Computation of Phase Equilibrium and Phase Envelopes

Ritschel, Tobias Kasper Skovborg; Jørgensen, John Bagterp

Publication date:
2017

Document Version
Publisher's PDF, also known as Version of record

Computation of Phase Equilibrium and Phase Envelopes

Abstract

In this technical report, we describe the computation of phase equilibrium and phase envelopes based on expressions for the fugacity coefficients. We derive those expressions from the residual Gibbs energy. We consider 1) ideal gases and liquids modeled with correlations from the DIPPR database and 2) nonideal gases and liquids modeled with cubic equations of state. Next, we derive the equilibrium conditions for an isothermal-isobaric (constant temperature, constant pressure) vapor-liquid equilibrium process (PT flash), and we present a method for the computation of phase envelopes. We formulate the involved equations in terms of the fugacity coefficients. We present expressions for the first-order derivatives. Such derivatives are necessary in computationally efficient gradient-based methods for solving the vapor-liquid equilibrium equations and for computing phase envelopes. Finally, we describe a Matlab program that computes the phase envelope of a mixture. We present the source code and discuss practical details of the implementation.

Tobias K. S. Ritschel
John Bagterp Jørgensen

DTU Compute Technical Report-2017-11
ISSN: 1601-2321

Technical University of Denmark
Department of Applied Mathematics and Computer Science &
Center for Energy Resources Engineering
Kgs. Lyngby
December 15, 2017

DTU Compute
Department of Applied Mathematics and Computer Science

Center for Energy Resources Engineering
CERE

Contents

1 Introduction

This technical report is structured in the following way: the purpose of Sections 2-4 is to derive expressions for the fugacity coefficients which are relevant to the vapor-liquid equilibrium equations that we present in Section 5 and 7. The vapor-liquid equilibrium equations (and therefore also the fugacity coefficients) are central in the phase envelope computations that we describe in Section 8. In the remainder of this section, we give a brief overview of the content of the sections.

We derive expressions for the residual Gibbs energy for ideal gas and liquid mixtures in Section 2 and for nonideal gas and liquid mixtures in Section 3. The residual Gibbs energy of ideal gases is zero by definition. We obtain the expression for the residual Gibbs energy of ideal liquids directly from the Gibbs energy of the ideal gas and liquid. We construct expressions for the residual Gibbs energy of nonideal mixtures from the residual Helmholtz energy. That is because the partial derivative of Helmholtz energy with respect to volume is negative pressure and because pressure appears explicitly in cubic equations of state. We do not use the expressions for the residual Gibbs energy in the actual computations. We only use them to derive expressions for the fugacity coefficients in Section 4.

In Section 4, we first introduce the chemical potential which is the partial derivative of Gibbs energy with respect to a given mole number. Next, we define the fugacity based on the chemical potential of the mixture of interest and the chemical potential of a pure component ideal gas. We then define the fugacity coefficient and derive an expression for the fugacity coefficient based on the residual Gibbs energy. The fugacity and the chemical potentials are not involved in the actual computations. They only serve to introduce the relation between the fugacity coefficients and the residual Gibbs energy.

In Section 5, we derive vapor-liquid equilibrium conditions for an isothermal-isobaric (constant temperature, constant pressure) vapor-liquid mixture. The second law of thermodynamics states that the entropy of a closed system is maximal at equilibrium. An isothermal-isobaric system is not closed, but the combination of the isothermal-isobaric system and its surroundings is. The condition of maximal entropy of the isothermal-isobaric system and its surroundings is equivalent to a condition of minimal Gibbs energy for the system alone (Callen, 1985). We therefore formulate the vapor-liquid equilibrium problem as an optimization problem where the objective function is the Gibbs energy of the system. The equilibrium conditions are then the first-order optimality conditions of this optimization problem. We rewrite these first-order optimality conditions in terms of the fugacity coefficients.

In Section 6, we introduce the equilibrium constants (sometimes called equilibrium ratios) and use the vapor-liquid equilibrium conditions to derive an expression for the equilibrium constants based on the fugacity coefficients. In Section 7, we use the fact that the equilibrium constants for an ideal vapor-liquid mixture are independent of the composition to solve the vapor-liquid equilibrium problem efficiently.

In section 8, we describe the equations that define what is called the phase envelope. The phase envelope is a collection of vapor-liquid equilibrium states that have the same vapor fraction, i.e. the same total moles of vapor as compared to the total moles of both vapor and liquid. The phase envelope equations are therefore based on the vapor-liquid equilibrium conditions. The method for phase envelope computations that we present is described by Michelsen and Mollerup (2007,

Chap. 12). The method involves a choice of equilibrium constants, temperature, and pressure as independent variables, i.e. the variables that are solved for. The method constructs the phase envelope in a sequential manner for a given specified vapor fraction. It solves the equations for a specified equilibrium constant, temperature, or pressure. Next, it uses the sensitivities of the solution to create an initial estimate for solving the equations with a slightly different specified equilibrium constant, temperature or pressure.

We collect the derivatives of the expressions for the fugacity coefficients, the vapor-liquid equilibrium equations, and the phase envelope equations in Section 9.

In Section 10, we describe, in more detail, the algorithm by Michelsen and Mollerup (2007) for computing the phase envelope. We also describe an example of a Matlab program that implements the algorithm and computes the phase envelope for a hydrocarbon mixture. We present and discuss the programs Matlab code.

1.1 A note on nomenclature

In Sections 2-4, we describe thermodynamic properties of either a vapor or a liquid mixture. When we describe properties of vapor mixtures, e.g. the Gibbs energy of an ideal gas, we denote the composition (in moles) by a vector, n^v. In that case, we denote the mole fraction of component i by y_i. For liquid mixtures, we denote the composition vector n^l and the mole fraction x_i. When we describe the properties of nonideal mixtures, and when we define fugacity and fugacity coefficients, the expressions are the same for both vapor mixtures and liquid mixtures. We therefore denote the composition by n and the mole fraction by z_i.

Later, in Section 5 to 8, we consider mixtures that exist in both a vapor phase and a liquid phase. Again, the vapor phase has composition n^v and mole fraction y_i, and the liquid phase has composition n^l and mole fraction x_i. But in that case, n will denote the composition of the entire mixture, i.e. $n_i = n_i^v + n_i^l$, and z_i will be the corresponding mole fraction.

2 Residual Gibbs energy of ideal vapor-liquid mixtures

We derive the Gibbs energy of ideal gas and liquid mixtures from the enthalpy and entropy of those mixtures. We use the pure component molar enthalpy and entropy of each component in the mixture to derive expressions for the enthalpy and the entropy of the mixtures. The residual Gibbs energy of a mixture is, by definition, the Gibbs energy of that mixture minus the Gibbs energy of an ideal gas at the same state, e.g. temperature, pressure, and composition (in moles). The residual Gibbs energy of an ideal gas mixture is therefore zero by definition. We use the Gibbs energy of an ideal gas mixture and an ideal liquid mixture to derive an expression for the residual Gibbs energy of a liquid mixture.

2.1 Gibbs energy of ideal gas mixtures

We consider an ideal gas mixture of N_C components. The molar enthalpy and entropy of the i'th component at temperature T and pressure P are

$$h_i^{ig}(T) = h_i^{ig}(T_0) + \int_{T_0}^{T} c_{P,i}^{ig}(T)dT, \tag{1a}$$

$$s_i^{ig}(T,P) = s_i^{ig}(T_0,P_0) + \int_{T_0}^{T} \frac{c_{P,i}^{ig}(T)}{T}dT - R\ln\frac{P}{P_0}. \tag{1b}$$

The ideal gas enthalpy is independent of pressure. $h_i^{ig}(T_0)$ and $s_i^{ig}(T_0,P_0)$ are the ideal gas enthalpy and entropy of formation at reference temperature T_0 and P_0. $c_{P,i}^{ig}(T)$ is the ideal gas heat capacity at constant pressure. Values of $h_i^{ig}(T_0)$ and $s_i^{ig}(T_0,P_0)$ and correlations for $c_{P,i}^{ig}(T)$ are available in databases such as the DIPPR database[1] (Thomson, 1996). However, terms that involve $h_i^{ig}(T_0)$, $s_i^{ig}(T_0,P_0)$, and $c_{P,i}^{ig}(T)$ will cancel out in the computations that we describe in this technical report. The molar gibbs energy of component i is

$$\begin{aligned} g_i^{ig}(T,P) &= h_i^{ig}(T) - Ts_i^{ig}(T,P) \\ &= h_i^{ig}(T_0) - Ts_i^{ig}(T_0,P_0) - RT\ln P_0 + \int_{T_0}^{T} c_{P,i}^{ig}(T)dT - T\int_{T_0}^{T}\frac{c_{P,i}^{ig}(T)}{T}dT + RT\ln P \\ &= \Gamma_i(T) + RT\ln P, \end{aligned} \tag{2}$$

where we have introduced the auxiliary variable

$$\Gamma_i(T) = h_i^{ig}(T_0) - Ts_i^{ig}(T_0,P_0) - RT\ln P_0 + \int_{T_0}^{T} c_{P,i}^{ig}(T)dT - T\int_{T_0}^{T}\frac{c_{P,i}^{ig}(T)}{T}dT. \tag{3}$$

$h_i^{ig}(T_0)$, $s_i^{ig}(T_0,P_0)$, and $c_{P,i}^{ig}(T)$ appear only in the auxiliary variable $\Gamma_i(T)$ and it is this variable that will cancel out in the computations in later sections. The enthalpy and entropy of the ideal gas mixture are

$$H^{ig}(T,n^v) = \sum_i n_i^v h_i^{ig}(T), \tag{4a}$$

$$S^{ig}(T,P,n^v) = \sum_i n_i^v s_i^{ig}(T,P) - R\sum_i n_i^v \ln y_i. \tag{4b}$$

n^v is a vector of compositions (in moles). The i'th component, n_i^v, denotes the number of moles of component i. y_i is the vapor mole fraction of component i: $y_i = n_i^v / \left(\sum_j n_j^v\right)$. The sums are over all components. The second term in the expression for the entropy is a mixing term. The Gibbs

[1] www.aiche.org/dippr/events-products/801-database

energy of the ideal gas mixture is

$$\begin{aligned} G^{ig}(T,P,n^v) &= H^{ig}(T,n^v) - TS^{ig}(T,P,n^v) \\ &= \sum_i n_i^v h_i^{ig}(T) - T\sum_i n_i^v s_i^{ig}(T,P) + RT\sum_i n_i^v \ln y_i \\ &= \sum_i n_i^v g_i^{ig}(T,P) + RT\sum_i n_i^v \ln y_i \\ &= \sum_i n_i^v \left(\Gamma_i(T) + RT\ln P\right) + RT\sum_i n_i^v \ln y_i. \end{aligned} \tag{5}$$

2.2 Gibbs energy of ideal liquid mixtures

The vaporization change in molar enthalpy of component i, $\Delta h_i^{vap}(T)$, is related to the vaporization change in molar entropy, $\Delta s_i^{vap}(T)$, by

$$\Delta h_i^{vap}(T) = T\Delta s_i^{vap}(T). \tag{6}$$

We do not present an expression for $\Delta s_i^{vap}(T)$ because it will cancel out in later expressions. The molar ideal liquid enthalpy at saturation is

$$\begin{aligned} h_i^{sat}(T) &= h_i^{ig}(T) - \Delta h_i^{vap}(T) \\ &= h_i^{ig}(T) - T\Delta s_i^{vap}(T). \end{aligned} \tag{7}$$

It is necessary to know the saturation pressure in order to provide expressions for the molar ideal liquid entropy at saturation. We use the following correlation from the DIPPR database

$$P_i^{sat}(T) = \exp\left(\ln P_i^{sat}(T)\right), \tag{8a}$$

$$\ln P_i^{sat}(T) = A_i + \frac{B_i}{T} + C_i\ln(T) + D_i T^{E_i}. \tag{8b}$$

A_i, B_i, C_i, D_i, and E_i are component-specific parameters. They are also specific to this particular correlation. The ideal liquid entropy at saturation is

$$s_i^{sat}(T) = s_i^{ig}(T, P_i^{sat}(T)) - \Delta s_i^{vap}(T). \tag{9}$$

The molar ideal liquid Gibbs energy at saturation is

$$\begin{aligned} g_i^{sat}(T) &= h_i^{sat}(T) - Ts_i^{sat}(T) \\ &= h_i^{ig}(T) - T\Delta s_i^{vap}(T) - Ts_i^{ig}(T, P_i^{sat}(T)) + T\Delta s_i^{vap}(T) \\ &= h_i^{ig}(T) - Ts_i^{ig}(T, P_i^{sat}(T)) \\ &= \Gamma_i(T) + RT\ln P_i^{sat}(T). \end{aligned} \tag{10}$$

In order to provide expressions for the molar ideal liquid enthalpy and entropy at pressures, P, that are different from the saturation pressure, $P_i^{sat}(T)$, it is necessary to know the liquid volume as a function of temperature, as well as its first order temperature derivative. We use the following

correlation from the DIPPR database

$$v_i^l(T) = \frac{B_i^{1+\left(1-\frac{T}{C_i}\right)^{D_i}}}{A_i}. \tag{11}$$

Again, A_i, B_i, C_i, and D_i are component-specific parameters. They are also specific for this correlation, i.e. they are different from the parameters in the correlation for the saturation pressure. The molar ideal liquid enthalpy and entropy are

$$h_i^{id}(T,P) = h_i^{sat}(T) + \left(v_i^l(T) - T\frac{\partial v_i^l}{\partial T}\right)(P - P_i^{sat}(T)), \tag{12a}$$

$$s_i^{id}(T,P) = s_i^{sat}(T) - \frac{\partial v_i^l}{\partial T}(P - P_i^{sat}(T)). \tag{12b}$$

Terms that involve the temperature derivative will cancel out in later expressions. The molar ideal liquid Gibbs energy is

$$\begin{aligned} g_i^{id}(T,P) &= h_i^{id}(T,P) - Ts_i^{id}(T,P) \\ &= h_i^{sat}(T) + \left(v_i^l(T) - T\frac{\partial v_i^l}{\partial T}\right)(P - P_i^{sat}(T)) - Ts_i^{sat}(T) + T\frac{\partial v_i^l}{\partial T}(P - P_i^{sat}(T)) \\ &= h_i^{sat}(T) - Ts_i^{sat}(T) + v_i^l(T)(P - P_i^{sat}(T)) \\ &= g_i^{sat}(T) + v_i^l(T)(P - P_i^{sat}(T)) \\ &= \Gamma_i(T) + RT\ln P_i^{sat}(T) + v_i^l(T)(P - P_i^{sat}(T)). \end{aligned} \tag{13}$$

The ideal liquid mixture enthalpy and entropy are

$$H^{id}(T,P,n^l) = \sum_i n_i^l h_i^{id}(T,P), \tag{14a}$$

$$S^{id}(T,P,n^l) = \sum_i n_i^l s_i^{id}(T,P) - R\sum_i n_i^l \ln x_i. \tag{14b}$$

n^l is a vector of compositions in moles, i.e. n_i^l is the moles of component i. $x_i = n_i^l / \left(\sum_j n_j^l\right)$ is the liquid mole fraction of component i. The second term in the expression for entropy is a mixing term. The ideal liquid mixture Gibbs energy is

$$\begin{aligned} G^{id}(T,P,n^l) &= H^{id}(T,P,n^l) - TS^{id}(T,P,n^l) \\ &= \sum_i n_i^l h_i^{id}(T,P) - T\sum_i n_i^l s_i^{id}(T,P) + RT\sum_i n_i^l \ln x_i \\ &= \sum_i n_i^l g_i^{id}(T,P) + RT\sum_i n_i^l \ln x_i \\ &= \sum_i n_i^l \left(\Gamma_i(T) + RT\ln P_i^{sat}(T) + v_i^l(T)(P - P_i^{sat}(T))\right) + RT\sum_i n_i^l \ln x_i. \end{aligned} \tag{15}$$

2.3 Residual Gibbs energy of ideal vapor and liquid mixtures

The residual Gibbs energy is the difference between the Gibbs energy of the mixture and the Gibbs energy of the mixture if it was considered to be an ideal gas mixture. The residual Gibbs energy of an ideal gas mixture is therefore zero:

$$G^{R,ig}(T,P,n^v) = 0 \tag{16}$$

We use the Gibbs energy of ideal gas mixtures to derive the expression for the residual Gibbs energy of an ideal liquid mixture. It is the difference between the ideal liquid Gibbs energy and the ideal gas Gibbs energy evaluated at the same temperature, T, pressure, P, and composition, n^l:

$$\begin{aligned} G^{R,id}(T,P,n^l) &= G^{id}(T,P,n^l) - G^{ig}(T,P,n^l) \\ &= \left(\sum_i n_i^l \left(\Gamma_i(T) + RT \ln P_i^{sat}(T) + v_i^l(T)(P - P_i^{sat}(T)) \right) + RT \sum_i n_i^l \ln x_i \right) \\ &- \left(\sum_i n_i^l \left(\Gamma_i(T) + RT \ln P \right) + RT \sum_i n_i^l \ln x_i \right) \\ &= \sum_i n_i^l \left(RT \ln \frac{P_i^{sat}(T)}{P} + v_i^l(T)(P - P_i^{sat}(T)) \right) \end{aligned} \tag{17}$$

It is important to note that the ideal gas Gibbs energy is evaluated at the liquid composition, n^l, and not n^v in the above expression. That is also the reason that x_i appears in place of y_i which means that certain terms cancel out. Unlike the ideal gas heat capacity, $c_{P,i}^{ig}(T)$, the saturation pressure, $P_i^{sat}(T)$, and the liquid volume, $v_i^l(T)$, *are* necessary in the computations that we present in this report.

2.4 Summary

The residual Gibbs energy of ideal gas and liquid mixtures are

$$G^{R,ig}(T,P,n^v) = 0, \tag{18a}$$

$$G^{R,id}(T,P,n^l) = \sum_i n_i^l \left(RT \ln \frac{P_i^{sat}(T)}{P} + v_i^l(T)(P - P_i^{sat}(T)) \right). \tag{18b}$$

They will be used in expressions for the fugacity coefficients in Section 4.

3 Residual Gibbs energy of nonideal mixtures

In this section, we first present an expression for the residual Gibbs energy of a nonideal mixture without assuming a specific equation of state. Then we describe cubic equations of state with van der Waals' mixing rules. Finally, we present the expression for the residual Gibbs energy of a nonideal mixture based on a cubic equation of state. The derivation of the expression based on the cubic equation of state is presented in Appendix B. We will not use the expression for the Gibbs energy

of an ideal gas mixture that was derived in Section 2. Instead, we use the ideal gas law for the ideal gas mixture.

3.1 Residual Gibbs energy with pressure-explicit equations of state

In this section, we consider a nonideal vapor or liquid mixture. We treat both phases in the same way. We use the residual Helmholtz energy in the derivation. We will consider the Helmholtz energy a function of temperature, T, volume, V, and composition (in moles), n. This is only for the sake of the derivation. Eventually, we describe the residual Gibbs energy as function of temperature, T, pressure, P, and mole numbers, n. We omit the superscript v or l on the composition vector because we treat both phases in the same way. The residual Helmholtz energy is

$$A^R(T,V,n) = A(T,V,n) - A^{ig}(T,V,n). \tag{19}$$

$A(T,V,n)$ is the Helmholtz energy of the mixture and $A^{ig}(T,V,n)$ is the Helmholtz energy of the mixture if it was an ideal gas. We use the fact that the volume derivative of Helmholtz energy is pressure with a negative sign:

$$\begin{aligned}\frac{\partial A^R}{\partial V}(T,V,n) &= \frac{\partial A}{\partial V}(T,V,n) - \frac{\partial A^{ig}}{\partial V}(T,V,n)\\ &= -\left(P(T,V,n) - P^{ig}(T,V,n)\right).\end{aligned} \tag{20}$$

$P^{ig}(T,V,n)$ is obtained from the ideal gas law, i.e. $P^{ig}(T,V,n) = NRT/V$, where $N = \sum_j n_j$ is the total number of moles in the mixture and R is the gas constant. $P(T,V,n)$ is obtained from an equation of state. The derivation that we present in this section is therefore best suited to equations of state that are explicit in pressure. That is the case for the cubic equations of state that we describe later. We obtain an expression for the residual Helmholtz energy of the nonideal mixture:

$$A^R(T,V,n) = -\int_{V=\infty}^{V} \left(P(T,V,n) - \frac{NRT}{V}\right) dV. \tag{21}$$

The choice of integrating from infinite volume is a matter of convenience. We introduce the compressibility factor, $Z = \frac{PV}{NRT}$. We use relations between residual properties to derive an expression for the residual Gibbs energy evaluated at T, P, and n (Michelsen and Mollerup, 2007, Table 6):

$$\begin{aligned}G^R(T,P,n) &= G^R(T,V,n) - NRT\ln Z\\ &= A^R(T,V,n) + PV - NRT - NRT\ln Z\\ &= -\int_{V=\infty}^{V}\left(P(T,V,n) - \frac{NRT}{V}\right)dV + NRTZ - NRT - NRT\ln Z\\ &= -\int_{V=\infty}^{V}\left(P(T,V,n) - \frac{NRT}{V}\right)dV + NRT(Z-1) - NRT\ln Z.\end{aligned} \tag{22}$$

The above expression only becomes useful when we substitute a specific equation of state for $P(T,V,n)$.

3.2 Cubic equations of state

In this section, we describe cubic equation equations of state and van der Waals' mixing rules. The cubic equations of state are formulated in terms of the molar volume, $v = V/N$:

$$\begin{aligned} P = P(T, V, n) &= \frac{RT}{v - b_m} - \frac{a_m}{(v + \epsilon b_m)(v + \sigma b_m)} \\ &= \frac{RT}{V/N - b_m} - \frac{a_m}{(V/N + \epsilon b_m)(V/N + \sigma b_m)}. \end{aligned} \tag{23}$$

The parameters ϵ and σ are specific for each cubic equation of state. Two of the most popular cubic equations of state were developed by Soave (1972) and by Peng and Robinson (1976). The parameters that are specific to each equation of state are shown in Table 1. a_m and b_m are mixing parameters given by van der Waals' mixing rules:

$$a_m = a_m(T, n) = \sum_i \sum_j z_i z_j a_{ij} \tag{24a}$$

$$b_m = b_m(n) = \sum_i z_i b_i \tag{24b}$$

$z_i = n_i/N$ is the mole fraction of component i. We define the pure component parameter b_i shortly. The binary parameters a_{ij} are

$$a_{ij} = a_{ij}(T) = (1 - k_{ij})\sqrt{\hat{a}_{ij}}. \tag{25}$$

We assume that the binary interaction parameters, k_{ij}, are given constants, i.e. we do not provide an expression for them. They can either be measured experimentally or predicted with a model. The parameter $\hat{a}_{ij}$ is the product of the pure component parameters a_i and a_j:

$$\hat{a}_{ij} = \hat{a}_{ij}(T) = a_i a_j. \tag{26}$$

The pure component parameters a_i and b_i are

$$a_i = a_i(T) = \alpha(T_{r,i}, \omega_i)\Psi \frac{R^2 T_{c,i}^2}{P_{c,i}}, \tag{27a}$$

$$b_i = \Omega \frac{RT_{c,i}}{P_{c,i}}. \tag{27b}$$

$T_{r,i} = T/T_{c,i}$ is the reduced temperature. $T_{c,i}$ and $P_{c,i}$ are the critical temperature and pressure of component i. ω_i is the acentricity factor. We use values from the DIPPR database for the critical temperature, pressure, and for the acentricity factor. Ψ and Ω are parameters that are specific to the equation of state. Their values are shown in Table 1. The function, α, is

$$\alpha(T_{r,i}, \omega_i) = \left(1 + m(\omega_i)(1 - T_{r,i}^{1/2})\right)^2. \tag{28}$$

$m(\omega_i)$ is a polynomium that is specific for each equation of state. Its expression is shown in Table 1. This concludes the description of the cubic equations of state. Next, we describe how to solve the

Table 1: Parameters in the Soave-Redlich-Kwong (SRK) and the Peng-Robinson (PR) cubic equations of state.

	ϵ	σ	Ω	Ψ	$m(\omega)$
PR	$1+\sqrt{2}$	$1-\sqrt{2}$	0.07779	0.45724	$m(\omega) = 0.37464 + 1.54226\omega - 0.26992\omega^2$
SRK	1	0	0.08664	0.42748	$m(\omega) = 0.480 + 1.574\omega - 0.176\omega^2$

cubic equation of state of state for the compressibility factor.

3.2.1 Solution of cubic equations of state

We reformulate the cubic equation of state as a polynomium in the compressibility factor Z:

$$q(Z) = Z^3 + \sum_{m=0}^{2} d_m Z^m = 0. \tag{29}$$

The polynomial coefficients are

$$d_2 = d_2(A,B) = B(\epsilon + \sigma - 1) - 1, \tag{30a}$$
$$d_1 = d_1(A,B) = A - B(\epsilon + \sigma) + B^2(\epsilon\sigma - \epsilon - \sigma), \tag{30b}$$
$$d_0 = d_0(A,B) = -\left(AB + \left(B^2 + B^3\right)\epsilon\sigma\right), \tag{30c}$$

where A and B are functions of temperature, T, pressure, P, and composition, n:

$$A = A(T,P,n) = \frac{Pa_m}{R^2T^2}, \tag{31a}$$
$$B = B(T,P,n) = \frac{Pb_m}{RT}. \tag{31b}$$

The compressibility factors are therefore also functions of temperature, T, pressure, P, and composition, n, i.e. $Z = Z(T,P,n)$. The polynomium $q(Z) = q(Z;T,P,n)$ will have either one or three roots depending on the given T, P, and n. When it has three roots, the smallest root is the liquid phase compressibility and the largest root is the vapor phase compressibility. This is essentially what distinguishes the residual Gibbs energy of the vapor phase from the residual Gibbs energy of the liquid phase. For given T, P, and n, we use Newton's method to solve for the compressibility factor:

$$Z_{k+1} = Z_k - q(Z_k)/q'(Z_k), \tag{32a}$$
$$q'(Z_k) = 3Z_k^2 + 2d_2Z_k + d_1. \tag{32b}$$

We use the stopping criterium

$$|q(Z_{k+1})| < \epsilon, \tag{33}$$

for a given tolerance, ϵ. The initial guess for the Newton iterations depends on whether we are searching for a vapor root or a liquid root:

$$Z_0 = 1, \qquad \text{(Vapor)} \tag{34a}$$

$$Z_0 = B. \qquad \text{(Liquid)} \tag{34b}$$

It is important to note that in vapor-liquid equilibrium computations, the vapor and the liquid phases will have different compositions, n^v and n^l. It is therefore necessary to solve the polynomium $q(Z; T, P, n^v)$ for the vapor compressibility and $q(Z; T, P, n^l)$ for the liquid compressibility. The compressibilities are not roots of the same polynomium.

3.3 Residual Gibbs energy with cubic equations of state

We insert the cubic equation of state into the expression for the residual Gibbs energy. The evaluation of the integral is described in Appendix B. The final expression becomes

$$\begin{aligned} G^R(T,P,n) &= -\int_{V=\infty}^{V} \left(\frac{RT}{V/N - b_m} - \frac{a_m}{(V/N + \epsilon b_m)(V/N + \sigma b_m)} - \frac{NRT}{V} \right) dV \\ &\quad + NRT(Z-1) - NRT \ln Z \\ &= -NRT \ln\left(\frac{Z-B}{Z}\right) - \frac{N}{\epsilon - \sigma}\frac{a_m}{b_m} \ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right) + NRT(Z-1) - NRT \ln Z \\ &= NRT(Z-1) - NRT \ln(Z-B) - \frac{N}{\epsilon - \sigma}\frac{a_m}{b_m} \ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right). \end{aligned} \tag{35}$$

We reiterate that the above expression describes the residual Gibbs energy of both the vapor phase, $G^{R,v}(T, P, n^v)$, and of the liquid phase, $G^{R,l}(T, P, n^l)$. The cubic equation of state is a third order polynomium, $q(Z; T, P, n)$, in the compressibility factor. The vapor phase compressibility, $Z^v = Z^v(T, P, n^v)$, is the largest root of $q(Z^v; T, P, n^v)$ and the liquid phase compressibility factor, $Z^l = Z^l(T, P, n^l)$, is the smallest root of $q(Z^l; T, P, n^l)$.

3.4 Summary

The residual Gibbs energy of a nonideal mixture (either a vapor or a liquid mixture) based on a cubic equation of state is

$$G^R(T,P,n) = NRT(Z-1) - NRT \ln(Z-B) - \frac{N}{\epsilon - \sigma}\frac{a_m}{b_m} \ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right). \tag{36}$$

4 Fugacity and fugacity coefficients

In this section, we introduce the chemical potential in order to define the fugacity. Next, we introduce the fugacity coefficients which are the quantities that we will actually use in the vapor-liquid equilibrium computations. That is, the chemical potentials and the fugacities only serve the purpose of introducing expressions for the fugacity coefficients.

4.1 Definition of chemical potential

The chemical potential of component i in a mixture is by definition the partial derivative of the Gibbs energy of that mixture with respect to mole number i:

$$\mu_i(T,P,n) = \frac{\partial G}{\partial n_i}(T,P,n). \tag{37}$$

The same definition holds for any mixture, i.e. also for ideal gas and liquid mixtures:

$$\mu_i^{ig}(T,P,n) = \frac{\partial G^{ig}}{\partial n_i^v}(T,P,n^v), \tag{38a}$$

$$\mu_i^{id}(T,P,n) = \frac{\partial G^{id}}{\partial n_i^l}(T,P,n^l). \tag{38b}$$

4.2 Chemical potential of a pure component ideal gas

The definition of the fugacity of component i depends on the chemical potential of a pure component (pc) ideal gas consisting entirely of component i. The Gibbs energy of such a pure component ideal gas is

$$\begin{aligned} G_i^{pc,ig}(T,P,n_i^v) &= n_i^v g_i^{ig}(T,P) \\ &= n_i^v \left(\Gamma_i(T) + RT\ln P\right). \end{aligned} \tag{39}$$

The chemical potential of that pure component ideal gas is therefore

$$\begin{aligned} \mu_i^{pc,ig}(T,P,n_i^v) &= \frac{\partial G_i^{pc,ig}}{\partial n_i^v}(T,P,n_i^v) \\ &= g_i^{ig}(T,P) \\ &= \Gamma_i(T) + RT\ln P. \end{aligned} \tag{40}$$

We note that the pure component ideal gas chemical potential is independent of the mole number, i.e. $\mu_i^{pc,ig}(T,P,n_i^v) = \mu_i^{pc,ig}(T,P)$.

4.3 Definition of fugacity

We define the fugacity of a mixture (vapor or liquid) of composition n. First, we recall that the Gibbs energy of an ideal gas mixture of composition n^v (in moles) is

$$G^{ig}(T,P,n^v) = \sum_i n_i^v \left(\Gamma_i(T) + RT\ln P\right) + RT\sum_i n_i^v \ln y_i. \tag{41}$$

Next, we express the ideal gas mixture chemical potential of component i in terms of the chemical potential of a pure component ideal gas that consists of component i. Because we define fugacity in the same way for both vapor and liquid mixtures, we use n for the composition instead of n^v. Similarly, we use $z_i = n_i/N$, where $N = \sum_j n_j$, for the mole fraction of component i. The i'th

chemical potential of an ideal gas mixture is

$$\begin{aligned}\mu_i^{ig}(T,P,n) &= \frac{\partial G^{ig}}{\partial n_i^v}(T,P,n) \\ &= \Gamma_i(T) + RT\ln P + RT\ln z_i \\ &= \Gamma_i(T) + RT\ln P_0 + RT\ln\frac{z_i P}{P_0} \\ &= \mu_i^{pc,ig}(T,P_0) + RT\ln\frac{z_i P}{P_0}. \end{aligned} \tag{42}$$

The differentiation of the first sum in the expression (41) for the Gibbs energy of an ideal gas mixture is straightforward. We describe the differentiation of the second sum in Appendix A. The fugacity of component i, $f_i(T,P,n)$, is defined such that the above expression is valid for the chemical potential of mixtures that are not ideal gas mixtures. It is therefore implicitly defined by replacing $z_i P$ with $f_i(T,P,n)$ in the above:

$$\mu_i(T,P,n) = \mu_i^{pc,ig}(T,P_0) + RT\ln\frac{f_i(T,P,n)}{P_0}. \tag{43}$$

Note that it is still the chemical potential of a pure component ideal gas that appears on the right-hand side of the above expression. Next, we introduce the fugacity coefficients and use the above expression to relate the fugacity coefficients to the residual Gibbs energy.

4.4 Fugacity coefficients

In this section, we introduce the fugacity coefficients and relate them to the residual Gibbs energy. First, we isolate the chemical potential of the pure component ideal gas in (42):

$$\mu_i^{pc,ig}(T,P_0) = \mu_i^{ig}(T,P,n) - RT\ln\frac{z_i P}{P_0}. \tag{44}$$

Then we substitute the chemical potential of the pure component ideal gas into (43):

$$\begin{aligned}\mu_i(T,P,n) &= \mu_i^{pc,ig}(T,P_0) + RT\ln\frac{f_i(T,P,n)}{P_0} \\ &= \mu_i^{ig}(T,P,n) - RT\ln\frac{z_i P}{P_0} + RT\ln\frac{f_i(T,P,n)}{P_0} \\ &= \mu_i^{ig}(T,P,n) + RT\ln\frac{f_i(T,P,n)}{z_i P}. \end{aligned} \tag{45}$$

We define the fugacity coefficient of component i as

$$\phi_i(T,P,n) = \frac{f_i(T,P,n)}{z_i P}. \tag{46}$$

We substitute the fugacity coefficient into (45), isolate the term that contains the fugacity coefficient, and use the definition of the chemical potentials:

$$\begin{aligned} RT \ln \phi_i(T,P,n) &= \mu_i(T,P,n) - \mu_i^{ig}(T,P,n) \\ &= \frac{\partial G}{\partial n_i}(T,P,n) - \frac{\partial G^{ig}}{\partial n_i}(T,P,n) \\ &= \frac{\partial}{\partial n_i}\left(G(T,P,n) - G^{ig}(T,P,n)\right) \\ &= \frac{\partial G^R}{\partial n_i}(T,P,n) \end{aligned} \tag{47}$$

We thus obtain an expression for the logarithm of the fugacity coefficients:

$$\ln \phi_i(T,P,n) = \frac{1}{RT}\frac{\partial G^R}{\partial n_i}(T,P,n). \tag{48}$$

It is only the logarithms of the fugacity coefficients that we use in the vapor-liquid equilibrium computations. It is therefore not necessary to isolate the fugacity coefficients in the above. Next, we derive expressions for the logarithmic fugacity coefficients of ideal and nonideal gas and liquid mixtures.

4.4.1 Ideal gas mixture

The residual Gibbs energy of ideal gas mixtures is zero. The logarithmic fugacity coefficients are therefore also zero:

$$\begin{aligned} \ln \phi_i^{ig}(T,P,n^v) &= \frac{1}{RT}\frac{\partial G^{R,ig}}{\partial n_i}(T,P,n) \\ &= 0. \end{aligned} \tag{49}$$

4.4.2 Ideal liquid mixture

The residual Gibbs energy of ideal liquid mixtures is linear in the mole numbers. The differentiation is therefore straightforward. The logarithmic fugacity coefficients are

$$\begin{aligned} \ln \phi_i^{id}(T,P,n^l) &= \frac{1}{RT}\frac{\partial G^{R,id}}{\partial n_i^l} \\ &= \frac{1}{RT}\left(RT \ln \frac{P_i^{sat}(T)}{P} + v_i^l(T)(P - P_i^{sat}(T))\right) \\ &= \ln \frac{P_i^{sat}(T)}{P} + \frac{v_i^l(T)(P - P_i^{sat}(T))}{RT}. \end{aligned} \tag{50}$$

The fugacity coefficients of ideal liquids are therefore independent of the composition, i.e. $\phi_i^{id}(T,P,n^l) = \phi_i^{id}(T,P)$.

4.4.3 Nonideal mixture

We present the derivation of the fugacity coefficients for nonideal mixtures in Appendix C. The final expression is

$$\ln \phi_i(T,P,n) = (Z-1)\frac{b_i}{b_m} - \ln(Z-B) - \frac{1}{\epsilon-\sigma}\frac{1}{RTb_m}\left(2\sum_j z_j a_{ij} - a_m\frac{b_i}{b_m}\right)\ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right). \tag{51}$$

The above expression applies to both vapor and liquid mixtures. The logarithmic fugacity coefficients of vapor mixtures, $\ln\phi_i^v(T,P,n^v)$, use the vapor phase compressibility, $Z^v = Z^v(T,P,n^v)$, which is the largest root of the cubic equation of state, $q(Z^v;T,P,n^v)$. The logarithmic fugacity coefficients of liquid mixtures use the liquid phase compressibility, $Z^l(T,P,n^l)$, which is the smallest root of $q(Z^l;T,P,n^l)$. Furthermore, several of the quantities in the above expression depend on the composition. It is therefore different values of $a_m(T,n)$, $b_m(n)$, $B(T,P,n)$, and z_i that appear in the expressions for the vapor and liquid logarithmic fugacity coefficients.

4.5 Summary

The expressions for the logarithmic fugacity coefficients are

$$\ln \phi_i^{ig}(T,P,n^v) = 0, \tag{52a}$$

$$\ln \phi_i^{id}(T,P,n^l) = \ln\frac{P_i^{sat}(T)}{P} + \frac{v_i^l(T)(P-P_i^{sat}(T))}{RT}, \tag{52b}$$

$$\ln \phi_i(T,P,n) = (Z-1)\frac{b_i}{b_m} - \ln(Z-B) - \frac{1}{\epsilon-\sigma}\frac{1}{RTb_m}\left(2\sum_j z_j a_{ij} - a_m\frac{b_i}{b_m}\right)\ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right). \tag{52c}$$

The latter expression applies to both nonideal vapor and liquid mixtures.

5 Vapor-liquid equilibrium

In this section, we formulate the vapor-liquid equilibrium problem as an optimization problem. We then derive the vapor-liquid equilibrium conditions as the first-order optimality conditions of this optimization problem. Next, we reformulate the equilibrium conditions in terms of the fugacity coefficients that we introduced in Section 4. Finally, we use Newton's method to solve the equilibrium conditions.

5.1 Equilibrium conditions

We consider a mixture that exists in both a vapor phase (v) and a liquid phase (l). The mixture contains N_C components. The vapor phase has composition n^v, and the liquid phase has composition n^l. Both n^v and n^l are vectors of mole numbers. Both phases have the same temperature, T, and the same pressure, P. The mixture is isothermal and isobaric, i.e. the temperature and the pressure are

specified constants. The problem is to find the vapor-liquid composition (in moles) at equilibrium. The condition of equilibrium is that Gibbs energy of the mixture is minimal. We formulate this condition as the optimization problem

$$\min_{n^v, n^l} \quad G^v(T, P, n^v) + G^l(T, P, n^l), \tag{53a}$$

$$s.t. \qquad n_i^v + n_i^l = n_i, \quad i = 1, \ldots, N_C. \tag{53b}$$

The total moles of each component, n_i, is specified. The constraint ensures that mass is conserved, i.e. that mass is distributed among the vapor phase and the liquid phase. Because the constraint is linear, we use it to eliminate the liquid moles, i.e. $n^l = n - n^v$. By doing so, we obtain an unconstrained optimization problem:

$$\min_{n^v} G^v(T, P, n^v) + G^l(T, P, n - n^v). \tag{54}$$

The first-order optimality conditions require that the derivatives of the objective function with respect to the vapor mole numbers are zero:

$$\frac{\partial G^v}{\partial n_i^v}(T, P, n^v) - \frac{\partial G^l}{\partial n_i^l}(T, P, n - n^v) = 0. \tag{55}$$

We have used that $\frac{\partial}{\partial n_i^v}\left(G^l(T, P, n - n^v)\right) = -\frac{\partial G^l}{\partial n_i^l}(T, P, n - n^v)$. For brevity, we write n^l instead of $n - n^v$ in most of the following equations in this section, but it is implicitly understood that n^l has been eliminated. Next, we rewrite the equilibrium conditions in terms of the chemical potentials:

$$\begin{aligned}\frac{\partial G^v}{\partial n_i^v}(T, P, n^v) - \frac{\partial G^l}{\partial n_i^l}(T, P, n^l) &= \mu_i^v(T, P, n^v) - \mu_i^l(T, P, n^l) \\ &= \left(\mu_i^v(T, P, n^v) - \mu_i^{pc,ig}(T, P_0)\right) - \left(\mu_i^l(T, P, n^l) - \mu_i^{pc,ig}(T, P_0)\right).\end{aligned} \tag{56}$$

We can obtain an expression for the difference in each parentheses from the definition of the fugacities:

$$\begin{aligned}&\left(\mu_i^v(T, P, n^v) - \mu_i^{pc,ig}(T, P_0)\right) - \left(\mu_i^l(T, P, n^l) - \mu_i^{pc,ig}(T, P_0)\right) \\ &\qquad = RT \ln \frac{f_i^v(T, P, n^v)}{P_0} - RT \ln \frac{f_i^l(T, P, n^l)}{P_0}.\end{aligned} \tag{57}$$

From the definition of the fugacity coefficients, we know that $f_i^v(T, P, n^v) = \phi_i^v(T, P, n^v) y_i P$ and $f_i^l(T, P, n^l) = \phi_i^l(T, P, n^l) x_i P$. We use those expressions to introduce the fugacity coefficients:

$$\begin{aligned}RT \ln \frac{f_i^v(T, P, n^v)}{P_0} - RT \ln \frac{f_i^l(T, P, n^l)}{P_0} &= RT\left(\ln f_i^v(T, P, n^v) - \ln f_i^l(T, P, n^l)\right) \\ &= RT\left(\ln\left(\phi_i^v(T, P, n^v) y_i P\right) - \ln\left(\phi_i^l(T, P, n^l) x_i P\right)\right) \\ &= RT\left(\ln \phi_i^v(T, P, n^v) + \ln y_i - \ln \phi_i^l(T, P, n^l) - \ln x_i\right).\end{aligned} \tag{58}$$

We recall that we have eliminated the liquid mole numbers, i.e. $n^l = n - n^v$. We solve the vapor-liquid equilibrium problem by solving the following nonlinear equations for the vapor mole numbers, n^v:

$$g_i(T, P, n^v) = \ln \phi_i^v(T, P, n^v) + \ln y_i - \ln \phi_i^l(T, P, n - n^v) - \ln x_i = 0. \tag{59}$$

The liquid mole numbers are $x_i = n_i^l / \left(\sum_j n_j^l\right) = (n_i - n_i^v) / \left(\sum_j n_j - n_j^v\right)$.

5.2 Solution of the equilibrium conditions

We solve the vapor-liquid equilibrium conditions with Newton's method:

$$n^{v,k+1} = n^{v,k} - \left(\frac{\partial g}{\partial n^v}\right)^{-1} g(T, P, n^{v,k}). \tag{60}$$

5.3 Summary

The vapor-liquid equilibrium conditions for a vapor-liquid mixture at temperature T, pressure P, and total composition n are

$$g_i(T, P, n^v) = \ln \phi_i^v(T, P, n^v) + \ln y_i - \ln \phi_i^l(T, P, n - n^v) - \ln x_i = 0. \tag{61}$$

We solve the above equilibrium conditions for the vapor composition, n^v. The liquid mole numbers are $x_i = (n_i - n_i^v) / \left(\sum_j n_i - n_i^v\right)$. We solve the vapor-liquid equilibrium equations with Newton's method:

$$n^{v,k+1} = n^{v,k} - \left(\frac{\partial g}{\partial n^v}\right)^{-1} g(T, P, n^{v,k}). \tag{62}$$

After solution, the liquid mole numbers are computed by $n^l = n - n^v$.

6 Equilibrium constants

In this section, we introduce the equilibrium constants and provide an expression for them in terms of the logarithmic fugacity coefficients. The equilibrium constants are useful when solving vapor-liquid equilibrium problems for ideal gas and liquid mixtures. We also use them as independent variables in the phase envelope computations in Section 8.

6.1 Equilibrium constants

The i'th equilibrium constant (sometimes called the equilibrium ratio) is the ratio between the vapor mole fraction and the liquid mole fraction of component i:

$$K_i = \frac{y_i}{x_i}. \tag{63}$$

We can therefore express the equilibrium conditions,

$$\ln \phi_i^v(T, P, n^v) + \ln y_i - \ln \phi_i^l(T, P, n - n^v) - \ln x_i = 0, \tag{64}$$

in terms of the equilibrium constants:

$$\ln K_i + \ln \phi_i^v(T, P, n^v) - \ln \phi_i^l(T, P, n^l) = 0. \tag{65}$$

At equilibrium, we can therefore obtain the following expression for the logarithmic equilibrium constants:

$$\ln K_i(T, P, n^v, n^l) = \ln \phi_i^l(T, P, n^l) - \ln \phi_i^v(T, P, n^v). \tag{66}$$

6.1.1 Ideal vapor-liquid mixture

For an ideal vapor-liquid mixture, the logarithmic equilibrium constants are equal to the logarithmic fugacity coefficients of the ideal liquid phase:

$$\begin{aligned} \ln K_i^{id}(T, P, n^v, n^l) &= \ln \phi_i^{id}(T, P, n^l) - \ln \phi_i^{ig}(T, P, n^v) \\ &= \ln \phi_i^{id}(T, P, n^l) \\ &= \ln \frac{P_i^{sat}(T)}{P} + \frac{v_i^l(T)(P - P_i^{sat}(T))}{RT} \end{aligned} \tag{67}$$

The equilibrium constants are therefore independent of the composition vectors, i.e. $K_i^{id}(T, P, n^v, n^l) = K_i^{id}(T, P)$. That can be exploited in the ideal vapor-liquid equilibrium computations.

6.2 Summary

The equilibrium constants for ideal and nonideal vapor-liquid mixtures are

$$\ln K_i^{id}(T, P) = \ln \frac{P_i^{sat}(T)}{P} + \frac{v_i^l(T)(P - P_i^{sat}(T))}{RT}, \tag{68a}$$

$$\ln K_i(T, P, n^v, n^l) = \ln \phi_i^l(T, P, n^l) - \ln \phi_i^v(T, P, n^v). \tag{68b}$$

7 Vapor-liquid equilibrium of ideal mixtures

In this section, we exploit the fact that the equilibrium constants of ideal vapor-liquid mixtures, $K_i^{id}(T, P)$, are independent of composition. We can therefore solve the vapor-liquid equilibrium by 1) evaluating the equilibrium constants at the specified temperature, T, and pressure, P, 2) solve for the vapor fraction, and 3) compute the vapor-liquid composition (in moles) from the equilibrium constants and the vapor fraction. The ideal vapor-liquid equilibrium constants are

$$K_i^{id}(T, P) = \exp(\ln K_i^{id}(T, P)), \tag{69a}$$

$$\ln K_i^{id}(T, P) = \ln \frac{P_i^{sat}(T)}{P} + \frac{v_i^l(T)(P - P_i^{sat}(T))}{RT}. \tag{69b}$$

We express the vapor mole fractions, y_i, in terms of the equilibrium constants, $K_i^{id}(T,P)$, and the liquid mole fractions, x_i:

$$y_i = K_i^{id}(T,P)x_i. \tag{70}$$

The total moles in the vapor and liquid phases are $N^v = \sum_i n_i^v$ and $N^l = \sum_i n_i^l$. The total moles in the mixture is $N = N^v + N^l$. The vapor fraction is $\beta = N^v/N$ and $1-\beta = 1 - N^v/N = (N - N^v)/N = N^l/N$. The total mole fractions are $z_i = n_i/N$. We derive the expressions for x_i from the mass balance:

$$n_i^v + n_i^l = n_i, \tag{71a}$$

$$\frac{n_i^v}{N} + \frac{n_i^l}{N} = \frac{n_i}{N}, \tag{71b}$$

$$\frac{N^v}{N}\frac{n_i^v}{N^v} + \frac{N^l}{N}\frac{n_i^l}{N^l} = \frac{n_i}{N}, \tag{71c}$$

$$\beta y_i + (1-\beta)x_i = z_i, \tag{71d}$$

$$\beta K_i^{id}(T,P)x_i + (1-\beta)x_i = z_i, \tag{71e}$$

$$(1 - \beta + \beta K_i^{id}(T,P))x_i = z_i. \tag{71f}$$

The liquid mole fraction, x_i, is therefore

$$x_i = \frac{z_i}{1 + \beta(K_i^{id}(T,P) - 1)}. \tag{72}$$

Because we solve for β, y and x will not sum to one during the solution procedure. We therefore require that both y and x sum to one. When $0 < \beta \leq 1$, the condition that

$$\sum_i x_i = 1, \tag{73}$$

together with $\sum_i z_i = 1$ implies that

$$\sum_i y_i = 1. \tag{74}$$

We omit the derivation. It is therefore sufficient to only require that $\sum_i x_i = 1$ for $0 < \beta \leq 1$. $\beta = 0$ implies that $x_i = z_i$ such that $\sum_i x_i = 1$ is satisfied independent of the equilibrium ratios, $K_i^{id}(T,P)$. In that case, we therefore require that

$$\sum_i y_i = 1. \tag{75}$$

The following condition is equivalent to $\sum_i x_i = 1$ for $0 < \beta \leq 1$ and to $\sum_i y_i = 1$ for $\beta = 0$:

$$\sum_i (y_i - x_i) = 0. \tag{76}$$

That is, $\sum_i y_i = 1$ and $\sum_i x_i = 1$ clearly imply that $\sum_i (y_i - x_i) = 0$. Because of the way that we compute y_i and x_i, $\sum_i (y_i - x_i)$ also implies that $\sum_i y_i = 1$ and $\sum_i x_i = 1$. We insert the expressions

for y_i and x_i to obtain an equation in β:

$$\begin{aligned}\sum_i (y_i - x_i) &= \sum_i \left(K_i^{id}(T,P) - 1\right) x_i \\ &= \sum_i \frac{K_i^{id}(T,P) - 1}{1 + \beta(K_i^{id}(T,P) - 1)} z_i = 0,\end{aligned} \tag{77}$$

We therefore solve the ideal vapor-liquid equilibrium problem by solving the scalar nonlinear equation,

$$f(\beta) = \sum_i \frac{K_i^{id}(T,P) - 1}{1 + \beta(K_i^{id}(T,P) - 1)} z_i = 0, \tag{78a}$$

$$f'(\beta) = -\sum_i \left(\frac{K_i^{id}(T,P) - 1}{1 + \beta(K_i^{id}(T,P) - 1)}\right)^2 z_i. \tag{78b}$$

$f(\beta) = 0$ is called the Rachford-Rice equation. Once we have solved the Rachford-Rice equation for β, we compute the vapor-liquid composition (in moles) by

$$\begin{aligned}n_i^v &= \frac{n_i^v}{N^v}\frac{N^v}{N} N \\ &= y_i \beta N \\ &= \frac{\beta K_i^{id}(T,P)}{1 + \beta(K_i^{id}(T,P) - 1)} n_i.\end{aligned} \tag{79}$$

Next, we compute the liquid mole numbers by $n^l = n - n^v$.

7.1 Solution of the Rachford-Rice equation

We solve the Rachford-Rice equation with Newton's method:

$$\beta_{k+1} = \beta_k - \frac{f(\beta_k)}{f'(\beta_k)}. \tag{80}$$

7.2 Summary

We compute the ideal vapor-liquid equilibrium constants with

$$K_i^{id}(T,P) = \exp(\ln K_i^{id}(T,P)), \tag{81a}$$

$$\ln K_i^{id}(T,P) = \ln \frac{P_i^{sat}(T)}{P} + \frac{v_i^l(T)(P - P_i^{sat}(T))}{RT}. \tag{81b}$$

Next, we solve the Rachford-Rice equation,

$$f(\beta) = \sum_i \frac{K_i^{id}(T,P) - 1}{1 + \beta(K_i^{id}(T,P) - 1)} z_i = 0, \tag{82}$$

for β using Newton's method,

$$\beta_{k+1} = \beta_k - \frac{f(\beta_k)}{f'(\beta_k)}, \tag{83}$$

where the derivative is

$$f'(\beta) = -\sum_i \left(\frac{K_i^{id}(T,P) - 1}{1 + \beta(K_i^{id}(T,P) - 1)} \right)^2 z_i. \tag{84}$$

We compute the vapor-liquid composition (in moles) by

$$n_i^v = \frac{\beta K_i^{id}(T,P)}{1 + \beta(K_i^{id}(T,P) - 1)} n_i, \tag{85a}$$

$$n_i^l = n_i - n_i^v. \tag{85b}$$

8 Computation of phase envelopes

In this section, we describe the equations that we solve in order to construct the phase envelope, and we discuss how to solve them. The phase envelope consists of two curves; the bubble-point curve where $\beta = 0$ and the dew-point curve where $\beta = 1$. We will consider the more general case of computing isocurves where $\beta \in [0, 1]$ is constant. In the method that we present in this section, the logarithmic equilibrium constants, $\ln K_i$, the logarithmic temperature, $\ln T$, and the logarithmic pressure, $\ln P$, are the independent variables, i.e. the variables that we solve for. The vapor mole fractions, y_i, and the liquid mole fractions, x_i, are dependent variables that are functions of the vapor fraction, β, the equilibrium constants, K_i, and the total mole fractions, z_i. The vapor fraction, β, and the vector of total mole fractions, z, are parameters in the problem. We introduce $N_C + 2$ equations that define one point on the isocurve. N_C is the number of components in the mixture. We solve these equations repeatedly in a sequential manner in order to construct the isocurve. The first N_C equations are the vapor-liquid equilibrium conditions formulated in terms of the equilibrium constants, i.e. (65). We repeat the equilibrium conditions here:

$$\ln K_i + \ln \phi_i^v(T,P,y) - \ln \phi_i^l(T,P,x) = 0. \tag{86}$$

The mole numbers do not appear explicitly in the expression for the fugacity coefficients. They only appear implicitly through the mole fractions. We can therefore write $\ln \phi_i^v(T,P,n^v) = \ln \phi_i^v(T,P,y(n^v)) = \ln \phi_i^v(T,P,y)$ where $y_i = n_i^v / \left(\sum_j n_j^v \right)$ and similarly for the liquid fugacity coefficients. That is why y and x appear in place of n^v and n^l in the above equilibrium conditions. However, because y and x are dependent variables, they will not necessarily sum to one during the solution of the equations. We therefore treat them as mole numbers when we evaluate the logarithmic fugacity coefficients. That is, we evaluate the vapor phase logarithmic fugacity coefficients, $\ln \phi_i^v(T,P,y)$, as $\ln \phi_i^v(T,P,n^v)$ where $n^v = y$, and similarly for the liquid fugacity coefficients. Also, because y_i and x_i are functions of K_i, we cannot explicitly isolate K_i in the vapor-liquid equilibrium equations above. We compute y_i and x_i in the same way that we did for the ideal vapor-liquid equilibrium problem:

$$x_i = \frac{z_i}{1 + \beta(K_i - 1)}, \tag{87a}$$

$$y_i = K_i x_i. \tag{87b}$$

However, here we do not use the ideal vapor-liquid equilibrium constants, and the equilibrium constants are independent variables. The next equation is the Rachford-Rice equation (78a). However, we implement it in the form,

$$\sum_i (y_i - x_i) = 0, \tag{88}$$

where we do not substitute the expressions for the vapor-liquid mole fractions. That is because we need to compute y_i and x_i anyway, in order to evaluate the logarithmic fugacity coefficients. The Rachford-Rice equation ensures that $\sum_i y_i = 1$ and $\sum_i x_i = 1$ are both satisfied[2]. In order to define the last equation, we collect the independent variables in a vector,

$$X = [\ln K; \ln T; \ln P] \in \mathbb{R}^{N_C+2}, \tag{89}$$

where $\ln K = [\ln K_1; \cdots; \ln K_{N_C}]$. The last isocurve equation states that one of the independent variables should be specified, i.e. the s'th component of X should have the value S:

$$X_s - S = 0. \tag{90}$$

That means that we specify either temperature, pressure, or one of the equilibrium constants in order to compute a point on the isocurve. Michelsen and Mollerup (2007, Chap. 12) discuss different strategies for selecting the specified variable. The isocurve is constructed by solving the isocurve equations for a sufficient number of values of S. We write the isocurve equations compactly as

$$F(X; S) = 0, \tag{91}$$

where

$$F_i = \ln K_i + \ln \phi_i^v(T, P, y) - \ln \phi_i^l(T, P, x), \qquad i = 1, \ldots, N_C, \tag{92a}$$

$$F_{N_C+1} = \sum_i (y_i - x_i), \tag{92b}$$

$$F_{N_C+2} = X_s - S. \tag{92c}$$

8.1 Solution of the isocurve equations

As mentioned, we construct the isocurve by solving the isocurve equations for a number of specified values of the s'th variable. For the m'th value of the specified variable, S^m, we solve the isocurve equations, $F(X^m; S^m) = 0$, with Newton's method:

$$X^{m,k+1} = X^{m,k} - \left(\frac{\partial F}{\partial X}\right)^{-1} F(X^{m,k}; S^m). \tag{93}$$

8.2 Computation of initial guess

When we have solved the isocurve equations for one value of the specified variable, we want to compute the sensitivities of the solution in order to compute an initial guess for the subsequent

[2]It is not immediately obvious that this is true. It is because of the way that we compute y_i and x_i.

Newton iterations. We differentiate the equation $F(X;S) = F(X(S);S) = 0$ with respect to S:

$$\frac{\partial F}{\partial X}\frac{\partial X}{\partial S} + \frac{\partial F}{\partial S} = 0. \tag{94}$$

We isolate the sensitivities of X with respect to S:

$$\frac{\partial X}{\partial S} = -\left(\frac{\partial F}{\partial X}\right)^{-1}\frac{\partial F}{\partial S}. \tag{95}$$

Once we have solved $F(X^m;S^m) = 0$ for X^m, we can compute an initial guess, $X^{m+1,0}$, for the solution of the isocurve equations for the next value of the specified variable, $F(X^{m+1};S^{m+1}) = 0$:

$$X^{m+1,0} = X^m + \frac{\partial X^m}{\partial S^m}(S^{m+1} - S^m). \tag{96}$$

8.3 Summary

The isocurve equations (or phase envelope equations if $\beta = 0$ or $\beta = 1$) are

$$F(X;S) = 0, \tag{97}$$

where the independent variables are $X = [\ln K; \ln T; \ln P]$. The equations are

$$F_i = \ln K_i + \ln \phi_i^v(T,P,y) - \ln \phi_i^l(T,P,x), \qquad i = 1,\ldots,N_C, \tag{98a}$$

$$F_{N_C+1} = \sum_i (y_i - x_i), \tag{98b}$$

$$F_{N_C+2} = X_s - S, \tag{98c}$$

where the vapor-liquid mole fractions are dependent variables:

$$x_i = \frac{z_i}{1+\beta(K_i-1)}, \tag{99a}$$

$$y_i = K_i x_i. \tag{99b}$$

We solve the equations with Newton's method:

$$X^{m,k+1} = X^{m,k} - \left(\frac{\partial F}{\partial X}\right)^{-1} F(X^{m,k};S^m). \tag{100}$$

We compute initial guesses for the Newton iterations with

$$X^{m+1,0} = X^m + \frac{\partial X^m}{\partial S^m}(S^{m+1} - S^m), \tag{101}$$

where the sensitivities are

$$\frac{\partial X}{\partial S} = -\left(\frac{\partial F}{\partial X}\right)^{-1}\frac{\partial F}{\partial S}. \tag{102}$$

9 Derivatives

In this section, we provide the derivatives that are necessary to solve the vapor-liquid equilibrium problem (for both ideal and nonideal mixtures) and to compute the isocurves (i.e. curves where β is constant). In order to do so, we also provide the derivatives of the logarithmic fugacity coefficients.

9.1 Logarithmic fugacity coefficients

We provide the derivatives of the logarithmic fugacity coefficients for ideal liquids and nonideal mixtures. The logarithmic fugacity coefficients of ideal gases are zero, and the derivatives are therefore also zero.

9.1.1 Ideal liquid mixture

The logarithmic fugacity coefficients of an ideal liquid mixture are

$$\ln \phi_i^{id}(T,P) = \ln \frac{P_i^{sat}(T)}{P} + \frac{v_i^l(T)(P - P_i^{sat}(T))}{RT}. \tag{103}$$

The liquid volume is given by the DIPPR correlation,

$$v_i^l = \frac{B_i^{1+\left(1-\frac{T}{C_i}\right)^{D_i}}}{A_i}. \tag{104}$$

The saturation pressure is given by the DIPPR correlation,

$$P_i^{sat} = \exp\left(\ln P_i^{sat}\right), \tag{105a}$$

$$\ln P_i^{sat} = A_i + \frac{B_i}{T} + C_i \ln(T) + D_i T^{E_i}. \tag{105b}$$

Note that the parameters, A_i, B_i, C_i, and D_i in the DIPPR correlation for the saturation pressure are not the same as those in the DIPPR correlation for the liquid volume. The derivatives of the logarithmic fugacity coefficients are

$$\frac{\partial \ln \phi_i^{id}}{\partial T} = \frac{\partial \ln P_i^{sat}}{\partial T} + \frac{1}{RT}\left(\left(\frac{\partial v_i^l}{\partial T} - \frac{v_i^l(T)}{T}\right)(P - P_i^{sat}(T)) - v_i^l(T)\frac{\partial P_i^{sat}}{\partial T}\right), \tag{106a}$$

$$\frac{\partial \ln \phi_i^{id}}{\partial P} = -\frac{1}{P} + \frac{v_i^l(T)}{RT}. \tag{106b}$$

The derivative of the liquid volume is

$$\frac{\partial v_i^l}{\partial T} = -\ln B_i \frac{D_i}{C_i}\left(1 - \frac{T}{C_i}\right)^{D_i - 1} v_i^l. \tag{107}$$

The derivative of the saturation pressure is

$$\frac{\partial \ln P_i^{sat}}{\partial T} = \frac{1}{T}\left(C_i - \frac{B_i}{T} + D_i E_i T^{E_i}\right), \tag{108a}$$

$$\frac{\partial P_i^{sat}}{\partial T} = P_i^{sat}(T)\frac{\partial \ln P_i^{sat}}{\partial T}. \tag{108b}$$

9.1.2 Nonideal mixture

The logarithmic fugacity coefficients of nonideal mixtures have a complicated expression. We therefore introduce auxiliary functions such that we can write it as

$$\ln \phi_i = (Z-1)\frac{b_i}{b_m} - g_z - \frac{1}{\epsilon - \sigma} g_{\phi,i} f. \tag{109}$$

The auxiliary functions are

$$f = \ln\left(\frac{Z + \epsilon B}{Z + \sigma B}\right), \tag{110a}$$

$$g_z = \ln(Z - B), \tag{110b}$$

$$g_{\phi,i} = \frac{1}{RTb_m}\left(2\sum_{j=1}^{N_C} z_j a_{ij} - a_m \frac{b_i}{b_m}\right). \tag{110c}$$

The derivatives of the logarithmic fugacity coefficients are

$$\frac{\partial \ln \phi_i}{\partial T} = \frac{\partial Z}{\partial T}\frac{b_i}{b_m} - \frac{\partial g_z}{\partial T} - \frac{1}{\epsilon - \sigma}\left(\frac{\partial g_{\phi,i}}{\partial T} f + g_{\phi,i}\frac{\partial f}{\partial T}\right), \tag{111a}$$

$$\frac{\partial \ln \phi_i}{\partial P} = \frac{\partial Z}{\partial P}\frac{b_i}{b_m} - \frac{\partial g_z}{\partial P} - \frac{1}{\epsilon - \sigma} g_{\phi,i}\frac{\partial f}{\partial P}, \tag{111b}$$

$$\frac{\partial \ln \phi_i}{\partial n_k} = \frac{\partial Z}{\partial n_k}\frac{b_i}{b_m} - (Z-1)\frac{b_i}{b_m^2}\frac{\partial b_m}{\partial n_k} - \frac{\partial g_z}{\partial n_k} - \frac{1}{(\epsilon - \sigma)}\left(\frac{\partial g_{\phi,i}}{\partial n_k} f + g_{\phi,i}\frac{\partial f}{\partial n_k}\right). \tag{111c}$$

The derivatives of the auxiliary function f are

$$\frac{\partial f}{\partial T} = \frac{\partial f}{\partial Z}\frac{\partial Z}{\partial T} + \frac{\partial f}{\partial B}\frac{\partial B}{\partial T}, \tag{112a}$$

$$\frac{\partial f}{\partial P} = \frac{\partial f}{\partial Z}\frac{\partial Z}{\partial P} + \frac{\partial f}{\partial B}\frac{\partial B}{\partial P}, \tag{112b}$$

$$\frac{\partial f}{\partial n_k} = \frac{\partial f}{\partial Z}\frac{\partial Z}{\partial n_k} + \frac{\partial f}{\partial B}\frac{\partial B}{\partial n_k}, \tag{112c}$$

where

$$\frac{\partial f}{\partial Z} = \frac{1}{Z + \epsilon B} - \frac{1}{Z + \sigma B}, \tag{113a}$$

$$\frac{\partial f}{\partial B} = \frac{\epsilon}{Z + \epsilon B} - \frac{\sigma}{Z + \sigma B}. \tag{113b}$$

The derivatives of the auxiliary function g_z are

$$\frac{\partial g_z}{\partial T} = \frac{\partial g_z}{\partial Z}\frac{\partial Z}{\partial T} + \frac{\partial g_z}{\partial B}\frac{\partial B}{\partial T}, \tag{114a}$$

$$\frac{\partial g_z}{\partial P} = \frac{\partial g_z}{\partial Z}\frac{\partial Z}{\partial P} + \frac{\partial g_z}{\partial B}\frac{\partial B}{\partial P}, \tag{114b}$$

$$\frac{\partial g_z}{\partial n_k} = \frac{\partial g_z}{\partial Z}\frac{\partial Z}{\partial n_k} + \frac{\partial g_z}{\partial B}\frac{\partial B}{\partial n_k}, \tag{114c}$$

where

$$\frac{\partial g_z}{\partial Z} = \frac{1}{Z-B}, \tag{115a}$$

$$\frac{\partial g_z}{\partial B} = -\frac{1}{Z-B}. \tag{115b}$$

The derivatives of the auxiliary function $g_{\phi,i}$ are

$$\frac{\partial g_{\phi,i}}{\partial T} = \frac{1}{T}\left(\frac{1}{Rb_m}\left(2\sum_{j=1}^{N_C} z_j \frac{\partial a_{ij}}{\partial T} - \frac{\partial a_m}{\partial T}\frac{b_i}{b_m}\right) - g_{\phi,i}\right), \tag{116a}$$

$$\frac{\partial g_{\phi,i}}{\partial n_k} = \frac{1}{b_m}\left[\left(\frac{2}{N}\left(a_{ik} - \sum_{j=1}^{N_C} x_j a_{ij}\right) - \frac{b_i}{b_m}\left(\frac{\partial a_m}{\partial n_k} - \frac{1}{b_m}\frac{\partial b_m}{\partial n_k}\right)\right) - \frac{\partial b_m}{\partial n_k} g_{\phi,i} + \frac{1}{RT}\right]. \tag{116b}$$

The compressibility factor, Z, satisfies the cubic polynomium $q(Z;T,P,n) = 0$. We obtain the derivatives of the compressibility factor with the inverse function theorem:

$$\frac{\partial Z}{\partial T} = -\left(\frac{\partial q}{\partial Z}\right)^{-1}\frac{\partial q}{\partial T}, \tag{117a}$$

$$\frac{\partial Z}{\partial P} = -\left(\frac{\partial q}{\partial Z}\right)^{-1}\frac{\partial q}{\partial P}, \tag{117b}$$

$$\frac{\partial Z}{\partial n_k} = -\left(\frac{\partial q}{\partial Z}\right)^{-1}\frac{\partial q}{\partial n_k}. \tag{117c}$$

The derivatives of the polynomium are

$$\frac{\partial q}{\partial Z} = 3Z^2 + \sum_{m=1}^{2} m d_m Z^{m-1} \tag{118a}$$

$$\frac{\partial q}{\partial T} = \sum_{m=0}^{2} \frac{\partial d_m}{\partial T} Z^m, \tag{118b}$$

$$\frac{\partial q}{\partial P} = \sum_{m=0}^{2} \frac{\partial d_m}{\partial P} Z^m, \tag{118c}$$

$$\frac{\partial q}{\partial n_k} = \sum_{m=0}^{2} \frac{\partial d_m}{\partial n_k} Z^m. \tag{118d}$$

The derivatives of the polynomial coefficients are

$$\frac{\partial d_m}{\partial T} = \frac{\partial d_m}{\partial A}\frac{\partial A}{\partial T} + \frac{\partial d_m}{\partial B}\frac{\partial B}{\partial T}, \tag{119a}$$
$$\frac{\partial d_m}{\partial P} = \frac{\partial d_m}{\partial A}\frac{\partial A}{\partial P} + \frac{\partial d_m}{\partial B}\frac{\partial B}{\partial P}, \tag{119b}$$
$$\frac{\partial d_m}{\partial n_k} = \frac{\partial d_m}{\partial A}\frac{\partial A}{\partial n_k} + \frac{\partial d_m}{\partial B}\frac{\partial B}{\partial n_k}, \tag{119c}$$

where

$$\frac{\partial d_2}{\partial A} = 0, \tag{120a}$$
$$\frac{\partial d_2}{\partial B} = (\epsilon + \sigma - 1), \tag{120b}$$
$$\frac{\partial d_1}{\partial A} = 1, \tag{120c}$$
$$\frac{\partial d_1}{\partial B} = -(\epsilon + \sigma) + 2(\epsilon\sigma - \epsilon - \sigma)B, \tag{120d}$$
$$\frac{\partial d_0}{\partial A} = -B, \tag{120e}$$
$$\frac{\partial d_0}{\partial B} = -\left(A + \epsilon\sigma(2B + 3B^2)\right). \tag{120f}$$

The derivatives of A are

$$\frac{\partial A}{\partial T} = \frac{\partial a_m}{\partial T}\frac{P}{R^2T^2} - \frac{2}{T}A, \tag{121a}$$
$$\frac{\partial A}{\partial P} = \frac{a_m}{R^2T^2}, \tag{121b}$$
$$\frac{\partial A}{\partial n_k} = \frac{\partial a_m}{\partial n_k}\frac{P}{R^2T^2}. \tag{121c}$$

The derivatives of B are

$$\frac{\partial B}{\partial T} = -\frac{b_m P}{RT^2}, \tag{122a}$$
$$\frac{\partial B}{\partial P} = \frac{b_m}{RT}, \tag{122b}$$
$$\frac{\partial B}{\partial n_k} = \frac{\partial b_m}{\partial n_k}\frac{P}{RT}. \tag{122c}$$

The derivatives of the van der Waals' mixing parameter a_m are

$$\frac{\partial a_m}{\partial T} = \sum_{i=1}^{N_C}\sum_{j=1}^{N_C} z_i z_j \frac{\partial a_{ij}}{\partial T}, \tag{123a}$$
$$\frac{\partial a_m}{\partial n_k} = \frac{2}{N}\left(\sum_{i=1}^{N_C} z_i a_{ik} - a_m\right). \tag{123b}$$

The derivative of the van der Waals' mixing parameter b_m is

$$\frac{\partial b_m}{\partial n_k} = \frac{b_k - b_m}{N}. \tag{124}$$

The derivative of a_{ij} is

$$\frac{\partial a_{ij}}{\partial T} = \frac{1 - k_{ij}}{2\sqrt{\hat{a}_{ij}}} \frac{\partial \hat{a}_{ij}}{\partial T}, \tag{125}$$

where the derivative of $\hat{a}_{ij}$ are

$$\frac{\partial \hat{a}_{ij}}{\partial T} = \frac{\partial a_i}{\partial T} a_j + a_i \frac{\partial a_j}{\partial T}. \tag{126}$$

The derivative of the pure component parameter a_i is

$$\frac{\partial a_i}{\partial T} = \frac{\partial \alpha}{\partial T} \Psi \frac{R^2 T_{c,i}^2}{P_{c,i}}. \tag{127}$$

The derivative of the function α is

$$\frac{\partial \alpha}{\partial T} = -\alpha \frac{m(\omega_i)}{\sqrt{\alpha T T_{c,i}}}. \tag{128}$$

9.2 Vapor-liquid equilibrium equations

The vapor-liquid equilibrium conditions are

$$g_i(T, P, n^v) = \ln \phi_i^v(T, P, n^v) + \ln y_i - \ln \phi_i^l(T, P, n^l) + \ln x_i, \tag{129}$$

where the liquid mole numbers are functions of the vapor mole numbers, i.e. $n^l = n - n^v$. The derivatives of $g_i(T, P, n^v)$ are

$$\frac{\partial g_i}{\partial n_k^v} = \frac{\partial \ln \phi_i^v}{\partial n_k^v} + \frac{\partial \ln y_i}{\partial n_k^v} + \frac{\partial \ln \phi_i^l}{\partial n_k^l} + \frac{\partial \ln x_i}{\partial n_k^l}. \tag{130}$$

The derivatives of the logarithmic mole fractions are

$$\frac{\partial \ln y_i}{\partial n_k^v} = \frac{\delta_{ik}}{n_k^v} - \frac{1}{N^v}, \tag{131a}$$

$$\frac{\partial \ln x_i}{\partial n_k^l} = \frac{\delta_{ik}}{n_k^l} - \frac{1}{N^l}. \tag{131b}$$

δ_{ik} is one if $i = k$ and zero if $i \neq k$.

9.3 Ideal vapor-liquid equilibrium equations

For ideal vapor-liquid mixtures, we solve the Rachford-Rice equation:

$$f(\beta) = \sum_i \frac{K_i^{id}(T,P) - 1}{1 + \beta(K_i^{id}(T,P) - 1)} z_i = 0. \tag{132}$$

The derivative of the $f(\beta)$ with respect to the vapor fraction is

$$f'(\beta) = -\sum_i \left(\frac{K_i^{id}(T,P) - 1}{1 + \beta(K_i^{id}(T,P) - 1)} \right)^2 z_i. \tag{133}$$

9.4 Phase envelope equations

The independent variables in the phase envelope computations are

$$X = [\ln K; \ln T; \ln P]. \tag{134}$$

The isocurve equations (or phase envelope equations if $\beta = 0$ or $\beta = 1$) are

$$F_i = \ln K_i + \ln \phi_i^v(T,P,y) - \ln \phi_i^l(T,P,x), \qquad i = 1, \ldots, N_C, \tag{135a}$$

$$F_{N_C+1} = \sum_i (y_i - x_i), \tag{135b}$$

$$F_{N_C+2} = X_s - S. \tag{135c}$$

The vapor-liquid mole fractions are dependent variables given by

$$x_i = \frac{z_i}{1 - \beta + \beta K_i}, \tag{136a}$$

$$y_i = K_i x_i. \tag{136b}$$

The derivatives of the vapor-liquid mole fractions with respect to the equilibrium constants are

$$\frac{\partial x_i}{\partial K_k} = \begin{cases} \frac{-\beta x_i^2}{z_i} & i = k, \\ 0 & i \neq k, \end{cases} \tag{137a}$$

$$\frac{\partial y_i}{\partial K_k} = \begin{cases} x_i + K_i \frac{\partial x_i}{\partial K_k} & i = k, \\ 0 & i \neq k. \end{cases} \tag{137b}$$

The derivatives of the vapor-liquid equilibrium conditions are therefore

$$\frac{\partial F_i}{\partial \ln K_k} = \delta_{ik} + K_k \left(\frac{\partial \ln \phi_i^v}{\partial n_k^v}(T,P,y) \frac{\partial y_k}{\partial K_k} - \frac{\partial \ln \phi_i^l}{\partial n_k^l}(T,P,x) \frac{\partial x_k}{\partial K_k} \right), \qquad i = 1, \ldots, N_C, \tag{138a}$$

$$\frac{\partial F_i}{\partial \ln T} = T \left(\frac{\partial \ln \phi_i^v}{\partial T}(T,P,y) - \frac{\partial \ln \phi_i^l}{\partial T}(T,P,x) \right), \qquad i = 1, \ldots, N_C, \tag{138b}$$

$$\frac{\partial F_i}{\partial \ln P} = P \left(\frac{\partial \ln \phi_i^v}{\partial P}(T,P,y) - \frac{\partial \ln \phi_i^l}{\partial P}(T,P,x) \right), \qquad i = 1, \ldots, N_C. \tag{138c}$$

The derivative of the equation that corresponds to the Rachford-Rice equation are

$$\frac{\partial F_{N_C+1}}{\partial K_k} = K_k \left(\frac{\partial y_k}{\partial K_k} - \frac{\partial x_k}{\partial K_k} \right). \tag{139}$$

The derivatives of the last equation are

$$\frac{\partial F_{N_C+2}}{\partial X} = e_s, \tag{140}$$

where element s of e_s is 1 and all other elements are zero. The derivative of the last equation with respect to the value of the specified variable is

$$\frac{\partial F_{N_C+2}}{\partial S} = -1. \tag{141}$$

10 An algorithm for computing phase envelopes and an example program

In this section, we describe an algorithm for computing isocurves, i.e. curves where the vapor fraction, β, is constant. The algorithm is described by Michelsen and Mollerup (2007, Chap. 12) and also by Michelsen (1980). It features automatic selection of the specified variable and the step size, and it uses cubic interpolation to enhance initial guesses for the Newton iterations. The algorithm sequentially constructs the isocurve. It begins at a point where the equilibrium constants can be approximated and sequentially computes points on the isocurve until it reaches the critical point.

10.1 Wilson's approximation of equilibrium constants

We approximate the equilibrium constants at the first point of the isocurve with the approximation by Wilson (1969):

$$K_i(T,P) = \frac{P_{c,i}}{P} \exp\left(5.373(1+\omega_i)\left(1 - \frac{T_{c,i}}{T}\right)\right). \tag{142}$$

$T_{c,i}$, $P_{c,i}$, and ω_i are the critical temperature, critical pressure, and the acentric factor. We use values of those parameters from the DIPPR database. We could potentially also use the ideal vapor-liquid equilibrium constants that we described in Section 6.

10.2 Cubic interpolation

It is essential to have good initial guesses for the Newton iterations. The algorithm therefore uses cubic interpolation to improve the estimates we described in Section 8. When we have computed two points on the isocurve and their sensitivities, we can determine a cubic polynomial for each of the variables that intersects these points. The polynomial is

$$p_i(S) = \sum_j a_{ji} S^j. \tag{143}$$

The superscript j on S is a power, and a_{ji} are the polynomial coefficients. Once we have solved for the m'th point on the isocurve, we interpolate the solution curves. We require that the polynomial and its derivatives intersect X^m and X^{m-1} and their derivatives with respect to S^m and S^{m-1}:

$$p_i(S^m) = X_i^m, \tag{144a}$$
$$p_i'(S^m) = \frac{\partial X_i^m}{\partial S^m}, \tag{144b}$$
$$p_i(S^{m-1}) = X_i^{m-1}, \tag{144c}$$
$$p_i'(S^{m-1}) = \frac{\partial X_i^{m-1}}{\partial S^{m-1}}. \tag{144d}$$

We write the above equations as a linear system of equations

$$Ma = b. \tag{145}$$

The system matrix, M, is a 4×4 matrix that contains powers of S^m and S^{m-1}:

$$M = \begin{bmatrix} 1 & S^m & (S^m)^2 & (S^m)^3 \\ 0 & 1 & 2S^m & 3(S^m)^2 \\ 1 & S^{m-1} & \left(S^{m-1}\right)^2 & \left(S^{m-1}\right)^3 \\ 0 & 1 & 2S^{m-1} & 3\left(S^{m-1}\right)^2 \end{bmatrix} \tag{146}$$

The matrix a is $4 \times N_C + 2$ and simply contains the polynomial coefficients a_{ji} (row j, column i). $N_C + 2$ is the number of elements in the vector X. The right-hand side, b, is also a $4 \times N_C + 2$ matrix and each column contains the right-hand sides of (144):

$$b_{1i} = X_i^m, \tag{147a}$$
$$b_{2i} = \frac{\partial X_i^m}{\partial S^m}, \tag{147b}$$
$$b_{3i} = X_i^{m-1}, \tag{147c}$$
$$b_{4i} = \frac{\partial X_i^{m-1}}{\partial S^{m-1}}. \tag{147d}$$

Once we have solved the linear system (145) for the polynomial coefficients, we compute the initial estimate, $X^{m+1,0}$, for the subsequent Newton iterations:

$$X_i^{m+1,0} = p_i(S^{m+1}). \tag{148}$$

The matrix M can become ill-conditioned if the difference between S^m and S^{m-1} is small, i.e. if the step is small[3]. In that case, it can be useful to revert to the linear approximation:

$$X^{m+1,0} = X^m + \frac{\partial X^m}{\partial S^m}(S^{m+1} - S^m). \tag{149}$$

[3]It is known that the monomial basis can lead to ill-conditioned system matrices in interpolation. It might be possible to remedy this by choosing another basis, e.g. the Lagrange polynomials.

We use the above linear approximation when the condition number of M is larger than 10^6. We also use the linear approximation when we do not have two points on the isocurve, e.g. when $m = 1$.

10.3 Automatic selection of specified variable

In each iteration, we choose the specified variable in subsequent iterations to be the variable whose sensitivity with respect to the value of the specified variable has the largest absolute value, i.e. we choose

$$\bar{s} = \arg\min_s \frac{\partial X_s^m}{\partial S^m} \tag{150}$$

In order to avoid rapid switching between variables, we only switch variables (i.e. set $s = \bar{s}$) if $\partial X_{\bar{s}}^m / \partial S^m > 1.1\ \partial X_s^m / \partial S^m$. That is, we only switch the specified variable if the sensitivity is 1.1 times larger than the sensitivity of the current specified variable.

10.4 Automatic step size selection

We use a number of heuristics to choose the step size. First, we specify a target number of Newton iterations, usually 3 or 4. If the Newton iterations require more iterations to converge, we half the step size. If they require less, we double the step size. We also half the step size if the specified variable has changed. In order to ensure that we move in the right direction when we switch variables, we multiply the step size with the sign of the difference between the specified variable in the previous two iterations. That is, if the specified variable is X_s, then we use the sign of $X_s^m - X_s^{m-1}$ to decide whether S should increase or decrease.

10.5 Algorithm for computing isocurves

Algorithm 1 provides an overview of the algorithm that we implement in Section 10.6.

10.6 Example program

We now describe a program that implements Algorithm 1. The program consists of a script and a number of functions. We use Matlab routines from ThermoLib to compute the logarithmic fugacity coefficients. ThermoLib is an open-source thermodynamic library created by Ritschel et al. (2017, 2016) and Gaspar et al. (2017). It is available at `www.psetools.org`. It implements the expressions for the logarithmic fugacity coefficients that we described in Section 4 and 9.1.2. The names of the example script and the functions are

- CreatePhaseEnvelope.m
  Example script
- ComputeWilsonKFactors.m
  Evaluates approximate equilibrium constants
- ComputeIsocurve.m
  Constructs an isocurve

Algorithm 1: An algorithm for computing isocurves.

Input: T, P, β, z, s
Output: N, $\{X^m\}_{m=1}^{N}$

Set $m = 1$;
Set $\Delta S^m = \Delta S^{\max}$;
while *the critical point has not been reached* **do**
  Solve the isocurve equations $F(X^m; S^m) = 0$ for X^m;
  Check whether X^m is close to the critical point, i.e. if $|\ln K_i| < \epsilon$ for all i;
  Compute the sensitivities $\partial X^m / \partial S^m$;
  Set $\bar{s} = \arg\min_s \partial X_s^m / \partial S^m$;
  if $\partial X_{\bar{s}}^m / \partial S^m > 1.1 \partial X_s^m / \partial S^m$ **and** $s \neq \bar{s}$ **then**
    Set $\Delta S^m = 0.5 \Delta S^m$;
    Set $s = \bar{s}$;
  end
  if *the number of Newton iterations exceeds the target number of iterations* **then**
    Set $\Delta S^m = 0.5 \Delta S^m$;
  else if *the number of Newton iterations is below the target number of iterations* **then**
    Set $\Delta S^m = 2 \Delta S^m$;
  end
  Set $\Delta S^m = \min(\Delta S^m, \Delta S^{\max})$;
  Compute the new specified variable, $S^{m+1} = X_s^m + \operatorname{sgn}\left(X_s^m - X_s^{m-1}\right) \Delta S^m$;
  if $m \geq 2$ **and** s *has not changed in this iteration* **and** $\operatorname{cond}(M) < 10^6$ **then**
    Compute $X^{m+1,0}$ with cubic interpolation;
  else
    Compute $X^{m+1,0}$ with the linear approximation;
  end
  Increment m by 1;
end

- IsocurveEquations.m
  Evaluates the isorcurve equations (and derivatives) described in Section 8

- CreateFlashOpts.m
  Default options for the computations

Listing 1 shows the first part of the example script, CreatePhaseEnvelope.m. First, the script runs another script, LoadLibrary.m, from ThermoLib. LoadLibrary adds all ThermoLib routines to the path such that we can run them. Next, we select the Peng-Robinson equation of state, and we manually specify the binary interaction parameters, k_{ij}. We use LoadParams from ThermoLib to construct a vector of parameters that we need to pass on to other ThermoLib routines. The first argument to LoadParams is a vector of indices that specifies the components in the mixture. 1, 2, 3, and 7 refer to methane, ethane, propane, and n-heptane, respectively. 363 is carbon-dioxide (CO_2). We will use the variable NC (number of components) later in the script. Finally, we specify the composition which is 60% methane, 8% ethane, 5% propane, 25% n-heptane, and 2% CO_2.

Listing 1: CreatePhaseEnvelope.m – Initialization

```
%% Create phase diagram
clc; clear all; close all;

% Add ThermoLib
run('C:\Users\Tobia\Dropbox\3 PhD Nonlinear Model Predictive Control for Oil
    ↪ Reservoirs\Projects\ThermodynamicLibrary\ComputeThermoLib\matlab\
    ↪ LoadLibrary');

%% Load parameters
% Equation of state
EoS = 'PR'; % Peng-Robinson

% Binary interaction parameters
kij = [
    0         0         0         0         0.1200
    0         0         0         0         0.1500
    0         0         0         0         0.1500
    0         0         0         0         0.1500
    0.1200    0.1500    0.1500    0.1500    0     ];

% Load DIPPR parameters
params = LoadParams([1:3, 7, 363], EoS, kij);

% Number of components
NC = params(5);

% Composition
z = [60; 8; 5; 25; 2]./100;
```

Listing 2 shows the remaining part of CreatePhaseEnvelope.m. This remaining part computes the bubble-point curve ($\beta = 0$) and the dew-point curve ($\beta = 1$). First, we use CreateFlashOpts.m to create a struct called opts that contains default settings for the computations. We discuss CreateFlashOpts.m later. Next, we select that the pressure of the first point on both isocurves is

specified. Pressure is variable number $N_C + 2$. We set the pressure of the first point on both the bubble-point and the dew-point curves to be 0.005 MPa. We use ComputeIsocurve to compute both the bubble-point curve and the dew-point curve. We use an initial guess of 88 K for the temperature of the first point on the bubble-point curve and 250 K for the dew-point curve. In the remaining part, we extract the temperature and pressure of the isocurves and plot them. Figure 1 shows the plot of the phase envelope. The phase envelope separates the two-phase region (inside) from the single-phase regions (outside). We see that the two-phase region extends up to around 18 MPa and to around 450 K. We also note that at low pressures, the phase envelope extends to very low temperatures, i.e. around 90 K. That means that it is necessary to bring the mixture to below 90 K in order to completely liquefy it. That is because the mixture contains a large amount of light gases. In particular, it contains 60% methane. The bubble-point and the dew-point curves meet at the critical point. It is marked with a black dot.

Listing 2: CreatePhaseEnvelope.m – Computation of the phase envelope

```
%% Compute phase envelope
% Create options structure
opts = CreateFlashOpts();

% The first specified variable is pressure
Idx = NC+2;

% Initial pressure
P0 = 0.005; % MPa

% Compute the bubble-point curve
T0 = 88;
b = 0;
BubblePointCurve = ComputeIsocurve(T0, P0, b, z, Idx, params, opts);

% Compute the dew-point curve
T0 = 250;
b = 1;
DewPointCurve = ComputeIsocurve(T0, P0, b, z, Idx, params, opts);

% Temperature and pressure
T = [exp(BubblePointCurve.X(NC+1, :)), exp(DewPointCurve.X(NC+1, end:-1:1))];
P = [exp(BubblePointCurve.X(NC+2, :)), exp(DewPointCurve.X(NC+2, end:-1:1))];

% Critical temperature and pressure (approximate)
Tc = exp(BubblePointCurve.X(NC+1, end));
Pc = exp(BubblePointCurve.X(NC+2, end));

% Create figure
figure(1);
plot(T,  P,  '-k', 'linewidth',   2); hold on;
plot(Tc, Pc, '.k', 'markersize', 20); hold off;
set(gca, 'fontsize', 14);
xlabel('Temperature [K]');
ylabel('Pressure [MPa]');
```

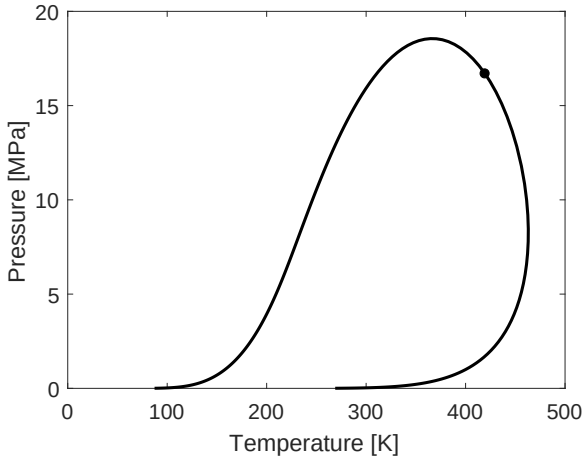

Figure 1: Phase envelope created with the example script CreatePhaseEnvelope.m.

Before we describe the ComputeIsocurve function, we show the ComputeWilsonKFactors function in Listing 3. It simply loads the relevant parameters and evaluates the equilibrium constants from (142). Note that the critical pressures are converted from Pa to MPa. That is required for the expression to be correct.

Listing 3: ComputeWilsonKFactors.m

```
function K = ComputeWilsonKFactors(T, P, params)

% Conversion factor
Pa2MPa = 1e-6; % MPa to Pa

% Extract relevant parameters
NoSinglePar  = params(1);
NC           = params(5);
omega    = params(NoSinglePar+1 + 24*NC + (1:NC) + 1);
Tc       = params(NoSinglePar+1 + 25*NC + (1:NC) + 1);
Pc       = params(NoSinglePar+1 + 26*NC + (1:NC) + 1)*Pa2MPa;

% Compute Wilson K factors
K = Pc./P.*exp(5.373*(1 + omega).*(1 - Tc./T));
```

Next, we describe the ComputeIsocurve function. Listing 4 shows the interface of ComputeIsocurve, the loading of parameters from the struct created with CreateFlashOpts, and the computation of initial guesses for the equilibrium constants. The function requires specification of temperature, T, pressure, P, the specified vapor fraction, b, the total composition, z, the index of the specified variable at the first point, Idx, the ThermoLib parameters, params, and finally, the options, opts.

Listing 4: ComputeIsocurve.m – Computation of an isocurve with β specified

```
function Env = ComputeIsocurve(T, P, b, z, Idx, params, opts)
%% Settings
% Target number of iterations
TargetIter = opts.TargetIter;

```

```
% Maximum number of Newton iterations
kmax = opts.MaxNewtonIter;

% Tolerance for the Newton iterations
newttol = opts.NewtonTol;

% Tolerance for the detection of the critical point
crittol = opts.CritPointTol;

% Maximum change in specification
maxdS = opts.MaxDeltaSpec;

% Whether to use polynomial regression or not
UseCubicInterpolation = opts.UseCubicInterpolation;

%% Compute initial guess
% Compute approximate equilibrium constants
K = ComputeWilsonKFactors(T, P, params);

% Initial guess for the first point
X = log([K; T; P]);
```

Listing 5 shows the first part of the computations that we carry out for each point on the isocurve. We solve the isocurve equations that we presented in Section 8 using Newton's method and we check whether we are closed to the critical point. The vapor phase and the liquid phase have the same composition at the critical point. That means that $y_i = x_i$ such that $K_i = y_i/x_i = 1$ and therefore $\ln K_i = 0$. That is how we detect whether we are close to the critical point. We present the function IsocurveEquations later. It essentially evaluates, $F(X;S)$, $\frac{\partial F}{\partial X}$, and $\frac{\partial F}{\partial S}$.

Listing 5: ComputeIsocurve.m – Solution of isocurve equations

```
%% Compute isocurve
% Initial specification
S(1) = X(Idx);

% Initial step
dS = maxdS;

i = 1;
CriticalPointReached = false;
while(~CriticalPointReached)
    % Evaluate function and Jacobian
    [F, dFdX, dFdS] = IsocurveEquations(X(:, i), b, z, S(i), Idx, params, opts);

    k = 0;
    Converged = (norm(F) < newttol);
    Diverged  = false;
    while(~Converged && ~Diverged)
        % Increment iteration counter
        k = k+1;

        % Newton update
```

```
49          X(:, i) = X(:, i) - dFdX\F;
50
51          % Evaluate function and Jacobian
52          [F, dFdX, dFdS] = IsocurveEquations(X(:, i), b, z, S(i), Idx, params,
              ↪ opts);
53
54          % Check for convergence
55          Converged = (norm(F) < newttol);
56          Diverged  = (k >= kmax);
57      end
58
59      % Check if the critical point has been reached
60      CriticalPointReached = (norm(X(1:end-2, i)) < crittol);
```

Listing 6 shows the remaining part of ComputeIsocurve.m. It computes the sensitivities, $\frac{\partial X}{\partial S}$, and uses them to select the specified variable. If the specified variable is changed, the step size is decreased. If the number of Newton iterations are above or below the target iterations, we double or half the step size, respectively. Next, we ensure that the step size is below the maximum step size, and we use X_s^m and X_s^{m-1} to determine whether S should increase or decrease. We use either cubic interpolation or a linear approximation to compute the initial guess for the subsequent Newton iterations.

Listing 6: ComputeIsocurve.m – Selection of specified variable and step size

```
62      % Compute sensitivities
63      dXdS(:, i) = -dFdX\dFdS;
64
65      % Update selection variable
66      [~, IdxNew] = max(abs(dXdS(:, i)));
67      SpecifiedVariableUpdated = false;
68      if(abs(dXdS(IdxNew, i)) > 1.1*abs(dXdS(Idx, i)))
69          SpecifiedVariableUpdated = true;
70          dS = 0.5*dS;
71          Idx = IdxNew;
72      end
73
74      % Update the step
75      if(k > TargetIter)
76          dS = dS*0.5;
77      elseif(k < TargetIter)
78          dS = dS*2;
79      end
80
81      % Apply upper bound to step size
82      dS = min(dS, maxdS);
83
84      % New specification
85      if(i > 1)
86          fac = sign(X(Idx, i) - X(Idx, i-1));
87      else
88          fac = 1;
89      end
90      S(i+1) = X(Idx, i) + fac*dS;
91
```

```
92      if(i > 1 && ~SpecifiedVariableUpdated && UseCubicInterpolation)
93          X(:, i+1) = PolynomialInitialEstimate ...
94              (S(i+1), X(:, i), S(i), dXdS(:, i), X(:, i-1), S(i-1), dXdS(:, i-1),
                ↪ opts);
95      else
96  %           % New initial estimate
97          X(:, i+1) = LinearInitialEstimate ...
98              (X(:, i), dS, dXdS(:, i));
99      end
100
101     % Display message
102     if(opts.Verbose > 0)
103         fprintf('(%3d, %d, %2d), (', i, Idx, k);
104         fprintf('%7.2f, ', exp(X(:, i)));
105         fprintf(')\n');
106     end
107
108     % Store data
109     Env.X(:, i) = X(:, i);
110     Env.k(i)    = k;
111
112     % Increment counter
113     i = i+1;
114 end
115 end
116
117 function Xip1 = PolynomialInitialEstimate(Sip1, Xi, Si, dXidS, Xim1, Sim1,
              ↪ dXim1dS, opts)
118 % Maximum condition number of system matrix in polynomial regression
119 maxcond = opts.MaxCondInterpol;
120
121 % System matrix
122 M = [
123     1, Si,      Si  ^2,    Si^3;
124     0, 1,     2*Si,      3*Si^2;
125     1, Sim1,    Sim1^2,    Sim1^3;
126     0, 1,     2*Sim1,    3*Sim1^2];
127
128 if(cond(M) < maxcond)
129     % Right-hand side
130     rhs = [Xi'; dXidS'; Xim1'; dXim1dS'];
131
132     % Polynomial coefficients
133     a = M\rhs;
134
135     % Extrapolate new initial estimate
136     Xip1 = [1, Sip1, Sip1^2, Sip1^3]*a;
137 else
138     % Step
139     dSip1 = Sip1 - Si;
140
141     % New initial estimate
142     Xip1 = LinearInitialEstimate(Xi, dSip1, dXidS);
```

```
143 end
144 end
145
146 function Xip1 = LinearInitialEstimate(Xi, dSip1, dXidS)
147 % New initial estimate
148 Xip1 = Xi + dXidS*dSip1;
149 end
```

Listing 7 shows the IsocurveEquations function. It essentially evaluates $F(X;S)$, $\frac{\partial F}{\partial X}$, and $\frac{\partial F}{\partial S}$. We mostly comment on the usage of the ThermoLib routines. In the phase envelope computations, we consider pressure in MPa. However, ThermoLib requires pressure to be in Pa. The ThermoLib routine MixFug computes the logarithmic fugacity coefficients as functions of temperature, pressure, and composition (in moles). It also computes both first and second order derivatives depending on the number of requested outputs. In the phase envelope computations, we only require the first order derivatives so we only request two outputs. Consequently, the second order derivatives are not computed. As we discussed in Section 4, we need to solve the cubic equation of state in order to evaluate the logarithmic fugacity coefficients. ThermoLib solves the cubic equation of state with Newton's method. We therefore specify both the tolerance (tol) and the maximum number of Newton iterations (maxit) when we call MixFug. The remainder of IsocurveEquations is a straightforward implementation of the isocurve equations and their derivatives. It only computes the Jacobian and the derivatives of F with respect to S if those outputs are requested.

Listing 7: IsocurveEquations.m – Evaluation of the isocurve equations

```
1  function [F, dFdX, dFdS] = IsocurveEquations(X, b, z, S, Idx, params, opts)
2  % Maximum Newton iterations for the equation of state
3  maxit = opts.MaxEosIter;
4
5  % Tolerance for Newton iterations for the equation of state
6  tol = opts.EosTol;
7
8  % Number of components
9  NC = params(5);
10
11 % Extract variables
12 K = exp(X(1:NC));
13 T = exp(X(NC+1));
14 P = exp(X(NC+2));
15
16 % Mole fractions
17 x = z./(1 - b + b*K);
18 y = K.*x;
19
20 % Conversion factor
21 MPa2Pa = 1e6;
22
23 % Compute fugacities with ThermoLib
24 [lnphiv, dlnphiv] = MixFug(T, P*MPa2Pa, y, 0, params, tol, maxit);
25 [lnphil, dlnphil] = MixFug(T, P*MPa2Pa, x, 1, params, tol, maxit);
26
27 % Temperature derivatives
```

```
dlnphivT = dlnphiv(:, 1);
dlnphilT = dlnphil(:, 1);

% Scale pressure derivatives
dlnphivP = dlnphiv(:, 2)*MPa2Pa;
dlnphilP = dlnphil(:, 2)*MPa2Pa;

% Scale composition derivatives
dlnphivnv = dlnphiv(:, 3:end);
dlnphilnl = dlnphil(:, 3:end);

%% Evaluate isocurve equations
F = zeros(NC+2, 1);
F(1:NC) = log(K) + lnphiv - lnphil;
F(NC+1) = sum(y - x);
F(NC+2) = X(Idx) - S;

if(nargout > 1)
    %% Evaluate Jacobian
    % Derivatives of K values wrt. ln K values
    dlnKdlnK = eye(NC,NC);

    % Derivatives of mole fractions wrt. ln K values
    dxdK = -b*x.^2./z;
    dydK = x + K.*dxdK;

    % Derivatives of fugacities wrt. ln K values
    dlnphivdlnK = repmat(K', NC, 1).*dlnphivnv*diag(dydK);
    dlnphildlnK = repmat(K', NC, 1).*dlnphilnl*diag(dxdK);

    % Jacobian wrt. ln K values
    dFdX = zeros(NC+2, NC+2);
    dFdX(1:NC, 1:NC) = dlnKdlnK + dlnphivdlnK - dlnphildlnK;

    % Jacobian wrt. ln T
    dFdX(1:NC, NC+1) = T*(dlnphivT - dlnphilT);

    % Jacobian wrt. ln P
    dFdX(1:NC, NC+2) = P*(dlnphivP - dlnphilP);

    % Jacobian wrt. ln K
    dFdX(NC+1, 1:NC) = K.*(dydK - dxdK);

    % Jacobian wrt. specified variable (ln K, ln T or ln P)
    dFdX(NC+2, Idx) = 1;

    if(nargout > 2)
        %% Derivatives wrt. S
        dFdS = zeros(NC+2, 1);
        dFdS(NC+2) = -1;
    end
end
```

For completeness, we show CreateFlashOpts in Listing 8. In our experience, the computations can be fairly sensitive to some of the parameters. In particular, large values of MaxDeltaSpec can cause failure to convergence in the Newton iterations. Furthermore, small values of CritPointTol can cause the algorithm to come too close to or overstep the critical point, again leading to failure to converge. Verbose can be set to 1 in order to print out information about each iteration. Large values of EosTol can also cause problems.

Listing 8: CreateFlashOpts.m – Default settings for the computations

```
function opts = CreateFlashOpts()

% Settings for Newton iterations
opts.MaxNewtonIter = 2e1;
opts.NewtonTol = 1e-6;

% Settings for the solution of the equation of state
opts.MaxEosIter = 2e1;
opts.EosTol = 1e-10;

% Target number of iterations (for selecting step size)
opts.TargetIter = 4;

% Maximum change in specification
opts.MaxDeltaSpec = 1e-1;

% Tolerance for the detection of the critical point
opts.CritPointTol = 1e-1;

% Whether to use polynomial regression to generate initial estimates or not
opts.UseCubicInterpolation = true;

% Maximum condition number of system matrix in polynomial regression
opts.MaxCondInterpol = 1e6;

% Set verbosity
opts.Verbose = 0;
```

10.7 Discussion

In this section, we have described a program that computes isocurves, i.e. curves where the vapor fraction, β, is constant. In order to create the phase envelope, we therefore compute the bubble-point curve ($\beta = 0$) and the dew-point curve ($\beta = 1$) separately. Often, we both want to compute the isocurve of β and $1 - \beta$. An example is the phase envelope. Michelsen and Mollerup (2007) suggest to combine the computations, e.g. such that we 1) compute the bubble-point curve, 2) "step over" the critical point, and 3) continue by computing the dew-point curve until a sufficiently low pressure is reached.

A Derivative of mixing term

The expression for the ideal gas mixture entropy (4b) involves a mixing term that contains $\sum_i n_i^v \ln y_i$. We use the derivative of the mixing term when we derive an expression for the chemical potential of an ideal gas mixture in (42). The derivative of the mixing term can appear to be incorrect at first sight. Because the derivative of this mixing term is important to the definition of fugacity, we present the differentiation of $\sum_i n_i^v \ln y_i$ here such that it is clear that it is correct. The derivative is

$$\begin{aligned}\frac{\partial}{\partial n_k^v}\left(\sum_i n_i^v \ln y_i\right) &= \sum_i \left(\frac{\partial n_i^v}{\partial n_k^v}\ln y_i + \frac{n_i^v}{y_i}\frac{\partial y_i}{\partial n_k^v}\right) \\ &= \sum_i \left(\delta_{ik}\ln y_i + \left(\sum_j n_j^v\right)\frac{\partial y_i}{\partial n_k^v}\right) \\ &= \ln y_k + \sum_i \left(\left(\sum_j n_j^v\right)\frac{\partial y_i}{\partial n_k^v}\right). \end{aligned} \tag{151}$$

The derivative of the mole fraction, y_i, is

$$\begin{aligned}\frac{\partial y_i}{\partial n_k^v} &= \frac{1}{\sum_j n_j^v}\frac{\partial n_i^v}{\partial n_k^v} - \frac{n_i^v}{\left(\sum_j n_j^v\right)^2} \\ &= \frac{1}{\sum_j n_j^v}\left(\delta_{ik} - y_i\right). \end{aligned} \tag{152}$$

We insert the above into the double sum in (151):

$$\begin{aligned}\sum_i \left(\left(\sum_j n_j^v\right)\frac{\partial y_i}{\partial n_k^v}\right) &= \sum_i \left(\delta_{ik} - y_i\right) \\ &= 1 - \sum_i y_i \\ &= 0. \end{aligned} \tag{153}$$

The derivative is therefore

$$\frac{\partial}{\partial n_k^v}\left(\sum_i n_i^v \ln y_i\right) = \ln y_k, \tag{154}$$

where it appears as if one has considered $\ln y_i$ to be independent of all the mole numbers, which is not the case.

B Derivation of the residual Gibbs energy for cubic equations of state

In this section, we evaluate the integral that appears in the expression for the residual Gibbs energy in (35). First, we split up the integral:

$$\begin{aligned}
&-\int_{V=\infty}^{V}\left(\frac{RT}{V/N-b_m}-\frac{a_m}{(V/N+\epsilon b_m)(V/N+\sigma b_m)}-\frac{NRT}{V}\right)dV \\
&\quad=-RT\int_{V=\infty}^{V}\left(\frac{1}{V/N-b_m}-\frac{NRT}{V}\right)dV+a_m\int_{V=\infty}^{V}\frac{1}{(V/N+\epsilon b_m)(V/N+\sigma b_m)}\,dV. \qquad (155)
\end{aligned}$$

We compute the antiderivative (or primitive function) of the integrand in the left term:

$$\int\left(\frac{1}{V/N-b_m}-\frac{NRT}{V}\right)dV=N\ln\left(\frac{1}{N}-\frac{b_m}{V}\right). \qquad (156)$$

Next, we evaluate the integral in the left term using the antiderivative:

$$\begin{aligned}
\int_{V=\infty}^{V}\left(\frac{1}{V/N-b_m}-\frac{NRT}{V}\right)dV&=N\left(\ln\left(\frac{1}{N}-\frac{b_m}{V}\right)-\ln\frac{1}{N}\right)\\
&=N\ln\left(1-\frac{Nb_m}{V}\right). \qquad (157)
\end{aligned}$$

The antiderivative of the integrand in the right term is

$$\int\frac{1}{(V/N+\epsilon b_m)(V/N+\sigma b_m)}\,dV=-\frac{N}{\epsilon-\sigma}\frac{1}{b_m}\ln\left(\frac{V+\epsilon Nb_m}{V+\sigma Nb_m}\right). \qquad (158)$$

The integral in the right term is therefore

$$\begin{aligned}
\int_{V=\infty}^{V}\frac{1}{(V/N+\epsilon b_m)(V/N+\sigma b_m)}\,dV&=-\frac{N}{\epsilon-\sigma}\frac{1}{b_m}\left(\ln\left(\frac{V+\epsilon Nb_m}{V+\sigma Nb_m}\right)-\ln 1\right)\\
&=-\frac{N}{\epsilon-\sigma}\frac{1}{b_m}\ln\left(\frac{V+\epsilon Nb_m}{V+\sigma Nb_m}\right). \qquad (159)
\end{aligned}$$

We now rewrite the two expressions that contain volume such that they contain the compressibility factor instead. We use that $PV=NRTZ$ such that $V=NRTZ/P$. We also recall that $B=Pb_m/(RT)$. The one expression is

$$\begin{aligned}
1-\frac{Nb_m}{V}&=1-\frac{Nb_m}{NRTZ/P}\\
&=1-\frac{Pb_m}{RTZ}\\
&=1-\frac{B}{Z}\\
&=\frac{Z-B}{Z}. \qquad (160)
\end{aligned}$$

The other expression is

$$\begin{aligned}\frac{V+\epsilon N b_m}{V+\sigma N b_m} &= \frac{NRTZ/P+\epsilon N b_m}{NRTZ/P+\sigma N b_m}\\ &= \frac{NRT/P}{NRT/P}\frac{Z+\epsilon P b_m/(RT)}{Z+\sigma P b_m/(RT)}\\ &= \frac{Z+\epsilon B}{Z+\sigma B}.\end{aligned} \tag{161}$$

The original integral is therefore

$$\begin{aligned}&-\int_{V=\infty}^{V}\left(\frac{RT}{V/N-b_m}-\frac{a_m}{(V/N+\epsilon b_m)(V/N+\sigma b_m)}-\frac{NRT}{V}\right)dV\\ &\quad=-NRT\ln\left(\frac{Z-B}{Z}\right)-\frac{N}{\epsilon-\sigma}\frac{a_m}{b_m}\ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right).\end{aligned} \tag{162}$$

We insert the expression for the integral into the expression for the residual Gibbs energy and simplify:

$$\begin{aligned}G^R(T,P,n) &= -\int_{V=\infty}^{V}\left(\frac{RT}{V/N-b_m}-\frac{a_m}{(V/N+\epsilon b_m)(V/N+\sigma b_m)}-\frac{NRT}{V}\right)dV\\ &\quad+NRT(Z-1)-NRT\ln Z\\ &= -NRT\ln\left(\frac{Z-B}{Z}\right)-\frac{N}{\epsilon-\sigma}\frac{a_m}{b_m}\ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right)+NRT(Z-1)-NRT\ln Z\\ &= NRT(Z-1)-NRT\ln(Z-B)-\frac{N}{\epsilon-\sigma}\frac{a_m}{b_m}\ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right).\end{aligned} \tag{163}$$

C Derivation of the logarithmic fugacity coefficients for cubic equations of state

The expression for the i'th logarithmic fugacity coefficient depends on the partial derivative of the residual Gibbs energy with respect to mole number i:

$$\ln\phi_i(T,P,n)=\frac{1}{RT}\frac{\partial G^R}{\partial n_i}. \tag{164}$$

The residual Gibbs energy (from a cubic equation of state) is

$$G^R(T,P,n)=NRT(Z-1)-NRT\ln(Z-B)-\frac{N}{\epsilon-\sigma}\frac{a_m}{b_m}\ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right). \tag{165}$$

We consider the molar residual Gibbs energy:

$$\begin{aligned}g^R(T,P,n) &= G^R(T,P,n)/N\\ &= RT(Z-1)-RT\ln(Z-B)-\frac{1}{\epsilon-\sigma}\frac{a_m}{b_m}\ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right).\end{aligned} \tag{166}$$

Conversely, the residual Gibbs energy is $G^R(T,P,n) = Ng^R(T,P,n)$, and the expression for the logarithmic fugacity coefficients is therefore

$$\begin{aligned}\ln\phi_i(T,P,n) &= \frac{1}{RT}\frac{\partial G^R}{\partial n_i}\\ &= \frac{1}{RT}\frac{\partial}{\partial n_i}\left(Ng^R(T,P,n)\right)\\ &= \frac{1}{RT}\left(g^R(T,P,n) + N\frac{\partial g^R}{\partial n_i}\right).\end{aligned} \tag{167}$$

The remainder of this section will be concerned with finding an expression for $\partial g^R/\partial n_i$ and use it to derive the expression for the logarithmic fugacity coefficients. First, we use straightforward differentiation:

$$\begin{aligned}\frac{\partial g^R}{\partial n_i} &= RT\frac{\partial Z}{\partial n_i} - RT\frac{1}{Z-B}\left(\frac{\partial Z}{\partial n_i} - \frac{\partial B}{\partial n_i}\right)\\ &\quad - \frac{1}{\epsilon-\sigma}\left[\left(\frac{\partial a_m}{\partial n_i}\frac{1}{b_m} - \frac{a_m}{b_m^2}\frac{\partial b_m}{\partial n_i}\right)\ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right) + \frac{a_m}{b_m}\frac{\partial}{\partial n_i}\left(\ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right)\right)\right).\end{aligned} \tag{168}$$

The derivative of the logarithmic coefficient is

$$\begin{aligned}\frac{\partial}{\partial n_i}\left(\ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right)\right) &= \left(\frac{Z+\epsilon B}{Z+\sigma B}\right)^{-1}\left[\frac{1}{Z+\sigma B}\left(\frac{\partial Z}{\partial n_i} + \epsilon\frac{\partial B}{\partial n_i}\right) - \frac{Z+\epsilon B}{(Z+\sigma B)^2}\left(\frac{\partial Z}{\partial n_i} + \sigma\frac{\partial B}{\partial n_i}\right)\right]\\ &= \frac{1}{Z+\epsilon B}\left(\frac{\partial Z}{\partial n_i} + \epsilon\frac{\partial B}{\partial n_i}\right) - \frac{1}{Z+\sigma B}\left(\frac{\partial Z}{\partial n_i} + \sigma\frac{\partial B}{\partial n_i}\right)\\ &= \frac{Z+\sigma B}{(Z+\epsilon B)(Z+\sigma B)}\left(\frac{\partial Z}{\partial n_i} + \epsilon\frac{\partial B}{\partial n_i}\right) - \frac{Z+\epsilon B}{(Z+\epsilon B)(Z+\sigma B)}\left(\frac{\partial Z}{\partial n_i} + \sigma\frac{\partial B}{\partial n_i}\right)\\ &= \frac{(\sigma-\epsilon)B}{(Z+\epsilon B)(Z+\sigma B)}\frac{\partial Z}{\partial n_i} + \frac{(\epsilon-\sigma)Z}{(Z+\epsilon B)(Z+\sigma B)}\frac{\partial B}{\partial n_i}\\ &= \frac{\epsilon-\sigma}{(Z+\epsilon B)(Z+\sigma B)}\left(Z\frac{\partial B}{\partial n_i} - B\frac{\partial Z}{\partial n_i}\right).\end{aligned} \tag{169}$$

We insert into the expression for $\partial g^R/\partial n_i$:

$$\begin{aligned}\frac{\partial g^R}{\partial n_i} &= RT\frac{\partial Z}{\partial n_i} - RT\frac{1}{Z-B}\left(\frac{\partial Z}{\partial n_i} - \frac{\partial B}{\partial n_i}\right)\\ &\quad - \frac{1}{\epsilon-\sigma}\left[\left(\frac{\partial a_m}{\partial n_i}\frac{1}{b_m} - \frac{a_m}{b_m^2}\frac{\partial b_m}{\partial n_i}\right)\ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right) + \frac{a_m}{b_m}\frac{\epsilon-\sigma}{(Z+\epsilon B)(Z+\sigma B)}\left(Z\frac{\partial B}{\partial n_i} - B\frac{\partial Z}{\partial n_i}\right)\right)\\ &= RT\frac{\partial Z}{\partial n_i} - RT\frac{1}{Z-B}\left(\frac{\partial Z}{\partial n_i} - \frac{\partial B}{\partial n_i}\right) - \frac{a_m}{b_m}\frac{1}{(Z+\epsilon B)(Z+\sigma B)}\left(Z\frac{\partial B}{\partial n_i} - B\frac{\partial Z}{\partial n_i}\right)\\ &\quad - \frac{1}{\epsilon-\sigma}\left(\frac{\partial a_m}{\partial n_i}\frac{1}{b_m} - \frac{a_m}{b_m^2}\frac{\partial b_m}{\partial n_i}\right)\ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right).\end{aligned} \tag{170}$$

We recall that $A = Pa_m/(R^2T^2)$ and $B = Pb_m/(RT)$ such that $A/B = a_m/(RTb_m)$. We use that relation to rewrite the three first terms:

$$\begin{aligned}
&RT\frac{\partial Z}{\partial n_i} - RT\frac{1}{Z-B}\left(\frac{\partial Z}{\partial n_i} - \frac{\partial B}{\partial n_i}\right) - \frac{a_m}{b_m}\frac{1}{(Z+\epsilon B)(Z+\sigma B)}\left(Z\frac{\partial B}{\partial n_i} - B\frac{\partial Z}{\partial n_i}\right) \\
&= \left(RT - \frac{RT}{Z-B} + \frac{a_m}{b_m}\frac{B}{(Z+\epsilon B)(Z+\sigma B)}\right)\frac{\partial Z}{\partial n_i} + \left(\frac{RT}{Z-B} - \frac{a_m}{b_m}\frac{Z}{(Z+\epsilon B)(Z+\sigma B)}\right)\frac{\partial B}{\partial n_i} \\
&= RT\left(1 - \frac{1}{Z-B} + \frac{A}{B}\frac{B}{(Z+\epsilon B)(Z+\sigma B)}\right)\frac{\partial Z}{\partial n_i} + RT\left(\frac{1}{Z-B} - \frac{A}{B}\frac{Z}{(Z+\epsilon B)(Z+\sigma B)}\right)\frac{\partial B}{\partial n_i} \\
&= RT\left(1 - \frac{1}{Z-B} + \frac{A}{(Z+\epsilon B)(Z+\sigma B)}\right)\frac{\partial Z}{\partial n_i} + RT\left(\frac{1}{Z-B} - \frac{Z}{B}\frac{A}{(Z+\epsilon B)(Z+\sigma B)}\right)\frac{\partial B}{\partial n_i}.
\end{aligned} \tag{171}$$

We now show that the expression in the left-most parentheses is zero. We modify the three terms such that they have the same denominator:

$$\begin{aligned}
&1 - \frac{1}{Z-B} + \frac{A}{(Z+\epsilon B)(Z+\sigma B)} \\
&\qquad = \frac{B(Z+\epsilon B)(Z+\sigma B)(Z-B) - B(Z+\epsilon B)(Z+\sigma B) + AZ(Z-B)}{B(Z+\epsilon B)(Z+\sigma B)(Z-B)}.
\end{aligned} \tag{172}$$

Next, we write out the terms in the numerator and express it as cubic polynomium in Z:

$$\begin{aligned}
&(Z+\epsilon B)(Z+\sigma B)(Z-B) - (Z+\epsilon B)(Z+\sigma B) + A(Z-B) \\
&\qquad = \left[Z^3 + (\epsilon+\sigma)BZ^2 + \epsilon\sigma B^2 Z - BZ^2 - (\epsilon+\sigma)B^2 Z - \epsilon\sigma B^3\right] \\
&\qquad\quad - \left[Z^2 + (\epsilon+\sigma)BZ + \epsilon\sigma B^2\right] + [AZ - AB] \\
&= Z^3 + ((\epsilon+\sigma)B - B - 1)Z^2 + (\epsilon\sigma B^2 - (\epsilon+\sigma)B^2 - (\epsilon+\sigma)B + A)Z + (-\epsilon\sigma B^3 - \epsilon\sigma B^2 - AB) \\
&= Z^3 + (B(\epsilon+\sigma-1) - 1)Z^2 + (A - B(\epsilon+\sigma) + B^2(\epsilon\sigma - \epsilon - \sigma))Z + (-(AB + (B^2+B^3)\epsilon\sigma))
\end{aligned} \tag{173}$$

This expression is exactly the polynomium from the cubic equation of state, $q(Z)$. The compressibility factor, Z, must satisfy the cubic equation of state. The above expression is therefore zero, i.e. $q(Z) = 0$, and the first term in (171) is therefore also zero. We use this fact to rewrite the expression in the right-most parentheses in (171). That is, from

$$1 - \frac{1}{Z-B} + \frac{A}{(Z+\epsilon B)(Z+\sigma B)} = 0, \tag{174}$$

we isolate the term that contains A:

$$\frac{A}{(Z+\epsilon B)(Z+\sigma B)} = \frac{1}{Z-B} - 1. \tag{175}$$

We insert the above into the expression in the right-most parentheses in (171) and simplify:

$$\begin{aligned}
\frac{1}{Z-B} - \frac{Z}{B}\frac{A}{(Z+\epsilon B)(Z+\sigma B)} &= \frac{1}{Z-B} - \frac{Z}{B}\left(\frac{1}{Z-B} - 1\right) \\
&= \frac{1}{Z-B} - \frac{Z}{B}\left(\frac{1-(Z-B)}{Z-B}\right) \\
&= \frac{B - Z(1-(Z-B))}{B(Z-B)} \\
&= \frac{B - Z + Z^2 - BZ}{B(Z-B)} \\
&= \frac{B(1-Z) - Z(1-Z)}{B(Z-B)} \\
&= \frac{-(Z-B)(1-Z)}{B(Z-B)} \\
&= \frac{Z-1}{B}.
\end{aligned} \tag{176}$$

The expression for $\partial g^R/\partial n_i$ therefore becomes

$$\frac{\partial g^R}{\partial n_i} = RT\frac{Z-1}{B}\frac{\partial B}{\partial n_i} - \frac{1}{\epsilon-\sigma}\frac{1}{b_m}\left(\frac{\partial a_m}{\partial n_i} - \frac{a_m}{b_m}\frac{\partial b_m}{\partial n_i}\right)\ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right). \tag{177}$$

We write out and simplify the first term:

$$\begin{aligned}
RT\frac{Z-1}{B}\frac{\partial B}{\partial n_i} &= RT\frac{Z-1}{B}\frac{P}{RT}\frac{\partial b_m}{\partial n_i} \\
&= RT(Z-1)\frac{1}{b_m}\frac{b_i - b_m}{N} \\
&= \frac{RT}{N}(Z-1)\left(\frac{b_i}{b_m} - 1\right).
\end{aligned} \tag{178}$$

The final expression for $\partial g^R/\partial n_i$ is thus

$$\frac{\partial g^R}{\partial n_i} = \frac{RT}{N}(Z-1)\left(\frac{b_i}{b_m} - 1\right) - \frac{1}{\epsilon-\sigma}\frac{1}{b_m}\left(\frac{\partial a_m}{\partial n_i} - \frac{a_m}{b_m}\frac{\partial b_m}{\partial n_i}\right)\ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right). \tag{179}$$

We are now ready to consider the expression for $g^R(T,P,n) + N\partial g^R/\partial n_i$. We insert the expressions for $g^R(T,P,n)$ and $\partial g^R/\partial n_i$:

$$\begin{aligned}
g^R(T,P,n) + N\frac{\partial g^R}{\partial n_i} &= \left[RT(Z-1) - RT\ln(Z-B) - \frac{1}{\epsilon-\sigma}\frac{a_m}{b_m}\ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right)\right] \\
&\quad + N\left[\frac{RT}{N}(Z-1)\left(\frac{b_i}{b_m} - 1\right) - \frac{1}{\epsilon-\sigma}\frac{1}{b_m}\left(\frac{\partial a_m}{\partial n_i} - \frac{a_m}{b_m}\frac{\partial b_m}{\partial n_i}\right)\ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right)\right] \\
&= RT(Z-1)\frac{b_i}{b_m} - \frac{1}{\epsilon-\sigma}\left(\frac{a_m}{b_m} + \frac{N}{b_m}\left(\frac{\partial a_m}{\partial n_i} - \frac{a_m}{b_m}\frac{\partial b_m}{\partial n_i}\right)\right)\ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right).
\end{aligned} \tag{180}$$

We write out and simplify the second factor in the second term:

$$\begin{aligned}\frac{a_m}{b_m}+\frac{N}{b_m}\left(\frac{\partial a_m}{\partial n_i}-\frac{a_m}{b_m}\frac{\partial b_m}{\partial n_i}\right) &= \frac{a_m}{b_m}+\frac{N}{b_m}\left(\frac{2}{N}\left(\sum_j z_j a_{ij}-a_m\right)-\frac{a_m}{b_m}\frac{b_i-b_m}{N}\right)\\ &= \frac{1}{b_m}\left(a_m+2\left(\sum_j z_j a_{ij}-a_m\right)-a_m\frac{b_i}{b_m}+a_m\right)\\ &= \frac{1}{b_m}\left(2\sum_j z_j a_{ij}-a_m\frac{b_i}{b_m}\right). \end{aligned} \tag{181}$$

The final expression for $g^R(T,P,n)+N\partial g^R/\partial n_i$ is

$$g^R(T,P,n)+N\frac{\partial g^R}{\partial n_i} = RT(Z-1)\frac{b_i}{b_m}-\frac{1}{\epsilon-\sigma}\frac{1}{b_m}\left(2\sum_j z_j a_{ij}-a_m\frac{b_i}{b_m}\right)\ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right), \tag{182}$$

and we therefore obtain the following expression for the logarithmic fugacity coefficients:

$$\begin{aligned}\ln\phi_i(T,P,n) &= \frac{1}{RT}\left(g^R(T,P,n)+N\frac{\partial g^R}{\partial n_i}\right)\\ &= (Z-1)\frac{b_i}{b_m}-\frac{1}{\epsilon-\sigma}\frac{1}{RTb_m}\left(2\sum_j z_j a_{ij}-a_m\frac{b_i}{b_m}\right)\ln\left(\frac{Z+\epsilon B}{Z+\sigma B}\right). \end{aligned} \tag{183}$$

References

Callen, H. B., 1985. Thermodynamics and and Introduction to Thermostatistics. John Wiley & Sons.

Gaspar, J., Ritschel, T. K. S., Jørgensen, J. B., 2017. An efficient and rigorous thermodynamic library and optimal-control of a cryogenic air separation unit. In: Espuna, A., Graells, M., Puigjaner, L. (Eds.), 27th European Symposium on Computer Aided Process Engineering - ESCAPE 27. Barcelona, Spain.

Michelsen, M., Mollerup, J., 2007. Thermodynamic Models: Fundamentals and Computational Aspects. Tie-Line Publications.

Michelsen, M. L., 1980. Calculation of phase envelopes and critical points for multicomponent mixtures. Fluid Phase Equilibria 4 (1-2), 1–10.

Peng, D.-Y., Robinson, D. B., 1976. A new two-constant equation of state. Ind. Eng. Chem. Fundam. 15 (1), 59–64.

Ritschel, T. K. S., Gaspar, J., Capolei, A., Jørgensen, J. B., 2016. An open-source thermodynamic software library. Tech. Rep. DTU Compute Technical Report-2016-12, Department of Applied Mathematics and Computer Science, Technical University of Denmark.

Ritschel, T. K. S., Gaspar, J., Jørgensen, J. B., 2017. A thermodynamic library for simulation and optimization of dynamic processes. In: Proceedings of the 20th World Congress of the International Federation of Automatic Control.

Soave, G., 1972. Equilibrium constants from a modified redlich-kwong equation of state. Chemical Engineering Science 27 (6), 1197–1203.

Thomson, G., 1996. The DIPPR® databases. Int. J. Thermophys. 17 (1), 223–232.

Wilson, G. M., 1969. A modified Redlich-Kwong equation of state, application to general physical data calculations. In: Proceedings of the 65th National AIChE Meeting. Cleveland, OH.