robinson EOS, or set to unity.
Solve equation (4.15) for $x_i$.
Evaluate $P_1^{\text{sat,corrected}}$ from eqn (4.25), then evaluate $T$ from eqn (4.26).
Do
    Evaluate $f_i^l$ from the Antoine equation or equation (4.19).
    Evaluate $\hat{\phi}_i$ from pure or mixed Peng Robinson EOS, or set to unity.
    Do
        Solve equation (4.15) for $x_i$.
        Multiply each $x_i$ by $1/\sum x_i$ , so that $\sum x_i = 1$.
        Evaluate $\gamma_i$ from equation (4.21), or set to unity.
    Loop until change in params from top of "Do" loop is $<$ threshold.
    Evaluate $P_1^{\text{sat,corrected}}$ in eqn (4.25), then evaluate $T$ in eqn (4.26).
Loop until change in parameters from top of "Do" loop is $<$ threshold.
```

Numerical Methods Used in Chemical Reaction Equilibria Solver

We assume that there are *m* species reacting in a closed system. To solve for the equilibrium composition, we use the Gibbs energy minimization technique[2]. Each species is in gas or solid phase. Initially, the number of moles of each species is $n_1,...,n_m$. Assume that there are *k* elements represented in the system. We know that the number of moles of each element remains constant, so for the equilibrium composition $n_1^{'},...,n_m^{'}$ we can write a matrix equation of the form

$$\mathbf{A}\begin{bmatrix} n_1^{'} \\ \vdots \\ n_m^{'} \end{bmatrix} = \mathbf{b} \tag{4.27}$$

Where **A** is a $k \times m$ matrix of constants, and **b** is a constant $k \times 1$ column vector. Each row of the matrix represents the mole conservation formula for the given element. To minimize the total Gibbs energy, a vector of Lagrange multipliers $\boldsymbol{\lambda} = (\lambda_1,...,\lambda_k)$ can be introduced, giving another *m* equations:

$$\Delta g_i^f + RT\ln\frac{\hat{f}_i}{f_i^0} + A_i\boldsymbol{\lambda} = 0 \qquad i = 1,...,m \tag{4.28}$$

$$\frac{\hat{f}_i}{f_i^0} = \begin{cases} y_i P, & \text{if species } i \text{ in gas phase} \\ 1, & \text{if solid phase} \end{cases} \tag{4.29}$$

Here A_i denotes the *i*th column of matrix **A**, Δg_i^f is the Gibbs energy of formation at the given temperature, and y_i is the vapor mole fraction of species *i*, which can be computed from the equilibrium number of moles $n_1^{'},...,n_m^{'}$. In solving for the equilibrium, we fix *P* and *T* as constants, so the Gibbs' energies of formation can be precomputed, and we have a system of $m+k$ equations in $m+k$ variables which is almost linear: if the log terms are treated as constants then the system is linear. To account for the constraints $n_1^{'} \geq 0,\ldots,n_m^{'} \geq 0$ we write $n_i^{'} = \exp(\eta_i^{'})$ and solve for the $m+k$ variables $\eta_1^{'},\ldots,\eta_m^{'}$, $\lambda_1,\ldots,\lambda_k$.

To solve the system, we define a vector-valued function $\mathbf{F} : \mathbf{R}^{m+k} \mapsto \mathbf{R}^{m+k}$ which takes as inputs the variables $\eta_1^{'},\ldots,\eta_m^{'},\lambda_1,\ldots,\lambda_k$ and returns as outputs the difference between the right-hand side and the left-hand side of equations (4.27) and (4.28). We then use the following algorithm:

```
Initialize $\lambda_i$ to zero.
Initialize $\eta_i^{'}$ to $\ln(n_i)$.
Repeat 4 times:
    Use simplex method to modify $\eta_i^{'}$ so $\|\mathbf{F}\|^2$ is minimized (200 iters).
    Use simplex method to modify $\lambda_i$ so $\|\mathbf{F}\|^2$ is minimized (200 iters).
    Use simplex method to modify $\eta_i^{'}$, $\lambda_i$ to minimize $\|\mathbf{F}\|^2$ (500 iters).
    Use modified Newton's method to find a root of $\mathbf{F}$ (150 iters).
```

The idea behind this algorithm is that the modified Newton's method converges slowly when far from a root, while the simplex method converges slowly while close to a root, so by alternating between the two methods, we obtain faster convergence.

V. INTEGRATION WITH TEXTBOOK

We have integrated the software with the textbook by Koretsky in two main ways: in-chapter examples use ThermoSolver to perform calculations, and end of chapter exercises ask students to use ThermoSolver. Some charts and tables in the book were also generated by the software, including the generalized correlation tables, compressibility charts, and a comparison of how ThermoSolver's g^E model parameter fitting solver compares against the model parameters determined by the standard DECHEMA reference book.

Integration with In-chapter Examples and End of Chapter Exercises

Some of the more complicated in-chapter examples derive the relevant equations and then state that ThermoSolver can be used to solve the equations and evaluate constants such as $\Delta h^0_{\text{rxn},298}$. For example, ThermoSolver is used to compute a saturation pressure, used to find the equilibrium constant for a fuel cell reaction, and used to solve a cracking of methane problem. The software is particularly helpful in these problems due to the built-in database and the ease with which various thermodynamic constants can be computed.

Sixteen end-of-chapter problems in the textbook ask the student to use ThermoSolver. The problems typically ask the student to compare results calculated by hand with results obtained from ThermoSolver. The textbook problems exercise the capabilities of ThermoSolver, with problems such as the calculation of state properties from various equations of state, the calculation of fugacity coefficients, fitting of parameters for g^E models to experimental data, etc.

Comparison Table of DECHEMA and ThermoSolver g^E Models

From the DECHEMA data collection, we selected 8 different classes of vapor liquid equilibrium systems (such as aliphatic hydrocarbons and aromatic hydrocarbons), and then randomly chose two binary VLE data sets from each class[12]. We chose from the isothermal data sets. We then used ThermoSolver to find the best parameters for the two-suffix Margules, three-suffix Margules, Wilson, van Laar, and NRTL models. We used ThermoSolver's objective function which minimizes relative error in γ_a and γ_b. The best-fit parameters found by ThermoSolver and the parameters provided by DECHEMA were then compared, by taking the difference between the pressure predicted by the model and the pressure measured at each experimental data point. The mean and maximum deviation in pressure were entered into a comparative table in the textbook.

For ten of the sixteen data sets, for every activity coefficient model, the mean and maximum deviation in pressure for the ThermoSolver fit and the DECHEMA fit differed by less than a factor of two. That is, ThermoSolver and DECHEMA provided fit parameters of roughly equal quality. For the remaining six data sets, three had significantly less pressure deviation for the ThermoSolver fit (hexane-benzene, pyridine-nonane, and toluene-chlorobenzene), and three had significantly less pressure deviation for the DECHEMA fit (water-acetic acid, chlorobenzene-propionic acid, and methyl acetate-ethyl acetate). In two of the systems containing acids, ThermoSolver's pressure deviations are presumably higher because DECHEMA corrects for gas-phase ion-molecule association whereas ThermoSolver does not.

Generalized Correlation Tables and Compressibility Charts

In the textbook by Koretsky, the charts and tables of compressibility, enthalpy departure from ideal gas, entropy departure from ideal gas, and fugacity coefficients are computed by the textbook software. The phase transitions in the tables occur along a fixed curve $P_r = f(T_r)$ (based on the $P_r^{\text{sat}(0)}$ term in equation (4.6)), while in reality different species have different phase transition curves that can be more accurately modeled by the Antoine equation.

VI. CONCLUSION

Our two main objectives in this thesis were to create an easy to use graphical interface and develop the numerical algorithms needed to solve the thermodynamic equations. We have met the former goal by developing a graphical program where each window includes directions for the student, using point and click navigation, integrating a database so that students need not look up physical properties, and including documentation with the program. Two small usability problems remain: the method of selecting species from the drop-down list boxes in the program is cumbersome, and one cannot currently open more than one solver window at a time. We intend to fix these issues with a future version of ThermoSolver which will be released in the summer of 2006. In most solvers, the numerical algorithms work as tested. However, in the parameter fitting for g^E models solver, for a few rare data sets, the solver may produce a bad fit. This behavior is due to the minimization algorithm finding a local minimum instead of the global minimum. This problem can currently be remedied by the user: one may adjust the model's coefficients to a new starting "guess" and click Solve again to get a good fit. In the chemical reaction equilibria solver, the algorithm will occasionally not converge if certain correction factors are chosen. This problem can again be circumvented by the user: one can select different correction factors and click Solve, so that a valid solution is found, and then move back to the desired correction factors. In the future we intend to fix this problem by using a more efficient root finder such as the Levenberg-Marquardt algorithm, or a constrained root finder[15].

Another goal of our program was to make thermodynamics easier for students with no programming knowledge to explore and understand. We believe that we have accomplished this goal, as ThermoSolver has been widely used by chemical engineering students at Oregon State University, and indeed has been used by many of the students in a chemical plant design course where ThermoSolver was not specifically indicated as a tool which the students should use. ThermoSolver provides a significant number of features with a fairly low "barrier of entry" to the user.

ThermoSolver could be improved by incorporating some of the features found in other software, as presented in Chapter 2. A general-purpose equation solver may be useful for solving problems which incorporate the thermodynamic properties of individual species or mixtures into a larger system. One could also extend the solvers to accept vector inputs in addition to scalars as in MATLAB. This added feature would allow solution of the same problem for every element in the vector, thus yielding solutions over specified ranges of variables.

BIBLIOGRAPHY

1. Barnes, C., and Koretsky, M. *Engineering and Chemical Thermodynamics, Student Companion Site*. 2004. Retrieved May 26, 2006, from http://www.wiley.com/college/koretsky .

2. Koretsky, M. *Engineering and Chemical Thermodynamics*, 1st Ed., Wiley (2003).

3. Elliott, J., and Lira, C. *Introductory Chemical Engineering Thermodynamics*, 1st Ed., Prentice Hall PTR (1999).

4. Sandler, S. *Chemical and Engineering Thermodynamics*, 3rd Ed., Wiley (1999).

5. Kyle, B. *Chemical and Process Thermodynamics*, 3rd Ed., Prentice Hall PTR (1999).

6. Çengel, Y., and Boles, M. *Thermodynamics: An Engineering Approach*, 4th Ed., McGraw-Hill Sci./Eng./Math (2001).

7. Smith, J., Van Ness, H., and Abbott, M. *Introduction to Chemical Engineering Thermodynamics*, ed. 5, New York: McGraw-Hill (1996).

8. Reid, R., Prausnitz, J., and Sherwood, T. *The Properties of Gases and Liquids*, 3rd Ed., New York: McGraw-Hill (1977).

9. NIST WebBook. Available: http://webbook.nist.gov/ .

10. Stull, D., Westrum, E., and Sinke, G. *The Chemical Thermodynamics of Organic Compounds*, New York: Wiley (1969).

11. Ohe, S. *Vapor-Liquid Equilibrium Data*, New York: Elsevier (1989).

12. Gmehling, G., Onken, U., and Alt, W. *Vapor-Liquid Equilibrium Data Collection*, multivolumes, Frankfurt: DECHEMA (1977-1980).

13. Peng, D., and Robinson, D. *A New Two-Constant Equation of State*. Industrial and Engineering Chemistry: Fundamentals. Vol. 15 (1976) 59-64.

14. Lee, B., and Kesler, M. *A Generalized Thermodynamic Correlation Based on Three-Parameter Corresponding States*. AIChE Journal, Vol 21, 510 (1975).

15. Press, W., Teukolsky, S., Vetterling, W., and Flannery, B. *Numerical Recipes in C*, 2nd Ed., Cambridge University Press (1992).

16. Nelder, J. and Mead, R. *A Simplex Method for Function Minimization*, Computer Journal 7 (1965) 308-313.