International Journal of Thermodynamics (IJoT)
ISSN 1301-9724 / e-ISSN 2146-1511
https://dergipark.org.tr/en/pub/ijot

Vol. 26 (No. 4), pp. 077-083, 2023
doi: 10.5541/ijot.1317496
Published online: December 1, 2023

Research Article

Calculation of Complex Chemical Equilibrium Using Optimization Package Ipopt

[1]*G.V. Belov, [2]N.M. Aristova

[1,2] Joint Institute for High Temperatures of Russian Academy of Sciences, Moscow, Russian Federation, 125412
[1] Bauman Moscow State Technical University, Moscow, Russian Federation, 105005
E-mail: [1]*gbelov@yandex.ru

Received 20 June 2023, Revised 3 September 2023, Accepted 8 October 2023

Abstract

An approach to the calculation of complex chemical equilibrium using the open-source optimization package Ipopt and the open-source package JuMP is proposed. The code of two procedures written in the open-source Julia programming language for calculating the equilibrium composition and properties of multicomponent heterogeneous thermodynamic systems is presented. The results of the test calculations showed a good performance of the code and a relatively high speed of calculations. Due to the compactness and simplicity of the code, it can be easily integrated into other applications, or used in combination with more complex models.

Keywords: *Chemical equilibrium; thermodynamics; Julia; JuMP; optimization package*

1. Introduction

One of the problems that many engineers and scientists facing in chemistry, chemical technology, plasma chemistry, combustion chemistry, gas dynamics, etc. is the need to calculate the equilibrium composition and properties of complex chemically reacting systems. In this article a complex system means a multicomponent heterogeneous thermodynamic system in which chemical and phase reactions are possible.

The calculation of equilibrium composition is a long-standing problem. Perhaps the first attempts to solve it in a general form were associated with the need to compute the characteristics of rocket fuels [1]. A detailed overview of the methods and algorithms used for this purpose is presented in the monograph [2]. Although the problem of calculating the equilibrium of complex thermodynamic systems is quite old [3-7], new developments appear in this area from time to time, see for example [8-11].

Two approaches can be distinguished that are applied to the calculation of the equilibrium composition - the analysis of the equilibrium of possible chemical reactions and the search for the coordinates of the constrained minimum of the thermodynamic potential. The second approach is more general, because it allows the use of well-developed algorithms for the optimization of functions with constraints.

Difficulties in solving the problem of calculating the equilibrium composition of a complex thermodynamic system are due to some of its features. The reliability of the calculation results essentially depends on the completeness of the thermodynamic database; in particular, the database must contain information on the largest possible number of substances in the condensed state. When creating a model of a thermodynamic system, information from the database is usually loaded automatically based on the list of reagents. However, not all phases that are automatically included in the composition of a model thermodynamic system can be present in it in an equilibrium state, and it is not always known in advance which phases these are. Therefore, in the process of calculations, it is necessary to determine not only the chemical but also the phase composition, while the phase rule should not be violated. This means that the mathematical formulation of the problem may contain constraints in the form of inequalities. A model thermodynamic system can contain a large number of substances (about 1000), which means a large problem dimension, while the matrix of indices of chemical elements can be quite large and very sparse. In addition, the calculated values of the equilibrium concentrations of substances vary in a very wide range: from approximately 100 to values of the order of 10^{-100} moles, which is much less than the machine zero.

Perhaps that is why, despite its long history, the problem of calculating the equilibrium composition still attracts the attention of many researchers. To date, there are several large universal software systems equipped with databases on the thermodynamic properties of substances, see, for example, [12, 13]. It is possible to calculate the equilibrium using Microsoft Excel and Matlab [14, 15]. There is also a freely distributed library designed to calculate the equilibrium composition [16, 17].

However, until now, the ready-made packages for solving the constrained optimization problems were very rarely used to calculate the equilibrium composition. A possible reason for this was the inconvenience of using such packages. This paper presents a convenient method for determining the phase and chemical compositions of a complex thermodynamic system using the free Ipopt optimization package [18].

2. Calculation of the Equilibrium Composition by Minimization of the Thermodynamic Potential

From a computational point of view calculation of the equilibrium composition means determining the coordinates

*Corresponding Author

of the conditional extremum of a function of several variables. The number of variables can vary from two to several hundred.

According to Duhem 's theorem [19], in the absence of external fields, the equilibrium state of a thermodynamic system whose initial masses are known is uniquely characterized by the values of two thermodynamic parameters. The most common pairs of parameters are: temperature-pressure (*T*, *p*), temperature-volume (*T*, *V*), enthalpy-pressure (*H*, *p*) for the combustion in a flow type reactor, entropy- pressure (*S*, *p*) for the isentropic expansion to a given pressure, internal energy-volume (*U*, *V*) for the combustion at a constant volume, and entropy-volume (*S*, *V*) for the isentropic expansion to a given volume.

The formulation of the problem in temperature - pressure coordinates is equivalent to determining the coordinates of the conditional minimum of the Gibbs energy *G*

$$\min_{x\in R^n} G(T,p,\mathrm{x})$$
$$T=const, p=const,$$
$$\sum_{i=1}^{N} \nu_{ji}\, x_i = b_j, j=1,\dots,m,$$
$$x_i \geq 0, i=1,\dots \mathrm{N} \qquad (1)$$

where x is the unknown vector of composition, whose components are the numbers of moles of substances, the matrix ν_{ji} defines the number of atoms of a chemical element j in the substance i (so-called formula matrix), N is the number of substances in the system, m is the number of chemical elements, and b_j is the amount of a chemical element j in the system.

One can express the Gibbs energy of a multicomponent heterogeneous system consisting of N_c single- component condensed phases and N_m mixtures as

$$G(T,p,x)=\sum_{i=1}^{N_c} G_i\, x_i + \sum_{j=1}^{N_m}\left[\sum_{i\in I_j} x_i\,(G_i + \mathrm{R}T\ln a_i)\right] \qquad (2)$$

where a_i is the activity of the substance i. In a dimensionless form, this relation can be represented as

$$g(T,p,x)=\sum_{i=1}^{N_c} g_i\, x_i + \sum_{j=1}^{N_m}\left[\sum_{i\in \mathrm{I}_j} x_i\,(g_i + \ln a_i)\right] \qquad (3)$$

where $g = G/\mathrm{R}T$, $g_i = G_i/\mathrm{R}T$.

The following relation is valid for the model «ideal gas, ideal solution, zero volume of condensed phases»

$$g(T,p,x)=\sum_{i=1}^{N_c} g_i\, x_i + \sum_{j=1}^{N_m}\left[\sum_{i\in \mathrm{I}_j} x_i\,(g_i + \ln x_i) - y_j \ln y_j\right] \qquad (4)$$

where $y_j = \sum_{i\in \mathrm{I}_j} x_i$.

The reduced Gibbs energy for the condensed component i is given by

$$g_i = [H_i^\circ(T) - TS_i^\circ(T)]/\mathrm{R}T \qquad (5)$$

and for gaseous substances by

$$g_i = [H_i^\circ(T) - TS_i^\circ(T)]/\mathrm{R}T + \ln(p/p^\circ) \qquad (6)$$

where p° is the standard pressure, $H_i^\circ(T), S_i^\circ(T)$ are the values of enthalpy and entropy of the substance i in the standard state at temperature T.

The formulation of the equilibrium conditions in the temperature-volume coordinates is much less common. In this case, to calculate the equilibrium, it is necessary to determine the coordinates of the conditional minimum of the Helmholtz energy *F*

$$\min_{x\in R^n} F(T,V,\mathrm{x})$$
$$T=const, V=const,$$
$$\sum_{i=1}^{N} \nu_{ji}\, x_i = b_j, j=1,\dots,m,$$
$$x_i \geq 0, i=1,\dots N \qquad (7)$$

One can express the Helmholtz energy of a multicomponent heterogeneous system consisting of N_c single-component condensed phases and N_m mixtures as

$$F(T,V,x)=\sum_{i=1}^{N_c} G_i\, x_i + \sum_{j=1}^{N_m}\sum_{i\in \mathrm{I}_j} x_i\,(G_i + \mathrm{R}T\ln a_i) - pV \qquad (8)$$

Let us assume the mixture with $j=1$ is in the gas phase. The set of indices of substances for this mixture can be denoted as I_g.

$$F(T,V,x)=\sum_{i=1}^{N_c} F_i^\circ\, x_i + \sum_{i\in \mathrm{I}_g} x_i\,(F_i + \mathrm{R}T\ln x_i) + \sum_{j=2}^{N_m}\left[\sum_{i\in \mathrm{I}_j} x_i\,(F_i^\circ + \mathrm{R}T\ln x_i) - \mathrm{R}T y_j \ln y_j\right] \qquad (9)$$

or in a dimensionless form

$$f(T,V,x)=\sum_{i=1}^{N_c} f_i^\circ\, x_i + \sum_{i\in \mathrm{I}_g} x_i\,(f_i + \ln x_i) + \sum_{j=2}^{N_m}\left[\sum_{i\in \mathrm{I}_j} x_i\,(f_i^\circ + \ln x_i) - y_j \ln y_j\right] \qquad (10)$$

The Helmholtz energy for the condensed component i is

$$f_i^\circ = [H_i^\circ(T) - TS_i^\circ(T)]/\mathrm{R}T \qquad (11)$$

for the component i in a gas phase

$$f_i = [H_i^\circ(T) - TS_i^\circ(T)]/\mathrm{R}T + \ln\left[\mathrm{R}T/(p^\circ V)\right] - 1 \qquad (12)$$

3. On the Packages JuMP and Ipopt

An open-source Julia programming language [20] was chosen to implement the above given calculation procedures. This language was designed for scientific and technical calculations. To find the equilibrium composition the packages JuMP and Ipopt were used.

To solve the problem of calculating the equilibrium composition an open-source optimization package Ipopt [21] was chosen. This package is designed to determine the coordinates of the conditional extremum of a non-linear function of many variables using the interior point method. The algorithm implemented in Ipopt is described in [18].

To provide access to a specialized optimization library, an algebraic modeling language is often used. Algebraic modeling languages (AMLs) are designed to describe optimization problems in a form convenient for the researcher. Modeling languages themselves do not solve

optimization problems; their purpose is to transfer the problem formulation to the optimization procedure and return the results of optimization in the most appropriate way. They are widely used in industry and science. Many of these languages are very effective for solving a wide range of problems, but they also have some disadvantages. In particular, using them the formulation of a problem can be very laborious, since in addition to the objective function and constraints, it is necessary to provide the gradient of the objective function, the Jacobian of the constraints, and the Hessian of the Lagrange function. Meanwhile, the most tedious part of this work (namely, calculating the gradient of the objective function, the Jacobian of the constraints, the Hessian of the Lagrange function) can be done by some of AMLs, and JuMP is one of them.

JuMP is an open-source modeling language [22, 23], which allows users to formulate a wide range of optimization problems using high-level algebraic syntax.

As it is mentioned in [23] JuMP is similar to such open-source modeling languages as YALMIP [24], CVX [25] and Pyomo [26]. These AMLs are built into general-purpose programming languages and are convenient to use. However, the low performance of languages such as MATLAB and Python does not allow taking full advantage of these AMLs. To solve this problem, JuMP was created, being integrated with the high-level programming language Julia.

When solving nonlinear optimization problems, AMLs generate procedures for the precise calculation of derivatives according to a given algebraic equation, which optimization packages can call directly. If necessary, automatic differentiation tools can be used to calculate derivatives instead of AML.

The technical tasks that AML must perform can be roughly divided into the following parts: load into memory the problem entered by the user, generate the input data, required by the optimization procedure according to the type of problem, transfer the problem into optimization library and get back the calculation results. All the tasks are solved by the JuMP package, which uses the automatic differentiation to evaluate the derivatives of the expressions entered by the user.

In addition to function gradients, optimization procedures often use matrices of second derivatives. Matrices of this kind can also be computed by the JuMP package using the automatic differentiation technique.

4. Implementation of the Equilibrium Calculation Algorithm in Julia for (*T*, *p*) and (*T*, *V*) Problems

The above equations (1)-(5) were used to create functions in the Julia language, designed to calculate the equilibrium composition. Gibbs and Helmholtz energies divided by *RT* are used as objective functions. The solution of the conditional minimization problem is implemented using the packages JuMP and Ipopt.

The input for the functions includes: the number of chemical elements (*m*), the number of substances (*k*), the number of solutions (*ns*), the number of pure condensed phases (*nc*), the array of dimensionless values of Gibbs (g) or Helmholtz (f) energies of substances, the array of indices of substances in phases-solutions (*jx*), the matrix (A), the amounts of chemical elements in the system (b). The substances in the list are ordered as follows. First, there are condensed substances that form pure phases, then gaseous substances, and then components of condensed solutions.

Each function returns the equilibrium concentrations of substances x_i (`equi_conc`), the numbers of moles of phases y_j (`phase_mols`) and the chemical potentials of elements λ_j (`lam`). In some cases, it is possible to determine the phase composition and equilibrium concentrations of substances present in the system in small amounts approximately only. The code of the functions is given in Figure 1.

```
function calc_Gibbs(m,k,ns,nc,g,jx,A,b)
model = Model(Ipopt.Optimizer)
@variable(model, x[1:k] >= 0, start = 1.e-3)
@variable(model, y[1:ns] >= 0, start = 1.e-3)
@NLobjective(model, Min, sum(x[i]*g[i] for i in
1:nc)+sum(sum(x[i]*(log(x[i]) + g[i]) for i in
jx[1,j]:jx[2,j]) - y[j]*log(y[j]) for j in
1:ns))
for j in 1:ns
@constraint(model, sum(x[i] for i in
jx[1,j]:jx[2,j]) == y[j])
end
@constraint(model, con, A'*x .== b)
JuMP.optimize!(model)
equi_conc=zeros(k)
for i in 1:k equi_conc[i]=value(x[i]) end
phase_mols=zeros(ns)
for i in 1:ns phase_moles[i]=value(y[i]) end
lam=zeros(m)
for i in 1:m lam[i]=shadow_price(con[i]) end
return equi_conc, phase_mols, lam
end

function calc_Helmholtz(m,k,ns,nc,f,jx,A,b)
model = Model(Ipopt.Optimizer)
@variable(model, x[1:k] >= 0, start = 1.e-3)
@variable(model, y[1:ns] >= 0, start = 1.e-3)
@NLobjective(model, Min, sum(x[i]*f[i] for i in
1:nc)+sum(x[i]*(log(x[i]) + f[i]) for i in
jx[1,1]:jx[2,1])+sum(sum(x[i]*(log(x[i]) +
f[i]) for i in jx[1,j]:jx[2,j]) -
y[j]*log(y[j]) for j in 2:ns))
for j in 2:ns
@constraint(model, sum(x[i] for i in
jx[1,j]:jx[2,j]) == y[j])
end
@constraint(model, con, A'*x .== b)
JuMP.optimize!(model)
@show objective_value(model)
equi_conc=zeros(k)
for i in 1:k equi_conc[i]=value(x[i]) end
phase_moles=zeros(ns)
for i in 1:ns phase_mols[i]=value(y[i]) end
lam=zeros(m)
for i in 1:m lam[i]=shadow_price(con[i]) end
return equi_conc, phase_mols, lam
```

Figure 1. The code of the functions calc_Gibbs and calc_Helmholtz.

The function `calc_Gibbs` can be called as it is shown in Figure 2.

```
equi_conc, phase_mols, lam =
calc_Gibbs(m,k,ns,nc,g,jx,A,b)
```

Figure 2. Example of the calling the calc_Gibbs function.

5. Implementation of the Algorithm When the Temperature is not Set

It is impossible to calculate the values of Gibbs and Helmholtz energies if the temperature is not assigned. In this case, it is necessary to find the temperature T as a root of a nonlinear equation

$$Z - \sum_{i=1}^{N} x_i(T) z_i(T) = 0 \quad (13)$$

where Z is the value of assigned parameter (e.g., enthalpy, entropy, internal energy), $x_i(T)$ is the equilibrium concentrations related to current value of the temperature, and $z_i(T)$ is the partial molar value of the parameter (enthalpy, entropy, internal energy respectively).

There are two approaches to solve the problem. The first of them involves the use of a special function to determine the root of the equation from the Roots.jl package [27]. As an example, we provide a function call to calculate the composition at given values of pressure and enthalpy, see Figure 3, where `tmin, tmax` are the upper and lower bounds of the interval for finding the root, `eps` is the maximum calculation error, `find_S` is a function that calculates the entropy of a thermodynamic system from the current values of temperature and chemical composition using information about the thermodynamic properties of substances.

```
T = findrootS(find_S, tmin, tmax,
eps)
```

Figure 3. Example of the calling the findrootS function.

The function `findrootS` is given in figure 4.

```
function findrootS(f,a,b,tol)
T=find_zero(f,[a,b],      atol=tol,
Order1())
return T
end
```

Figure 4. The code of the functions findrootS.

The second approach is based on the direct use of the Newton's method, where the root of the equation is determined iteratively (i is the number of iteration)

$$x_{i+1} \approx x_i - \mathrm{f}(x_i)/\mathrm{f}'(x_i) \quad (14)$$

The temperature is determined by one of the following formulae (C_p and C_v are heat capacities):

$$T_{i+1} \approx T_i + [H - H(T_i)]/C_p(T_i) \quad (15)$$

if the pressure is known and the value of enthalpy H is given;

$$T_{i+1} \approx T_i\left[1 + (S - S(T_i))/C_p(T_i)\right] \quad (16)$$

if the pressure is known and the entropy value S is given;

$$T_{i+1} \approx T_i + [U - U(T_i)]/C_v(T_i) \quad (17)$$

if the volume is known and the value of the internal energy U is given;

$$T_{i+1} \approx T_i\left[1 + \left(S - S(T_i)\right)/C_v(T_i)\right] \quad (18)$$

if the volume is known and the entropy value S is given.

In the case when the temperature is known and the value of entropy S is given, the pressure is determined by the formula

$$p_{i+1} \approx p_i + [S - S(p_i)]/[\partial S(p_i)/\partial p]_T = p_i - [S - S(p_i)]/[\partial v(p_i)/\partial T]_p \quad (19)$$

If the pressure is known, the composition is calculated using the function `calc_Gibbs`, if the volume is specified the procedure `calc_Helmholtz` is used.

6. On the Calculation of Thermodynamic Properties of Substances in the Standard State

To calculate the values of the Gibbs and Helmholtz energies of substances under standard conditions, it is necessary to know the corresponding values of enthalpy and entropy. The reference book [28] and the corresponding database IVTANTHERMO [29] contain information on the thermodynamic properties of pure substances in the form of tables and coefficients of the approximating polynomial a_i, which can be used to determine the values of entropy and enthalpy increment at a given temperature by the following formulas

$$S°(T) = a_1 + a_2(\ln X + 1) - a_3/X^2 + 2a_5X + 3a_6X^2 + 4a_7X^3 \quad (20)$$

$$H°(T) - H°(0) = T(a_2 - 2a_3/X^2 - a_4/X + a_5X + 2a_6X^2 + 3a_7X^3) \quad (21)$$

here $X = T/10000$. The standard pressure p° in the reference book [28] equals 1 atm (101325 Pa).

The enthalpy value can be calculated as follows

$$H^\circ(T) = \Delta_f H^\circ(298.15) + H^\circ(T) - H^\circ(0) - [H^\circ(298.15) - H^\circ(0)] \quad (22)$$

where $\Delta_f H^\circ(298.15)$ is the enthalpy of formation of the substance at 298.15 K.

In the NIST Chemistry Webbook [30] the coefficients of similar relations are given for approximating the temperature dependence of thermodynamic functions in the form

$$S°(T) = A\ln(t) + Bt + Ct^2/2 + Dt^3/3 - E/(2t^2) + G \quad (23)$$

$$H°(T) = H°(298.15) + At + Bt^2/2 + Ct^3/3 + Dt^4/4 - E/t + F - H \quad (24)$$

$t = T/1000$; A, B, C, D, E, F, G, H are the polynomial coefficients, J-mol-K. The standard pressure p° equals 1 bar.

When using NASA polynomials [31], the values of thermodynamic functions at a given temperature are calculated as follows (a_i, b_i are the coefficients)

$$S°(T)/R = -a_1T^{-2}/2 - a_2T^{-1} + a_3\ln T + a_4T + a_5T^2/2 + a_6T^3/3 + a_7T^4/4 + b_2 \quad (25)$$

$$H°(T)/RT = -a_1T^{-2} + a_2(\ln T)/T + a_3 + a_4T/2 + a_5T^2/3 + a_6T^3/4 + a_7T^4/5 + b_1/T \quad (26)$$

7. Results and Discussion

The algorithm described above was implemented as a set of Julia routines available at the GitHub repository [36] under the open-source MIT license. The routines are presented along with their usage examples, which can be run from the command line. As Julia is a multi-platform programming language, the examples can be run on different operating systems. The present version of the code has no GUI interface, although it can easily be created using any of Julia GUI packages. Below in this section, we describe some of the test cases in more detail. In all cases, the information on thermodynamic properties of pure substances from the IVTANTHERMO database [29] has been used.

Calculations were performed for several hundred relatively simple and more complex thermodynamic systems, namely, for homogeneous gas-phase systems with and without ionization, for heterogeneous systems with a gas phase and single-component condensed phases, as well as for complex heterogeneous thermodynamic systems with a gas phase, condensed solutions, and single-component condensed phases. All the problems were solved correctly, although in some cases it was necessary to resort to scaling. For this, the content of chemical elements was recalculated to 1 kg. The most difficult problem included a list of substances formed by 22 chemical elements, while about 130 condensed substances and 135 gaseous substances were selected from the database and included in the system, from which two solutions and several dozen single-component phases were formed. The analysis of the calculation results includes the following checks: the mass balance equation, the Kuhn-Tucker condition, and the Gibbs phase rule. The results of all calculations that were performed with thermodynamic systems of varying complexity were compared with the results of our previous code for calculating the equilibrium composition [32], they were almost identical.

Besides, the calculations were performed at conditions, where the rank of the material balance constraint matrix is less than the number of chemical elements. Usually, it is a difficult task for the numerical algorithms. An example would be systems with a single reactant, such as H_2O or CO_2 at relatively low temperatures, when there is practically only one chemical compound in equilibrium, and the presence of other reaction products is negligible. The program coped with these tasks successfully.

Figure 5 shows the calculation results for the emergency state of a nuclear reactor. Equilibrium concentrations (in mol) are presented for dominant substances only. The initial composition for this calculation is given in [33]:
$1014.5UO_2$ + 0.096Np + 2.754Pu + 0.824Ce + 0.215Y + 0.138%Te + 0.332La + 1.442Zr +0.389Ba + 0.899Ru + 1.15%Mo + 0.265Pr + 0.421Sr + $0.0385I_2$ + 0.859Nd + 0.043%Nb + 0.0064Am + 0.745Cs + 0.166Rh + 0.006Sb + $0.025Eu + 3725H_2O + 3725H_2$.

The calculation was carried out at a pressure of 1 bar and a temperature of 2000 K. A direct comparison of these results with the results of [33] is impossible as the thermodynamic database used in [33] is unavailable. Therefore, we present these simulation results as an illustration of studying a rather complex thermodynamic.

Table 1 shows the computed values of the performance characteristics of rocket fuel liquid oxygen-kerosene (O_2(L)+RP-1), combustion temperature $T_{chamber}$, specific impulse in vacuum I_{vac}, characteristic velocity C^*, obtained using the CEA code [34] and the RENGINE code, written in Julia. The conditions are as follows: the pressure in the combustion chamber is 200 bar, nozzle exit pressure is 0.4 bar, oxidizer/fuel ratio is 3. Some discrepancy in the results can be explained by the fact that different thermodynamic databases were used in the calculations. To compute the characteristics of the propellant, it is necessary to calculate the equilibrium composition in the combustion chamber, in the throat, and at the exit of the nozzle. The calculation of the composition in the combustion chamber is carried out at given values of pressure and enthalpy, the other two calculations are carried out at given values of pressure and entropy [35].

The time for preparing the program for calculation is about several seconds. It includes the time required to load the packages and the database, as well as the time to compile the code. The actual calculation time depends on many factors, namely the number of phases and substances, the type of problem, the type of computer (memory, processor), etc. In our calculations the computation time ranged from several hundredths to tenths of a second if the temperature was given. If the temperature was not set, the calculation time could increase to several seconds. Further optimization of the code is possible.

```
Thermodynamic properties:
p = 0.1          MPa
t = 2000         K
v = 1240.58      cub.m
s = 1972.06      kJ/K
h = -1.38049e+06 kJ
u = -1.50455e+06 kJ
Cp =    428.554     kJ/K
Cv =    366.525     kJ/K
Pure condensed phases
Phase 1:                Ru(c)    0.898991
Phase 2:                Rh(c)    0.164611
Phase 3:                Mo(c)    0.310964
Phase 4:             Nb2O5(c)
    0.0166375
Phase 5:              Y2O3(c)    0.107237
Phase 6:             Ce2O3(c)    0.411914
Phase 7:             Pr2O3(c)    0.13243
Phase 8:             Nd2O3(c)    0.429459
Phase 9:             Eu2O3(c)
    0.0123781
Phase 10:               UO2(c)   1006.11
Phase 11:                Np(c)
    0.0831693
Phase 12:              PuO2(c)   2.75149
Phase 13:            BaZrO3(c)   0.333271
Mixture 1, (gas phase):
              H2(g) 3741.23
             H2O(g) 3698.8
               H(g) 8.59741
           UO2OH(g) 8.00739
              OH(g) 0.843116
            CsOH(g) 0.593765
         Sr(OH)2(g) 0.405405
            LaO2(g) 0.331998
            MoO3(g) 0.301278
       MoO2(OH)2(g) 0.281665
           MoOOH(g) 0.145607
              Cs(g) 0.134998
              Te(g) 0.1149
            U2O6(g) 0.11274
```

Figure 5. Equilibrium composition and properties.

In our opinion, the use of an external optimization library may be appropriate in cases where high performance does not play a crucial role: for research purposes, for testing a thermodynamic model, and for educational purposes. The small size of the functions code makes them versatile,

readable, and ready to use after a short study. The presented functions can be easily integrated into a more complex algorithm that requires the calculation of the equilibrium composition.

Table 1. Computed performance characteristics of rocket fuel.

Program	$T_{chamber}$, K	I_{vac}, м/с	C*, м/с
CEA	3867.2	3574.8	1781.1
RENGINE	3871.7	3575.3	1781.5

8. Conclusion

Two algorithms and their implementations in the Julia programming language are presented for calculating the equilibrium composition and properties of multicomponent heterogeneous thermodynamic systems.

An algorithm for calculating the equilibrium composition for situations with unknown temperature is proposed and implemented.

The test calculations showed a good performance of the functions and a relatively high speed of calculations.

The main advantages of the proposed code are its openness, compactness, versatility, and simplicity, which allows it to be easily integrated into other applications or used in combination with more complex models.

A few small examples illustrating the possibility of using the functions can be found at [36].

Acknowledgements:

This work was supported by the Ministry of Science and Higher Education of the Russian Federation (State Assignment No. 075-01129-23-00).

The authors are grateful to Dr. Igor Morozov for his help in preparing the manuscript.

Nomenclature:

AML - algebraic modeling language;
C_p – heat capacity at constant pressure, J/(mol K);
C_v – heat capacity at constant volume, J/(mol K);
F – Helmholtz energy, J/mol;
G – Gibbs energy, J/mol;
H – enthalpy, J/mol;
I_g – indices of substances in the gas phase;
I_j – indices of substances in the mixture j;
N – number of substances;
R – gas constant, J/(mol K);
S – entropy, J/(mol K);
T – temperature, K;
V – volume, m^3;
a_i – activity of the substance i;
b_j – amount of a chemical element j in the system, mol;
f – dimensionless value of Helmholtz energy;
g – dimensionless value of Gibbs energy;
m – number of chemical elements in the system;
p – pressure, MPa;
x_i – amount of substance i, mol;
y_i – number of moles of phase i, mol;
λ_j – chemical potential of the element j, J/mol;
v_{ji} – number of atoms of a chemical element j in the substance i.

References:

[1] D. S. Villars, “A method of successive approximations for computing combustion equilibria on a high speed digital computer,” *J. Phys. Chem.*, vol. 63, pp. 521-525, Apr. 1959, doi: 10.1021/j150574a016.

[2] W. R. Smith and R. W. Missen, *Chemical Reaction Equilibrium Analysis: Theory and Algorithms*, New York, NY, USA: Wiley, 1982.

[3] W. D. White, S. M. Johnson and G. B. Dantzig, "Chemical equilibrium in complex mixtures," *J. Chem. Phys.*, vol. 28, pp. 751-755, May 1958, doi: 10.1063/1.1744264.

[4] R. J Duffin and C. Zener, "Geometric programming, chemical equilibrium, and the anti-entropy function," *Proc. Natl. Acad. Sci. U.S.A.*, vol. 63, pp. 629-636, Apr. 1969, doi: 10.1073/pnas.63.3.629.

[5] G. Eriksson, "Thermodynamic study of high temperature equilibria," *Acta. Chem. Scand.* vol. 25, pp. 2651-2658, Jul. 1971, doi: 10.3891/acta.chem.scand.25-2651.

[6] B. A. Murtagh and M. A. Saunders, "Large-scale linearly constrained optimization," *Math. Program.*, vol. 14, pp.41-72, Dec. 1978, doi: 10.1007/BF01588950.

[7] H. Greiner, "Computing complex chemical equilibria by generalized linear programming," *Math. Comput. Model.*, vol. 10, pp. 529-550, Jul. 1988, doi: 10.1016/0895-7177(88)90082-9.

[8] M. H. A. Piro and S. Simunovic, "Global optimization algorithms to compute thermodynamic equilibria in large complex systems with performance considerations," *Comput. Mater. Sci.*, vol. 118, pp. 87-96, Jun. 2016, doi: 10.1016/j.commatsci.2016.02.043.

[9] C. Tsanas, E. H. Stenby and W. Yan, "Calculation of multiphase chemical equilibrium by the modified RAND method," *Ind. Eng. Chem. Res.*, vol. 56, pp. 11983–11995, Oct. 2017, doi: 10.1021/acs.iecr.7b02714.

[10] B. Sundman, N. Dupin and B. Hallstedt, "Algorithms useful for calculating multi-component equilibria, phase diagrams and other kinds of diagrams," *Calphad*, vol. 75, p. 102330, Dec. 2021, doi: 10.1016/j.calphad.2021.102330.

[11] W. A. Roos and J. H. Zietsman, “Accelerating complex chemical equilibrium calculations - A review,” *Calphad*, vol. **77**, p. 102380, Jun. 2022, doi: 10.1016/j.calphad.2021.102380.

[12] Available: https://www.factsage.com/ (accessed Aug. 27, 2023).

[13] “Thermo-Calc Software.” Available: https://thermocalc.com/ (accessed Aug. 27, 2023).

[14] Y. Lwin, "Chemical equilibrium by Gibbs energy minimization on spreadsheets," *Int. J. Eng. Educ.* vol. 16, pp.335-339, Apr. 2000.

[15] L. Eriksson, "CHEPP-a chemical equilibrium program package for Matlab," *SAE trans.* , vol. 113, pp. 730-741, 2004.

[16] M. H. A. Piro, S. Simunovic, T. M. Besmann, B. J. Lewis and W. T. Thompson, "The thermochemistry library Thermochimica," *Comput. Mater. Sci.*, vol. 67, pp.266-272, Feb. 2013, doi: 10.1016/j.commatsci.2012.09.011.

[17] “ORNL-CEES / thermochimica.” Available: https://github.com/ORNL-CEES/thermochimica (accessed Aug. 27, 2023).

[18] A. Wächter and L. T. Biegler, “On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming,” *Math. Program.*, vol. 106, pp. 25–57, Mar. 2005. doi: 10.1007/s10107-004-0559-y.

[19] I. Prigogine and R. Defay, *Treatise on thermodynamics: Based on the methods of Gibbs and De Donder*, London: Longmans, 1954.

[20] J. Bezanson, A. Edelman, S. Karpinsky and V. B. Shah, “Julia: A fresh approach to numerical computing,” *SIAM Rev.*, vol. 59, pp. 65-98, Jan. 2017, doi: 10.1137/141000671.

[21] Available: https://coin-or.github.io/Ipopt/index.html (accessed Aug. 27, 2023).

[22] I. Dunning, J. Huchette and M. Lubin, “JuMP: A modeling language for mathematical optimization,” *SIAM Rev.*, vol. 59, pp. 295-320, Feb. 2017, doi: 10.1137/15M1020575.

[23] B. Legat, O. Dowson, J. D. Garcia and M. Lubin, “MathOptInterface: a data structure for mathematical optimization problems,” *INFORMS J. Comput.*, vol. 34, pp. 672-689, Feb. 2022, doi: 10.1287/ijoc.2021.1067.

[24] J. Lofberg, "YALMIP : a toolbox for modeling and optimization in MATLAB," in *Proc. 2004 IEEE Int. Con. on Robotics and Automation (IEEE Cat. No.04CH37508)*, Taipei, Taiwan, Sep. 2004, pp. 284-289, doi: 10.1109/CACSD.2004.1393890.

[25] “CVX: Matlab Software for Disciplined Convex Programming.” Available: http://cvxr.com/cvx/ (accessed Aug. 27, 2023).

[26] W. E. Hart, C. D. Laird, J.-P. Watson, D. L. Woodruff, G. A. Hackebeil, B. L. Nicholson and J. D. Siirola, *Pyomo-optimization modeling in Python.* Berlin: Springer, 2017.

[27] “JuliaMath / Roots.jl .” Available: https://github.com/JuliaMath/Roots.jl (accessed Aug. 27, 2023).

[28] L. V. Gurvich and I. V. Veyts, *Thermodynamic Properties of Individual Substances*, New York: Hemisphere Publishing Corp., 1989.

[29] G. V. Belov, S. A. Dyachkov, P. R. Levashov, I. V. Lomonosov, D. V. Minakov, I. V. Morozov, M. A. Sineva and V. N. Smirnov, “The IVTANTHERMO — online database for thermodynamic properties of individual substances with web interface,” *J. Phys.: Conf. Ser.*, vol. 946, p. 012120, 2018, doi: 10.1088/1742-6596/946/1/012120.

[30] “NIST Chemistry WebBook, SRD 69.” Available: webbook.nist.gov (accessed Aug. 27, 2023).

[31] B. J. McBride, “NASA Glenn coefficients for calculating thermodynamic properties of individual species,” NASA, Cleveland, Ohio, USA, Tech. Rep. NASA/TP-2002-211556, Sep. 2002. Available: https://ntrs.nasa.gov/api/citations/20020085330/downloads/20020085330.pdf (accessed Aug. 27, 2023).

[32] G. Belov, “On linear programming approach for the calculation of chemical equilibrium in complex thermodynamic systems,” *J. Math. Chem.*, vol. 47, pp. 446-456, Jan. 2010, doi: 10.1007/s10910-009-9580-y.

[33] A. D. Pelton, "Thermodynamic modeling and phase equilibrium calculations in nuclear materials," *Pure Appl. Chem.*, vol. 69, pp. 2245-2252, Nov. 1997, doi: 10.1351/pac199769112245.

[34] “CEARUN.” Available: https://cearun.grc.nasa.gov/ (accessed Aug. 27, 2023).

[35] G. P. Sutton and O. Biblartz, *Rocket Propulsion Elements*, New York, NY, USA: Wiley, 2017.

[36] “gvbelov / Heterogeneous-Equilibrium .” Available: https://github.com/gvbelov/Heterogeneous-Equilibrium (accessed Aug. 27, 2023).