Proceedings of IMECE2008
2008 ASME International Mechanical Engineering Congress and Exposition
October 31-November 6, 2008, Boston, Massachusetts, USA

IMECE2008-69210

AN OBJECT-ORIENTED ONLINE TOOL FOR SOLVING GENERALIZED CHEMICAL EQUILIBRIUM PROBLEMS

Christopher P. Paolini
San Diego State University
San Diego, CA, USA

Subrata Bhattacharjee
San Diego State University
San Diego, CA, USA

ABSTRACT

Analysis of chemical equilibria is a topic covered in both undergraduate and graduate courses such as physical chemistry, chemical thermodynamics, and engineering thermodynamics. Manual calculation of problems that require a student to solve for species concentrations, partial pressures, or mole fractions usually involve the method of equilibrium constants. Exercises in homework assignments or in-class examinations are frequently limited to reactions that involve no more than four gas phase species as the resulting arithmetic required to solve for the unknown quantity becomes too cumbersome and prone to error. Students who invest the requisite time in manually solving complex equilibrium problems in homework assignments need a tool to verify their answer. A Java web application ("applet") has been developed to assist engineering students who encounter general multiphase equilibrium problems involving many species. In addition to students, educators and professional researchers will benefit from a user friendly and free to use software package that can numerically compute equilibrium distributions for arbitrary reactions. The applet we present in this work can be used to analyze complex reactions involving twenty or more species and one such reaction, the combustion of isooctane and air, is presented as an example.

INTRODUCTION

Students majoring in chemistry, chemical engineering, mechanical engineering, and environmental engineering frequently encounter the topic of chemical equilibrium in physical chemistry or thermodynamics courses, both at the graduate and undergraduate level. An ability to know how to calculate quantities such as concentrations, partial pressures, or mole and mass fractions of species in a reaction that has reached an equilibrium state is a fundamental skill and must be learned by students of chemistry and engineering. On exams and homework assignments, students are often given problems where one is presented with an equilibrium reaction, given some thermochemical data and thermodynamic state information, and asked to use the data to calculate an unknown quantity such as the equilibrium concentration or mole fraction of one or more species. Problems encountered by students in contemporary textbooks [1,2] rarely involve more than four gas phase species which renders the task of manually solving an equilibrium problem more manageable. However, significant arithmetic is required to compute equilibrium constants and use them to solve for unknown species quantities that the probability of mathematical error is quite high when solving such problems by hand. Previous research by Tyson, Treagust, and Bucat [3] document the difficulties students often experience when asked to make predictions about the effect a change in species concentration or temperature has on an equilibrium mixture. Many efforts have been made to develop better tools to help with the understanding and calculation of chemical equilibrium. Many of these efforts have concentrated on the benefit object-oriented design has on software development and have created open source libraries and applications that employ a variety of numerical methods to solve particular classes of equilibrium problems in an object-oriented way [4,5,6]. Other efforts have centered on improving student understanding of chemical equilibrium through user-friendly and visually appealing applications that are web accessible using the java applet interface [7]. Unfortunately, all of these tools are either problem specific and focus on particular, predefined reactions or require substantial technical skill and knowledge to adapt the software to solve arbitrary problems faced by students and researchers. The goal of this work is to draw on the past contributions of others and create a web accessible and freely available application that can be used by students and researchers to solve any generalized chemical equilibrium problem.

NOMENCLATURE

a	Number of different elements in the reacting system
$A_{i,j}$	Number of atoms of type i in ideal gas j
b_j	Number of atoms of type j in the reacting system
C	The set of condensed species in the product mixture
j	Index used to reference a gaseous species
K_p	Partial pressure equilibrium constant
l	Index used to reference a condensed species
m	Number of unique species in the product composition
n_j	Number of moles of species j
n	Total number of moles in the equilibrium composition $= \sum_{j=1}^{m} n_j$
(g)	Species in the gas phase
(l)	Species in the liquid phase
H	Enthalpy, kJ
T	Temperature, K
S	Entropy, kJ·K^{-1}
U	Internal energy, kJ
P	Pressure, atm
P_j	Partial pressure of species j, atm
P°	Standard state pressure = 1 atm
R	Ideal gas constant = 8.314472 J·K^{-1}·mol^{-1}
V	Volume, m^3
y_j	Mole fraction of species j
ϕ	Equivalence ratio
μ_j	Chemical potential of the j^th species, J·mol^{-1}
μ_j°	Standard state chemical potential of the j^th species, J·mol^{-1}

A JAVA BASED WEB APPLICATION

To illustrate the educational advantage of our chemical equilibrium application, we have selected two problems to solve to demonstrate its suitability for use in engineering courses. One of the first reactions students learn during the study of chemical equilibrium thermodynamics is the Haber process for ammonia synthesis

$$N_2(g) + 3H_2(g) \rightleftharpoons 2NH_3(g) \qquad (1)$$

The Haber process is carried out at about 520°C and 500 atm in the presence of an iron-molybdenum catalyst. The catalyst increases the rate of the reaction given by (1) but does not affect the reaction stoichiometry. At equilibrium, the mole fractions of nitrogen, hydrogen, and ammonia gas are approximately 17%, 50%, and 33%. Students learn to solve this type of problem manually using the method of equilibrium constants. The partial pressure equilibrium constant K_p for (1) at 500°C is 1.066×10^{-5}. We can solve for the equilibrium mole fractions of N_2, H_2 and NH_3 by equating the known equilibrium constant to the partial pressure quotient

$$K_p = \left[\frac{\prod_{j=1}^{m} P_j^{\nu_j}}{\prod_{i=1}^{n} P_i^{\nu_i}}\right] \qquad (2)$$

where the index j iterates over all product species, i over all reactant species, and $\nu_{i,j}$ is the stoichiometric coefficient for species i or j in the equilibrium reaction. Equation (2) can be expressed in terms of moles where

$$K_p = \left[\frac{\prod_{j=1}^{m} n_j^{\nu_j}}{\prod_{i=1}^{n} n_i^{\nu_i}}\right]\left(\frac{P}{n_{Total}}\right)^{\left[\sum_{j=1}^{m}\nu_j\right]-\left[\sum_{i=1}^{n}\nu_i\right]} \qquad (3)$$

The actual reaction of (1) is given by

$$\begin{aligned} &N_2(g) + 3H_2(g) \longrightarrow \\ &n_{N_2}N_2(g) + n_{H_2}H_2(g) + n_{NH_3}NH_3(g) \end{aligned} \qquad (4)$$

We can then express (3) in terms of exactly one of the n_j unknown molar quantities. By performing a mass balance on nitrogen and hydrogen atoms we obtain three equations in the four unknowns $n_{N_2}, n_{H_2}, n_{NH_3}$ and n_{Total},

$$\begin{aligned} n_{H_2} &= 3n_{N_2} \\ n_{NH_3} &= 2\left(1 - n_{N_2}\right) \\ n_{Total} &= n_{N_2} + n_{H_2} + n_{NH_3} = 2\left(n_{N_2} + 1\right) \end{aligned} \qquad (5)$$

Substituting (5) into (3) we obtain

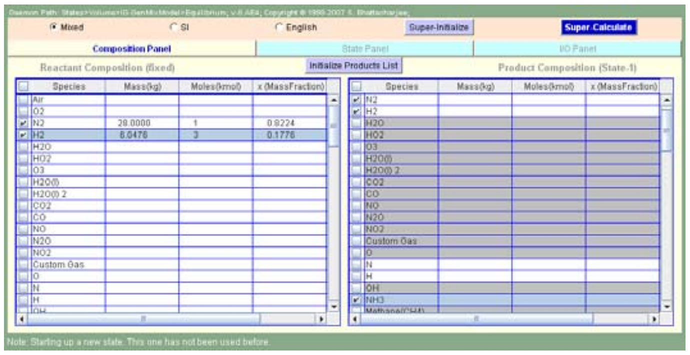
$$K_p = \left[\frac{\left(2\left(1-n_{N_2}\right)\right)^2}{\left(n_{N_2}\right)\left(3n_{N_2}\right)^3}\right]\left(\frac{500}{2\left(n_{N_2}+1\right)}\right)^{-2} = 1.066\times10^{-5} \qquad (6)$$

Solving for n_{N_2} and then using (5) one will find the following mole fractions which are close to the approximate values given above,

$$
\begin{aligned}
n_{N_2}/n_{Total} &= 0.180731 \\
n_{H_2}/n_{Total} &= 0.542194 \qquad (7) \\
n_{NH_3}/n_{Total} &= 0.277075
\end{aligned}
$$

With only three gaseous species and two elements, equation (6) is a fairly difficult and time consuming equation to solve manually. The equilibrium composition of reactions that involve three or four species can usually be hand calculated with reasonable effort using equilibrium constants. However, in reactions involving many species (e.g. combustion processes), using equilibrium constants is not practical and specialized software is required.

We now show the ease with which (1) can be solved using our Java Chemical Equilibrium Daemon. In Figure 1 we see a snapshot of the composition panel where the reactant and product species are specified. One checks the box next to each desired reactant species and then double-clicks in either the kmol or kg column to specify moles or mass, respectively. To solve the Haber ammonia synthesis problem just presented, one would check species N_2 and H_2 and then enter 1 and 3, respectively, in the corresponding kmol column of the *Reactant Composition* table. Next, since the problem specifies the mixture in equilibrium consists only of N_2, H_2 and NH_3, one checks the corresponding boxes next to each of these species in the *Product Composition* table.


Daemon Path: States>Volume>IG-GenMixModel>Equilibrium; v.8.AE4; Copyright © 1998-2007 S. Bhattacharjee;

(● Mixed) SI English

Reactant Composition (fixed)

	Species	Mass(kg)	Moles(kmol)	x (MassFraction)
	Air			
	O2			
✓	N2	28.0000	1	0.8224
✓	H2	6.0476	3	0.1776
	H2O			
	HO2			
	O3			
	H2O(l)			
	H2O(l) 2			
	CO2			
	CO			
	NO			
	N2O			
	NO2			
	Custom Gas			
	O			
	N			
	H			
	OH			

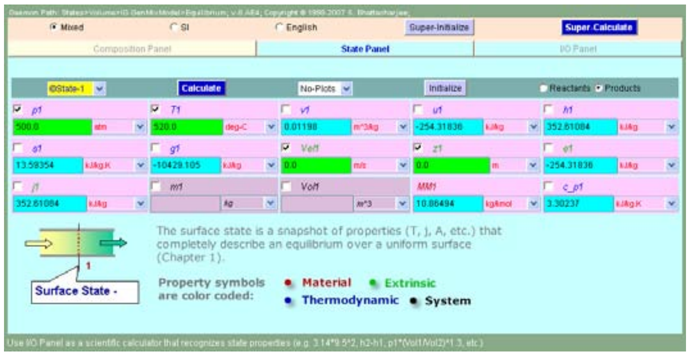
Product Composition (State-1)

	Species	Mass(kg)	Moles(kmol)	x (MassFraction)
✓	N2			
✓	H2			
	H2O			
	HO2			
	O3			
	H2O(l)			
	H2O(l) 2			
	CO2			
	CO			
	NO			
	N2O			
	NO2			
	Custom Gas			
	O			
	N			
	H			
	OH			
✓	NH3			
	Methane(CH4)			

Note: Starting up a new state. This one has not been used before.

Figure 1. Selecting reactant and product species using the Composition Panel.

Once the reactant and product species have been configured, temperature and pressure is then specified using the *State Panel* as shown in Figure 2. The thermodynamic state of the "frozen" reactant mixture can be calculated by clicking the radio button entitled *Reactants* in the *State Panel* and then clicking *Calculate*. The thermodynamic state of the equilibrium composition is found by clicking the radio button entitled *Products* and then clicking *Calculate* again. From Figure 2 one can see the calculated frozen state specific enthalpy (h_1) and entropy (s_1) is $352.61\ \text{kJ}\cdot\text{kg}^{-1}$ and $13.59\ \text{kJ}\cdot\text{kg}^{-1}K^{-1}$, respectively. Specifying any two intrinsic thermodynamic properties (shown in green) in the *State Panel* will automatically compute all other properties (shown in aquamarine). After computing the equilibrium distribution from the *State Panel*, the molar and mass distribution quantities can be found by returning to the *Composition Panel* as shown in Figure 3. One can see from the highlighted quantities the mole fractions of N_2, H_2 and NH_3 in the equilibrium composition are 0.1809430, 0.5428226, and 0.2762344, respectively. These values closely match the expected values obtained from our manual solution given in (7) using the equilibrium constant.


Daemon Path: States>Volume>IG-GenMixModel>Equilibrium; v.8.AE4; Copyright © 1998-2007 S. Bhattacharjee;

(● Mixed) SI English

Reactants (● Products)

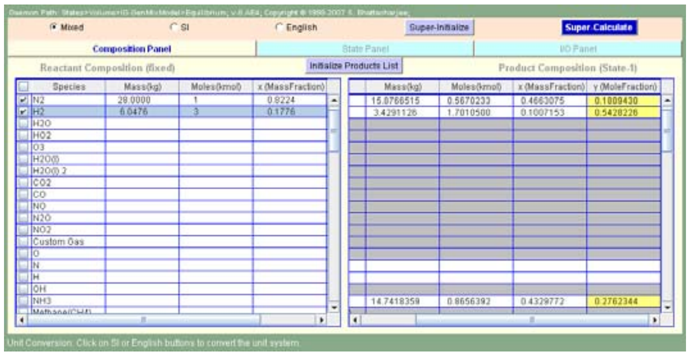
✓ p1: 500.0 atm	✓ T1: 520.0 deg-C	v1: 0.01198 m^3/kg	u1: -254.31836 kJ/kg	h1: 352.61084 kJ/kg
s1: 13.58354 kJ/kg.K	g1: -10429.105 kJ/kg	✓ Vel1: 0.0 m/s	✓ z1: 0.0 m	e1: -254.31836 kJ/kg
j1: 352.61084 kJ/kg	m1: kg	Vol1: m^3	MM1: 10.86494 kg/kmol	c_p1: 3.30237 kJ/kg.K

The surface state is a snapshot of properties (T, j, A, etc.) that completely describe an equilibrium over a uniform surface (Chapter 1).

Property symbols are color coded: ● Material ● Extrinsic ● Thermodynamic ● System

Use I/O Panel as a scientific calculator that recognizes state properties (e.g. 3.14*9.5^2, h2-h1, p1*(Vol1/Vol2)^1.3, etc.)

Figure 2. Thermodynamic properties such as temperature T and pressure p are specified using the *State Panel.*


Daemon Path: States>Volume>IG-GenMixModel>Equilibrium; v.8.AE4; Copyright © 1998-2007 S. Bhattacharjee;

(● Mixed) SI English

Reactant Composition (fixed)

	Species	Mass(kg)	Moles(kmol)	x (MassFraction)
✓	N2	28.0000	1	0.8224
✓	H2	6.0476	3	0.1776
	H2O			
	HO2			
	O3			
	H2O(l)			
	H2O(l) 2			
	CO2			
	CO			
	NO			
	N2O			
	NO2			
	Custom Gas			
	O			
	N			
	H			
	OH			
	NH3			
	Methane(CH4)			

Product Composition (State-1)

Mass(kg)	Moles(kmol)	x (MassFraction)	y (MoleFraction)
15.0766515	0.5670233	0.4663075	0.1809430
3.4291126	1.7010500	0.1007153	0.5428226
14.7418359	0.8656392	0.4329772	0.2762344

Unit Conversion: Click on SI or English buttons to convert the unit system.

Figure 3. After clicking the *Calculate* button with the *Products* radio button selected in the *State Panel*, the molar and mass composition of the product mixture is displayed in the *Composition Panel*.

THEORETICAL BACKGROUND

The theory underlying the computation of a mixture of ideal gases in equilibrium at a constant temperature and pressure is based on the minimization of the Gibbs free energy function. The Gibbs function G is defined as

$$G = H - TS = (U + PV) - TS = \sum_{j=1}^{m} \mu_j n_j \qquad (8)$$

A system will be in equilibrium when the change in Gibbs free energy of the system is zero. Differentiating (8) we obtain

$$dG = dU + PdV + VdP\text{-}TdS\text{-}SdT \tag{9}$$

At a constant temperature T and pressure P, (9) reduces to

$$dG = dU + PdV \text{ -} TdS \tag{10}$$

The combined first and second law of thermodynamics requires

$$dU - TdS + PdV \leq 0 \Rightarrow dG \leq 0 \tag{11}$$

which tells us the equilibrium state is also one where the Gibbs free energy of a system has reached the smallest possible or minimum value. From (8) and using the definition of the chemical potential for an ideal gas species *j*,

$$\mu_j = \mu_j^\circ + RT \ln\left(\frac{P_j}{P^\circ}\right) \tag{12}$$

Substituting (12) into (8) and dividing by RT to obtain a dimensionless form, we obtain

$$\frac{G}{RT} = \sum_{j=1}^{m}\left[n_j \frac{\mu_j^\circ}{RT} + n_j \ln\left(\frac{P_j}{P^\circ}\right)\right] \tag{13}$$

but

$$\ln\left(\frac{P_j}{P^\circ}\right) = \ln\left(\frac{P_j}{P}\frac{P}{P^\circ}\right) = \ln\left(\frac{y_j P}{P}\frac{P}{P^\circ}\right) = \ln y_j + \ln\left(\frac{P}{P^\circ}\right) \tag{14}$$

where the last term, $\ln\left(\frac{P}{P^\circ}\right)$, reduces to a constant since the system total pressure *P* is given as an input to the problem. Using (14), equation (13) now becomes

$$\frac{G}{RT} = \sum_{j=1}^{m}\left[n_j \frac{\mu_j^\circ}{RT} + n_j \ln y_j + n_j \ln\left(\frac{P}{P^\circ}\right)\right] \tag{15}$$

The minimum stationary point of (8) will be the vector of species molar values $\vec{n}$ where *dG* vanishes. Differentiating *G* we obtain

$$dG = \sum_{j=1}^{m} n_j d\mu_j + \sum_{j=1}^{m} \mu_j dn_j \tag{16}$$

but from the isothermal, isobaric Gibbs-Duhem equation we know that

$$\sum_{j=1}^{m} n_j d\mu_j = 0 \tag{17}$$

and so we seek the unique vector $\vec{n}$ such that

$$\sum_{j=1}^{m}\left[\frac{\mu_j^\circ}{RT} + \ln n_j - \ln n + \ln\left(\frac{P}{P^\circ}\right)\right] dn_j = 0 \tag{18}$$

Solving (18) amounts to solving a nonlinear constrained minimization problem. The method of Lagrange multipliers is used to solve (18) according to an atomic mass constraint

$$\sum_{j=1}^{m} A_{i,j} n_j = b_i \ , \ i = 1, \ldots, a \tag{19}$$

The total number of each atom present in the reactant mixture must be the same in the product mixture at equilibrium since mass is conserved. Because there cannot exist a negative number of moles of a species, we are also bound by the positive moles constraint

$$n_j \geq 0, \forall j, 1 \leq j \leq m \tag{20}$$

We can express the atomic constraint equations given by (19) and (20) as equality constraints

$$\phi_i = \phi_i\left(n_1, n_2, \ldots, n_m\right) = \sum_{j=1}^{m} A_{i,j} n_j - b_i = 0 \tag{21}$$

There will be *a* equations of the form (21) for each atom *i* present in the system. We define the Lagrangian function $\mathscr{L} = \mathscr{L}(\vec{n}, \vec{\lambda})$ as

$$\boldsymbol{L}(\vec{n}, \vec{\lambda}) = \mu(\vec{n}) + \sum_{i=1}^{a} \lambda_i \phi_i \tag{22}$$

The solution of (18) is thus the minimum stationary point where the gradient of (22) vanishes

$$\nabla \boldsymbol{L}(\vec{n}, \vec{\lambda}) = \sum_{j=1}^{m} \left(\frac{\partial \mu}{\partial n_j}\right)_{p,T,n_{i \neq j}} dn_j + \sum_{i=1}^{a} \lambda_i \sum_{j=1}^{m} \frac{\partial \phi_i}{\partial n_j} dn_j = 0 \tag{23}$$

OBJECT-ORIENTED DESIGN

Many of the existing software packages used to solve chemical equilibrium applications are written in a procedural programming language such as FORTRAN and C. For example, NASA CEA [8,9] and STANJAN [10] are both written in FORTRAN 77. Software developed using a procedural programming language is usually more difficult to maintain and extend since, in this paradigm, computational tasks are broken down into functions or subroutines that are designed to accomplish a specific operation. When new functionality is needed, it is usually difficult to refactor existing routines that are specifically designed. In contrast, our eequilibrium applet is written using an object-oriented paradigm with the Java programming language. In this paradigm, computational tasks are implemented within *objects* that model the problem domain. When new functionality is needed, only those classes which model the new capability need to be modified and this can usually be accomplished without having to modify other classes. An additional benefit of using Java is the ability to deliver the application as an applet that can be launched within the context of a web browser. This advantage relieves instructors from having to assist students with software installation issues as any computer with a web browser and the free Java Runtime Environment [11] can run our applet. To illustrate the object-oriented construction of our applet, we will discuss the design of one aspect of the software: the composition panel.

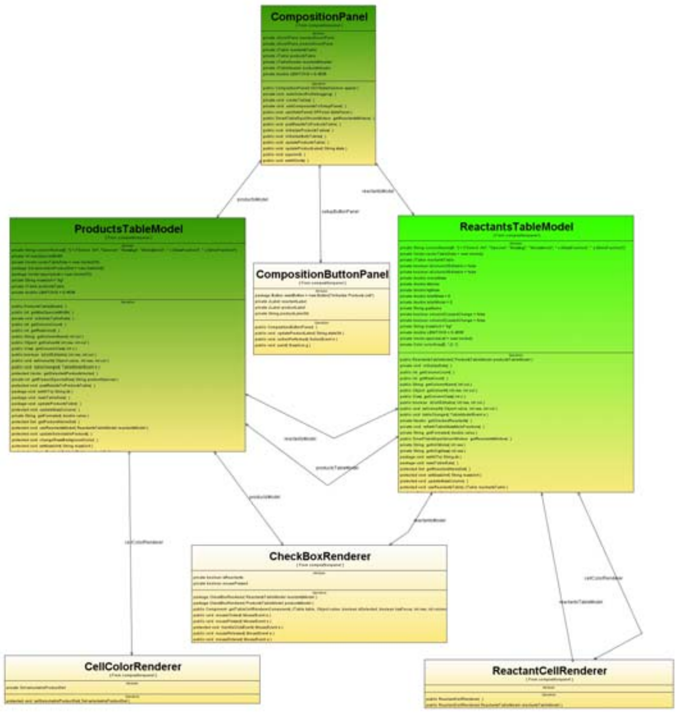


Figure 4. UML class diagram for the Composition Panel.

A UML class diagram [12,13] showing the Java classes involved in the composition panel is shown in Figure 4. The core class is the *CompositionPanel* which is designed using a model-view-controller pattern and has a *ProductsTableModel* and *ReactantsTableModel* class. The model-view-controller (MVC) architectural pattern [14] separates the data (called the model) in an application from the user interface (called the view). The controller code acts as a bridge between the model and view. As events are generated by the user while interacting with the interface, the controller processes these events and invokes actions on the model. In the case of the *CompositionPanel* class which acts as the view, as a user selects different species by clicking the checkbox widgets next to a species' name in the reactants table and specifies the quantity, either by mass [kg] or kmols, the corresponding unit of measurement is automatically calculated from the species' molecular weight as well as the species' mass and mole fractions. The *ReactantCellRenderer* class is used by the *ReactantsTableModel* to designate which unit of measurement was used to designate the quantity of each reactant species: if a user enters the number of kmoles of a reactact, the respective cell in the kmoles column turns green and the cell in the kilograms column turns blue, and vice versa. Each time species are selected or removed from either table, code serving as the controller updates data structures maintained by the *ReactantsTableModel* class that automatically updates mixture properties such as total system mass, moles, and mixture molecular weight which is needed by the solver.

The *ProductsTableModel* class is similar to the *ReactantsTableModel* class in that it serves as a container class to hold the set of species that may exist in a mixture. However, since the quantity of products in a reacting system in a state of equilibrium is unknown to the user prior to computation (this is what is being solved), the *CompositionPanel* class restricts the ability for a user to specify the quantity of any product species. Also, each time a user selects or deselects a reactant species, the *CompositionPanel* class updates the list of selectable species in the products list based on the set of atomic elements represented in the reactant mixture. For example, if a user selects carbon dioxide CO_2 and solid graphite $C(gr)$ as reactants, and then selects carbon monoxide CO, gaseous carbon C, solid graphite $C(gr)$, and oxygen O_2 as products, and then proceeds to unselect carbon dioxide CO_2 from the reactant list, carbon monoxide and oxygen will automatically be removed from the product list since they are no longer atomically feasible product species.

As shown in Figure 2, the state pull-down menu in the state panel identifies a particular thermodynamic state which is defined by entering any two intrinsic thermodynamic properties such as temperature and pressure. A user can define multiple states by pulling down the state menu and selecting an unused state label which is indicated by the pattern *State-n* without a leading copyright (©) symbol. Selecting a particular defined state in the State Panel will select that state in the Composition Panel where the user is then able to select different product species for each defined state. Once multiple states are defined, clicking the blue *Super-Calculate* button, either in the Composition Panel or State Panel, will iteratively compute the equilibrium distribution for each defined state. This convenient capability allows one to solve a multitude of equilibrium problems, each at different temperatures, pressures, and/or product species, all at one time based upon a single reactant mixture. Following a *Super-Calculate* operation, each fully computed state can be viewed by changing the State drop-down table in the State Panel and then clicking on the Composition Panel button.

SOLVING A COMPLEX EQUILIBRIUM PROBLEM INVOLVING MANY SPECIES

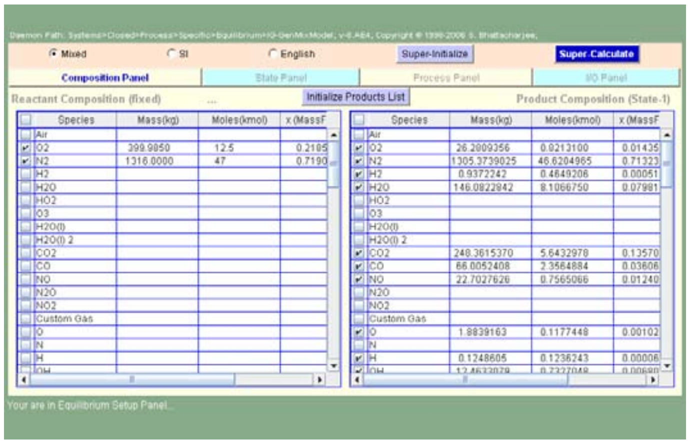
To illustrate the use of our applet to solve a more complex problem, consider the combustion of isooctane and air at $P = 50$ bar and $T = 3000\text{K}$ for a given equivalence ratio ϕ and molar stoichiometric ratio *s*. The equilibrium reaction is

$$C_8H_{18} + \frac{s}{\phi}(O_2 + 3.76N_2) \rightleftharpoons \{CO, O_2, CO_2, H_2O, N_2, NO, OH, O, H_2, C(s), HCN, C_2H_2, H, CH_4, CH_3, CN, C_2N_2, C_2H\}$$

where the molar stoichiometric ratio *s* for octane is 12.5 and the equivalence ratio ϕ is the ratio of the fuel-to-air ratio (FAR) to the stoichiometric fuel-to-air ratio (FAR$_{\text{stoichiometric}}$), or

$$\phi = \frac{FAR}{FAR_{stoichiometric}} = \frac{n_{Fuel}}{n_{oxidizer}} \div \left(\frac{n_{fuel}}{n_{oxidizer}}\right)_{stoichiometric} \tag{24}$$

Thus, for a given equivalence ratio of ϕ and 1 kmol of isooctane fuel, the number of kmols of air would be $12.5/\phi$. Figure 5 shows the result of using our applet to calculate the equilibrium distribution of isooctane combustion at $\phi = 1$. A plot of equilibrium distributions obtained from running our applet for a range of ϕ, $0.4 \le \phi \le 10$, is shown in Figure 6.


Daemon Path: Systems>Closed>Process>Specific>Equilibrium>IG-GenMixModel, v-6.A64, Copyright © 1998-2006 S. Bhattacharjee.
Mixed
SI
English
Super-Initialize
Super Calculate
Composition Panel
State Panel
Process Panel
I/O Panel
Initialize Products List

Reactant Composition (fixed)

	Species	Mass(kg)	Moles(kmol)	x (MassF
	Air			
✓	O2	399.9850	12.5	0.2105
✓	N2	1316.0000	47	0.7190
	H2			
	H2O			
	HO2			
	O3			
	H2O(l)			
	H2O(l) 2			
	CO2			
	CO			
	NO			
	N2O			
	NO2			
	Custom Gas			
	O			
	N			
	H			
	OH			

Product Composition (State-1)

	Species	Mass(kg)	Moles(kmol)	x (MassF
	Air			
✓	O2	26.2809356	0.8213100	0.01435
✓	N2	1305.3739025	46.6204965	0.71323
✓	H2	0.9372242	0.4649206	0.00051
✓	H2O	146.0822842	8.1066750	0.07981
	HO2			
	O3			
	H2O(l)			
	H2O(l) 2			
✓	CO2	248.3615370	5.6432978	0.13570
✓	CO	66.0052408	2.3564884	0.03606
✓	NO	22.7027626	0.7565066	0.01240
	N2O			
	NO2			
	Custom Gas			
✓	O	1.8839163	0.1177448	0.00102
	N			
✓	H	0.1248605	0.1236243	0.00006
✓	OH	12.4633079	0.7327048	0.00680

Your are in Equilibrium Setup Panel...

Figure 5. Combustion of isooctane and air at *P*=50 bar and *T*=3000K.

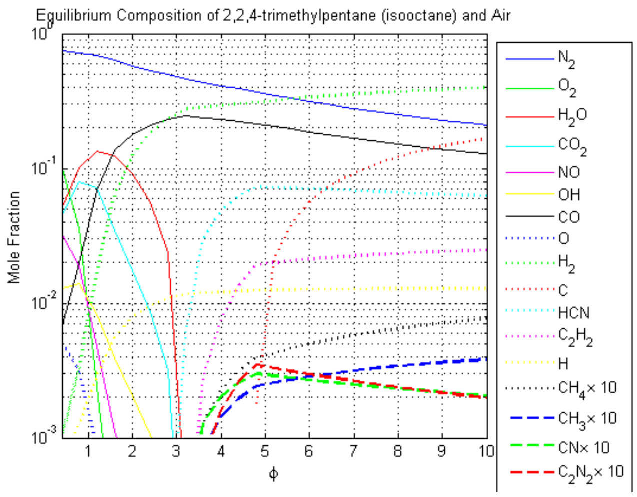


Figure 6. Equilibrium distribution of isooctane and air at *P*=50 bar and *T*=3000K for $0.4 \le \phi \le 10$.

NUMERICAL METHOD

The numerical method employed by our applet used to minimize a Gibbs free energy function with constraints is based on the Newton-Raphson technique used to solve a system of nonlinear equations. By expanding (23) we can write

$$\begin{aligned}
\nabla \boldsymbol{L}(\vec{n},\vec{\lambda}) &= \sum_{j=1}^{m} \left(\frac{\partial \mu}{\partial n_j}\right)_{p,T,n_{i\neq j}} dn_j + \sum_{i=1}^{a} \lambda_i \sum_{j=1}^{m} \frac{\partial \phi_i}{\partial n_j} dn_j \\
&= \sum_{j=1}^{m} \mu_j dn_j + \sum_{i=1}^{a} \lambda_i \left[\frac{\partial \phi_i}{\partial n_1} dn_1 + \frac{\partial \phi_i}{\partial n_2} dn_2 + \cdots + \frac{\partial \phi_i}{\partial n_m} dn_m\right] \\
&= \sum_{j=1}^{m} \mu_j dn_j + \lambda_1 \left[\frac{\partial \phi_1}{\partial n_1} dn_1 + \frac{\partial \phi_1}{\partial n_2} dn_2 + \cdots + \frac{\partial \phi_1}{\partial n_m} dn_m\right] + \\
&\lambda_2 \left[\frac{\partial \phi_2}{\partial n_1} dn_1 + \frac{\partial \phi_2}{\partial n_2} dn_2 + \cdots + \frac{\partial \phi_2}{\partial n_m} dn_m\right] + \cdots + \\
&\lambda_a \left[\frac{\partial \phi_a}{\partial n_1} dn_1 + \frac{\partial \phi_a}{\partial n_2} dn_2 + \cdots + \frac{\partial \phi_a}{\partial n_m} dn_m\right] \\
&= \left[\mu_1 + \lambda_1 \frac{\partial \phi_1}{\partial n_1} + \lambda_2 \frac{\partial \phi_2}{\partial n_1} + \cdots + \lambda_a \frac{\partial \phi_a}{\partial n_1}\right] dn_1 + \\
&\left[\mu_2 + \lambda_1 \frac{\partial \phi_1}{\partial n_2} + \lambda_2 \frac{\partial \phi_2}{\partial n_2} + \cdots + \lambda_a \frac{\partial \phi_a}{\partial n_2}\right] dn_2 + \cdots + \\
&\left[\mu_m + \lambda_1 \frac{\partial \phi_1}{\partial n_m} + \lambda_2 \frac{\partial \phi_2}{\partial n_m} + \cdots + \lambda_a \frac{\partial \phi_a}{\partial n_m}\right] dn_m = 0
\end{aligned} \tag{25}$$

In order to satisfy (25), all the terms in the square brackets must vanish to 0. We therefore seek values of Lagrange multipliers $\vec{\lambda}_i$ such that

$$\mu_j + \sum_{i=1}^{a} \lambda_i \frac{\partial \phi_i}{\partial n_j} = 0 \tag{26}$$

for all *j*. We observe

$$\lambda_i \frac{\partial \phi_i}{\partial n_j} = \lambda_i A_{i,j} \tag{27}$$

and hence solving (25) amounts to solving a system of *m* equations for each species *j* where the *j*th species equation is given by

$$\mu_j + \sum_{i=1}^{a} \lambda_i A_{i,j} = 0 \tag{28}$$

and additionally, *a* population constraint equations for each element *i* where the *i*th constraint equation is given by

$$\sum_{j=1}^{m} A_{i,j} n_j - b_i = 0 \tag{29}$$

and finally one constraint equation governing the total number of moles present in the system,

$$\sum_{j=1}^{m} n_j - n = 0 \tag{30}$$

From (28), (29), and (30), we must solve a system of $m + a + 1$ equations in $a + m + 1$ unknowns. We use the Newton-Raphson method to iteratively solve this system of equations. To construct our system matrix we let $\mathrm{f_j} = \mathrm{f_j}(\vec{x}) = \mathrm{f_i}(x_1, x_2, \ldots, x_n)$ represent the *j*th species equation of (28), or

$$f_j = \mu_j + \sum_{i=1}^{a} \lambda_i A_{i,j} = 0 \tag{31}$$

Expanding μ_j using (15) we have

$$f_j = \frac{\mu_j^{\circ}}{RT} + \ln n_j - \ln n + \ln\left(\frac{P}{P^{\circ}}\right) + \sum_{i=1}^{a} \frac{\lambda_i}{RT} A_{i,j} = 0 \tag{32}$$

By letting

$$x_1 = \ln n_j, x_2 = \ln n, x_3 = -\frac{\lambda_1}{RT}, x_4 = -\frac{\lambda_2}{RT}, \ldots, x_{a+2} = -\frac{\lambda_a}{RT} \tag{33}$$

we will obtain

$$\frac{\partial f_j}{\partial x_1} = \frac{\partial f_j}{\partial(\ln n_j)} = 1, \ \frac{\partial f_j}{\partial x_2} = \frac{\partial f_j}{\partial(\ln n)} = -1, \ \frac{\partial f_j}{\partial x_3} = \frac{\partial f_j}{\partial(-\lambda_1/RT)} = -A_{1,j}, \cdots \tag{34}$$

and our Newton-Raphson equation becomes

$$\sum_{i=1}^{n} \frac{\partial f}{\partial x_i}\delta x_i = -f(\vec{x}) \Rightarrow \delta \ln n_j - \delta \ln n + \sum_{i=1}^{a} A_{i,j}\frac{\lambda_i}{RT} = -\frac{\mu_j}{RT} - \sum_{i=1}^{a}\frac{\lambda_i}{RT}A_{i,j} \tag{35}$$

We will have *m* equations of type (35) for each species *j*. For the population constraint equations given by (29), we let $f_i = f_i(\vec{x}) = f_i(x_1, x_2, \ldots, x_n)$ represent the i^{th} population constraint equation such that

$$f_i = \sum_{j=1}^{m} A_{i,j} n_j - b_i = \sum_{j=1}^{m} A_{i,j} \exp(\ln n_j) - b_i = 0 \tag{36}$$

Then

$$\frac{\partial f_i}{\partial x_j} = \frac{\partial f_i}{\partial(\ln n_j)} = A_{i,j} n_j \tag{37}$$

and our Newton equation for (29) becomes

$$\sum_{j=1}^{m} A_{i,j} n_j \delta \ln n_j = b_i - \sum_{j=1}^{m} A_{i,j} n_j \tag{38}$$

We will have *a* of these equations for each unique atom present in the system. Finally, we accommodate the total system moles equation in a similar fashion by letting $f = f(\vec{x}) = f(x_1, x_2, \ldots, x_n)$ represent (30)**,**

$$f = \sum_{j=1}^{m} \exp(\ln n_j) - \exp(\ln n) = 0 \tag{39}$$

where

$$\frac{\partial f}{\partial x_j} = \frac{\partial f}{\partial(\ln n_j)} = n_j \text{ and } \frac{\partial f}{\partial(\ln n)} = -n \tag{40}$$

and thus the Newton equation for the total system moles constraint becomes

$$\begin{aligned}\sum_{i=1}^{n} \frac{\partial f}{\partial x_i}\delta x_i = -f(\vec{x}) \Rightarrow \sum_{j=1}^{m} n_j \delta \ln n_j - n\delta \ln n = \\ \exp(\ln n) - \sum_{j=1}^{m} \exp(\ln n_j) = n - \sum_{j=1}^{m} n_j\end{aligned} \tag{41}$$

We can represent our system of Newton equations in matrix form by combining (35), (38), and (41) to obtain

$$\begin{bmatrix} 1 & 0 & \cdots & 0 & -A_{1,1} & -A_{2,1} & \cdots & -A_{a,1} & -1 \\ 0 & 1 & \cdots & 0 & -A_{1,2} & -A_{2,2} & \cdots & -A_{a,2} & -1 \\ \vdots & & & & & & & & \vdots \\ 0 & 0 & \cdots & 1 & -A_{1,m} & -A_{2,m} & \cdots & -A_{a,m} & -1 \\ A_{1,1}n_1 & A_{1,2}n_2 & \cdots & A_{1,m}n_m & 0 & 0 & \cdots & 0 & 0 \\ A_{2,1}n_1 & A_{2,2}n_2 & \cdots & A_{2,m}n_m & 0 & 0 & \cdots & 0 & 0 \\ \vdots & & & & & & & & \vdots \\ A_{a,1}n_1 & A_{a,2}n_2 & \cdots & A_{a,m}n_m & 0 & 0 & \cdots & 0 & 0 \\ n_1 & n_2 & \cdots & n_m & 0 & 0 & \cdots & 0 & -n \end{bmatrix} \begin{bmatrix} \delta \ln n_1 \\ \delta \ln n_2 \\ \vdots \\ \delta \ln n_m \\ -\frac{\lambda_1}{RT} \\ -\frac{\lambda_2}{RT} \\ \vdots \\ -\frac{\lambda_a}{RT} \\ \delta \ln n \end{bmatrix} = \begin{bmatrix} -\frac{\mu_1}{RT} - \sum_{i=1}^{a} \frac{\lambda_i A_{i,1}}{RT} \\ -\frac{\mu_2}{RT} - \sum_{i=1}^{a} \frac{\lambda_i A_{i,2}}{RT} \\ \vdots \\ -\frac{\mu_m}{RT} - \sum_{i=1}^{a} \frac{\lambda_i A_{i,m}}{RT} \\ b_1 - \sum_{j=1}^{m} A_{1,j} n_j \\ b_2 - \sum_{j=1}^{m} A_{2,j} n_j \\ \vdots \\ b_a - \sum_{j=1}^{m} A_{a,j} n_j \\ n - \sum_{j=1}^{m} n_j \end{bmatrix} \tag{42}$$

Gordon and McBride show how (42) can be reduced from a rank *a*+*m*+1 system, which is a function of the number of candidate product species in the equilibrium composition, to a rank *a*+1system, which only depends on the number of different elements represented in the composition. For many combustion problems, only 5 elements are present: H, C, N, O, and sometimes S. This reduction fixes our Newton-Raphson system to the relatively low rank matrix that is iteratively solved until the solution converges,

$$\begin{bmatrix} \sum_{j=1}^{m} A_{1,j}A_{1,j}n_j & \sum_{j=1}^{m} A_{1,j}A_{2,j}n_j & \cdots & \sum_{j=1}^{m} A_{1,j}A_{a,j}n_j & \sum_{j=1}^{m} A_{1,j}n_j \\ \sum_{j=1}^{m} A_{2,j}A_{1,j}n_j & \sum_{j=1}^{m} A_{2,j}A_{2,j}n_j & \cdots & \sum_{j=1}^{m} A_{2,j}A_{a,j}n_j & \sum_{j=1}^{m} A_{2,j}n_j \\ \vdots & & & & \vdots \\ \sum_{j=1}^{m} A_{a,j}A_{1,j}n_j & \sum_{j=1}^{m} A_{a,j}A_{2,j}n_j & \cdots & \sum_{j=1}^{m} A_{a,j}A_{a,j}n_j & \sum_{j=1}^{m} A_{a,j}n_j \\ \sum_{j=1}^{m} A_{1,j}n_j & \sum_{j=1}^{m} A_{2,j}n_j & \cdots & \sum_{j=1}^{m} A_{a,j}n_j & \left(\sum_{j=1}^{m} n_j\right) - n \end{bmatrix} \begin{bmatrix} -\frac{\lambda_1}{RT} \\ -\frac{\lambda_2}{RT} \\ \vdots \\ -\frac{\lambda_a}{RT} \\ \delta \ln n \end{bmatrix} = \begin{bmatrix} b_1 - \sum_{j=1}^{m} A_{1,j}n_j + \sum_{j=1}^{m} A_{1,j}n_j \frac{\mu_j}{RT} \\ b_2 - \sum_{j=1}^{m} A_{2,j}n_j + \sum_{j=1}^{m} A_{2,j}n_j \frac{\mu_j}{RT} \\ \vdots \\ b_a - \sum_{j=1}^{m} A_{a,j}n_j + \sum_{j=1}^{m} A_{a,j}n_j \frac{\mu_j}{RT} \\ n - \sum_{j=1}^{m} n_j + \sum_{j=1}^{m} n_j \frac{\mu_j}{RT} \end{bmatrix} \tag{43}$$

The system derived in (43) assumes all species $j, 1 \le j \le m$, are gaseous components and are modeled using the ideal gas equation of state *pV=nRT*. Our species constraint equation given by (28) gives *m* equations (one for each species *j*, *j*=1,…,*m*) in *a* unknown Lagrange multipliers λ_i, *i*=1,…,*a* for *m* ideal gases in the product mixture. If the product mixture contains species in the liquid or solid phase, we must augment our Newton equations to accommodate *l* (letter "ell") species constraint equations for each condensate. The chemical potential of a species undergoing an isothermal process is given by

$$\mu = \mu^{\circ} + RT \ln a \tag{44}$$

where *a* refers to the activity of the species. If we assume a condensed species is incompressible, it can be shown that

$$\ln a \approx \frac{\bar{V}}{RT}(P - 1) \tag{45}$$

In our numerical method we assume $P \approx 1$ atm for each condensate in the product mixture and thus (44) reduces to

$$\mu_l = \mu_l^\circ \tag{46}$$

for each species $l \in C$. To accommodate the presence of condensates in (43), we modify the k^{th} Newton equation to incorporate l constraint equations for each condensate giving

$$\sum_{i=1}^{a}\sum_{j=1}^{m} A_{k,j}A_{i,j}n_j\left(-\frac{\lambda_i}{RT}\right) + \underbrace{\sum_{j=1}^{l} A_{k,j}}_{\substack{\text{Summation}\\ \text{over all}\\ \text{condensed}\\ \text{species}}} + \left[\sum_{j=1}^{m} A_{k,j}n_j\right]\delta\ln n = \tag{47}$$
$$b_k - \sum_{j=1}^{m} A_{k,j}n_j + \sum_{j=1}^{m} A_{k,j}n_j\frac{\mu_j}{RT}$$

For the l additional columns generated by adding the condensed species constraint equations, we add l additional linearly independent equations of the form

$$\sum_{i=1}^{a} A_{i,j}\left(-\frac{\lambda_i}{RT}\right) = \frac{\mu_j}{RT}, \quad j \in C \tag{48}$$

to (43) so our coefficient matrix is not rank deficient. With support for condensed species, the system matrix becomes

$$\begin{bmatrix}
\sum_{j=1}^{m} A_{1,j}A_{1,j}n_j & \sum_{j=1}^{m} A_{1,j}A_{2,j}n_j & \cdots & \sum_{j=1}^{m} A_{1,j}A_{a,j}n_j & A_{1,c_1} & A_{1,c_2} & \cdots & A_{1,c_l} & \sum_{j=1}^{m} A_{1,j}n_j \\
\sum_{j=1}^{m} A_{2,j}A_{1,j}n_j & \sum_{j=1}^{m} A_{2,j}A_{2,j}n_j & \cdots & \sum_{j=1}^{m} A_{2,j}A_{a,j}n_j & A_{2,c_1} & A_{2,c_2} & \cdots & A_{2,c_l} & \sum_{j=1}^{m} A_{2,j}n_j \\
\vdots & & & & & & & & \vdots \\
\sum_{j=1}^{m} A_{a,j}A_{1,j}n_j & \sum_{j=1}^{m} A_{a,j}A_{2,j}n_j & \cdots & \sum_{j=1}^{m} A_{a,j}A_{a,j}n_j & A_{a,c_1} & A_{a,c_2} & \cdots & A_{a,c_l} & \sum_{j=1}^{m} A_{a,j}n_j \\
A_{1,c_1} & A_{2,c_1} & \cdots & A_{a,c_1} & 0 & 0 & \cdots & 0 & 0 \\
A_{1,c_2} & A_{2,c_2} & & A_{a,c_2} & 0 & \ddots & & & 0 \\
\vdots & & \ddots & & \vdots & & \ddots & & \vdots \\
A_{1,c_l} & A_{2,c_l} & & A_{a,c_l} & 0 & & & 0 & 0 \\
\sum_{j=1}^{m} A_{1,j}n_j & \sum_{j=1}^{m} A_{2,j}n_j & \cdots & \sum_{j=1}^{m} A_{a,j}n_j & 0 & 0 & \cdots & 0 & \left(\sum_{j=1}^{m} n_j\right) - n
\end{bmatrix}
\begin{bmatrix}
-\frac{\lambda_1}{RT} \\ -\frac{\lambda_2}{RT} \\ \vdots \\ -\frac{\lambda_a}{RT} \\ \delta n_{c_1} \\ \delta n_{c_2} \\ \vdots \\ \delta n_{c_l} \\ \delta \ln n
\end{bmatrix}
=
\begin{bmatrix}
b_1 - \sum_{j=1}^{m} A_{1,j}n_j + \sum_{j=1}^{m} A_{1,j}n_j\frac{\mu_j}{RT} \\
b_2 - \sum_{j=1}^{m} A_{2,j}n_j + \sum_{j=1}^{m} A_{2,j}n_j\frac{\mu_j}{RT} \\
\vdots \\
b_a - \sum_{j=1}^{m} A_{a,j}n_j + \sum_{j=1}^{m} A_{a,j}n_j\frac{\mu_j}{RT} \\
\frac{\mu_{c_1}}{RT} \\
\frac{\mu_{c_2}}{RT} \\
\vdots \\
\frac{\mu_{c_l}}{RT} \\
n - \sum_{j=1}^{m} n_j + \sum_{j=1}^{m} n_j\frac{\mu_j}{RT}
\end{bmatrix} \tag{49}$$

We use a conjugate-gradient method to solve (49). After each iteration of solving (49), the solution vector,

$$\vec{x}^T = \left[-\frac{\lambda_1}{RT} \quad -\frac{\lambda_2}{RT} \quad \cdots \quad -\frac{\lambda_a}{RT} \quad \delta n_{c_1} \quad \delta n_{c_2} \quad \cdots \quad \delta n_{c_l} \quad \delta \ln n\right]^T \tag{50}$$

is used to construct the vector

$$\overrightarrow{\delta \ln n} = \begin{bmatrix}
\delta\ln n + \sum_{i=1}^{a} A_{i,1}\left(-\frac{\lambda_i}{RT}\right) - \frac{\mu_1}{RT} \\
\delta\ln n + \sum_{i=1}^{a} A_{i,2}\left(-\frac{\lambda_i}{RT}\right) - \frac{\mu_2}{RT} \\
\vdots \\
\delta\ln n + \sum_{i=1}^{a} A_{i,m}\left(-\frac{\lambda_i}{RT}\right) - \frac{\mu_m}{RT}
\end{bmatrix} \tag{51}$$

for all gas phase components. Note the first term in the j^{th} element of (51) is obtained from the last entry in $\vec{x}$. The other two terms represent deviations from the j^{th} gaseous species constraint. For the set of condensed species, the δn components of $\vec{x}$ are extracted to form

$$\overrightarrow{\delta n} = \begin{bmatrix} \delta n_{c_1} \\ \delta n_{c_2} \\ \vdots \\ \delta n_{c_l} \end{bmatrix} \tag{52}$$

From (51) and (52) an under-relaxation factor α is calculated that serves to dampen the step size taken to update the current iterates' molar distribution vector $\vec{n}$. Without an under-relaxation factor, large adjustments in the number of moles (or equivalently, the partial pressures since $p_j = y_j p$) of gaseous species having trace amounts may be taken which could lead to divergence [15]. To calculate the under-relaxation parameter, two candidate sub-parameters, α_1 and α_2, are first calculated and then the smallest value, not to exceed 1, is chosen. Sub-parameter α_1 is calculated only from values in (51) and only for gas phase components presently having a mole fraction greater than 10^{-8} and for which the change in the species moles is found to be increasing from the result of the current iteration (i.e. $\delta \ln n_j > 0$),

$$\alpha_1 = \frac{2}{\max\left(5\left|\delta \ln n\right|, \left|\delta \ln n_1\right|, \left|\delta \ln n_2\right|, \ldots, \left|\delta \ln n_m\right|\right)} \tag{53}$$

Equation (53) restricts the change in the number of moles in the system, or any gas phase component that meets the aforementioned criterion, to e^2. On the other hand, for gaseous species in trace amounts where the log of the species mole fraction is less than $\ln 10^{-8}$ and the change in the moles is also found to be increasing from the result of the current iteration, α_2 is calculated as

$$\alpha_2 = \min\left|\frac{\frac{1}{2}\ln 10^{-8} - \ln\frac{n_j}{n}}{\delta \ln\frac{n_j}{n}}\right| = \min\left|\frac{-\ln n_j + \ln n - 9.21034037197618}{\delta \ln n_j - \delta \ln n}\right| \tag{54}$$

The purpose of α_2 is to prevent gaseous species in trace amounts from growing to have a mole fraction log value greater than $\frac{1}{2}\ln 10^{-8} = \ln 10^{-4}$. For example, from (54) we must have

$$\min\left|\frac{\ln 10^{-4} - \ln y_j}{\delta \ln y_j}\right| = \alpha_2 \le 1 \tag{55}$$

for $y_j \le \ln 10^{-8}$. In the worst case, $\alpha_2 = 1$ and then

$$\left|\delta \ln y_j\right| = \left|\ln 10^{-4} - \ln y_j\right| \tag{56}$$

which limits the increase of the j^{th} gaseous species' mole fraction from exceeding 10^{-4} since the change in the j^{th} gaseous species' mole fraction is set to the difference between 10^{-4} and its present value. The under-relaxation parameter chosen for the current iterate is then the minimum of α_1 and α_2, or 1 if $\alpha_1, \alpha_2 > 1$, and thus

$$\alpha = \min(1, \alpha_1, \alpha_2) \tag{57}$$

Once α is calculated it is used to update the entries of vector $\vec{n}$ which correspond to the log of the molar amount of each gaseous species using the following correction equation,

$$\ln n_j^i = \ln n_j^{i-1} + \alpha(\delta \ln n_j) \tag{58}$$

where superscript i refers the iteration number. Similarly, for the condensed species in $\vec{n}$, the delta values in (52) are used to relax each condensate's molar quantity such that,

$$n_j^i = n_j^{i-1} + \alpha(\delta n_j) \tag{59}$$

Finally, the log of the total system mole count is updated,

$$\ln n^i = \ln n^{i-1} + \alpha(\delta \ln n) \tag{60}$$

After updates (58), (59), and (60) are applied, we take the exponential of the log of the current approximate number of moles of each gaseous species as well as the current approximate number of moles of each condensed species and use these values to re-form the coefficient matrix and right-hand side vector of (49) for the next iteration. We repeat this process until a convergence criterion is satisfied.

Even with under-relaxation, the Newton-Raphson method can fail to converge in some circumstances and so we place an upper limit on the number of iterations allowed and the size of the corrections taken after each iteration. We employ four convergence tests to determine when to terminate the search for the minima. First, based on our own experimentation, we configured a maximum iteration count to be

$$150 + \frac{m+l}{2} \tag{61}$$

where m is the number of possible gaseous species and l is the number of possible condensed species in the product mixture. This setting worked well in the many experiments we conducted. Besides a maximum iteration count, we use the NASA CEA convergence criteria [8] to conclude that the molar distribution which yields the minimum Gibbs function has been found if the following three conditions are satisfied

$$\left|\delta \ln n_j\right| \frac{n_j}{\sum_{j=1}^{m+l} n_j} \le \frac{1}{2}\tau \tag{62}$$

for all gaseous species, and

$$\left|\delta n_j\right| \frac{1}{\sum_{j=1}^{m+l} n_j} \le \frac{1}{2}\tau \tag{63}$$

for all condensed species, and

$$\left|\delta \ln n\right| \frac{n}{\sum_{j=1}^{m+l} n_j} \le \frac{1}{2}\tau . \tag{64}$$

for the total system moles count. The parameter τ is a convergence tolerance and we have used NASA's recommended value of 10^{-5} which yields final molar distribution values accurate to five decimal places. Criteria (62) places an upper bound on the minimum mole fraction correction: as a species mole fraction approaches the upper limit of 1, the magnitude of the correction must be greater than $\frac{\tau}{2}$ to fail the convergence test.

To illustrate the convergence behavior of this method, Figure 7 shows a plot of Gibbs free energy *G* versus the iteration number for the example Haber process for ammonia synthesis reaction presented earlier. One can see from the figure that 18 iterations were required to find the equilibrium distribution that yielded the minimum specific Gibbs free energy of approximately -1.042×10^4 J g^{-1} . This test executed in 1.27s on a Dell Precision 670 workstation with 2×2.80 GHz Intel Xeon™ microprocessors and 3GB of RAM.

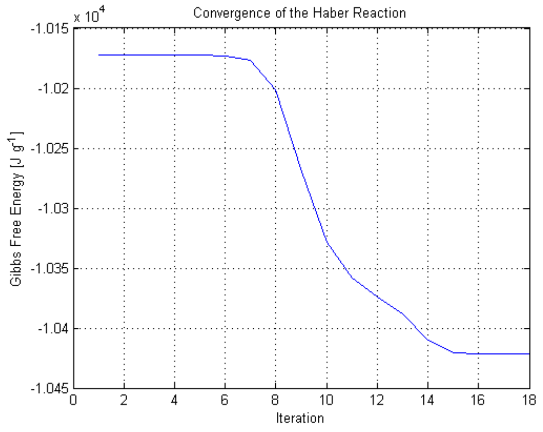


Figure 7. Plot showing the convergence of the system specific Gibbs free energy to a minimum value in the Haber reaction $N_2(g) + 3H_2(g) \rightleftharpoons 2NH_3(g)$

CONCLUSION AND FUTURE DIRECTIONS

Our chemical equilibrium applet has been expanded into two TEST *daemons* [16]: the *Open-System, Steady-State, Chemical Equilibrium Daemon* and the *Closed Process Chemical Equilibrium Daemon*, both currently available from The Expert System for Thermodynamics (TEST) website www.thermofluids.net. Additionally, these applets are available for use, free of charge, to the mechanical engineering community under the *Tools* section of our Computational Thermodynamics Laboratory website cheqs.sdsu.edu. Engineering students and educators can now use a web browser to specify and solve complex chemical equilibrium problems without needing to download, install, and navigate an awkward command-line interface intrinsic to existing codes, pay expensive licensing fees, or struggle with open-source software compilation. Both daemons allow the user to define an initial mixture composition (by mass or moles) and a set of candidate product species for a particular reaction. For a given pressure and temperature, the daemon calculates the equilibrium composition and the product mixture state by minimizing the system's Gibbs free energy function.

Because we used object-oriented design principles to construct our software, we can easily extend our applet to make use of emerging technologies such as W3C standardized Web services [17] to dynamically query thermochemical data such as the standard state entropy s_j° and enthalpy h_j° for each user selected product species. These two thermochemical properties can then be used to compute the chemical potential μ_j of each ideal gas which is equivalent to the Gibbs free energy of the gas or

$$\begin{aligned} g_j &= u_j + pv_j - Ts_j \\ &= h_j - Ts_j \\ &= h^\circ_{f,j,298} + (h_j^\circ - h^\circ_{f,j,298}) - Ts_j^\circ + RT\ln\left(\frac{P_j}{P^\circ}\right) \end{aligned} \tag{65}$$

We also plan to add functionality to allow users to define experimental species by allowing them to supply thermochemical data through the daemon's Configuration Panel which is then uploaded to a central database using Web Services [18]. User submitted thermochemical data would then be immediately available to online users of the daemon, thereby allowing students, educators, and researches to collaborate with one another when solving chemical equilibrium problems.

ACKNOWLEDGMENTS

This work was supported by NSF Office of CyberInfrastructure CI-TEAM Grant 0753283.

REFERENCES

[1] Çengel, Y. A. and Boles, M. A., *Thermodynamics – An Engineering Approach*, 4th Ed., Mc Graw Hill, 2002.

[2] Rock, P. A., *Chemical Thermodynamics*, University Science Books, Mill Valley, California, 1983.

[3] Tyson, L., Treagust, D. F., Bucat, R. B., The Complexity of Teaching and Learning Chemical Equilibrium, *J. Chem. Educ.* **1999** 76 554-558.

[4] Blauch, D.N., Java Classes for Managing Chemical Information and Solving Generalized Equilibrium Problems, *J. Chem. Inf. Comput. Sci.*, **2002**, 42, 2, 143 – 146.

[5] Meeussen, J. C. L., ORCHESTRA: An Object-Oriented Framework for Implementing Chemical Equilibrium Models *Environ. Sci. Technol.* **2003** *(37)*, 6:1175-1182.

[6] Goodwin, D. G., Cantera: Object-oriented software for reacting flows, California Institute of Technology, web site: http://www.cantera.org/, 2008.

[7] Sandberg, M. and Bellamy, M., JCE WebWare Equilibrium, *J. Chem. Educ*. **2003**, *80*, 456.

[8] Gordon, S. and McBride, B. J., *Computer Program for Calculation of Complex Chemical Equilibrium Compositions and Applications – I. Analysis*, NASA Reference Publication 1311, October 1994.

[9] McBride, B. J. and Gordon, S., *Computer Program for the Calculation of Complex Chemical Equilibrium Compositions and Applications – II. Users Manual and* program Description, NASA Reference Publication 1311, June 1996.

[10] Reynolds, W. C., *The Element-Potential Method for Chemical Equilibrium Analysis: Implementation in the Interactive program STANJAN*. Technical Report A-3391, Department of Mechanical Engineering, Stanford University, Palo Alto, CA, 1986.

[11] Sun Microsystems, *Java Runtime Environment*. http://www.java.com/. Accessed November 15, 2007.

[12] Bell, D., *UML Basics: An introduction to the Unified Modeling language*, http://www-128.ibm.com/developerworks/rational/library/769.html, accessed Aug 2006.

[13] *UML Class Diagrams for Java Programmers*, http://www.phptr.com/articles/article.asp?p=336264&rl=1, accessed Aug 2006.

[14] *Model-View-Controller*, Java BluePrints, Sun Microsystems, Inc., 2000-2002, http://java.sun.com/blueprints/patterns/MVC-detailed.html, accessed June 5, 2007.

[15] Zeleznik, F. J. and Gordon, S., Calculation of Detonation Properties and Effect of Independent Parameters on Gaseous Detonation, *J. Am. Rocket Soc.*, Vol.32, No.8, pp.1195-1202, (**1962**).

[16] Bhattacharjee, S., *The Expert System for Thermodynamics: A Visual Tour*. Prentice Hall, April 22, 2002.

[17] Graham, S., Davis, D., Simeonov, S., Daniels, G., Brittenham, P., Nakamura, Y., Fremantle, P., Koenig, D., Zentner C., *Building Web Services with Java : Making Sense of XML, SOAP, WSDL, and UDDI*, 2nd Edition, Sams, June 28, 2004.

[18] Paolini, C. P. and Bhattacharjee, S., A Web Service Infrastructure for Thermochemical Data, *J. Chem. Inf. Model*. **2008**, *accepted to appear.*