Exceptionally reliable density-solving algorithms for multiparameter mixture models from Chebyshev expansion rootfinding*

Ian H. Bell and Bradley K. Alpert

March 8, 2018

Abstract

Calculation of the density of a mixture for a given temperature, pressure, and composition from multi-parameter Helmholtz-energy-explicit mixture models (often referred to as the GERG formulation) sometimes fails; failures are caused by insufficiently accurate estimations of the density root, but also by insufficiently robust numerical methods that may not converge to the desired density solution. Furthermore, generally only one density solution is located at a time.

Polynomial expansions have the characteristic that *all* roots of the expansion can be obtained reliably. Therefore the approach we propose is to develop a very good approximation of the equation of state based on Chebyshev orthogonal polynomial expansions, and solve for all the roots of the Chebyshev expansion - proxies for the roots of the equation of state. In this paper, we limit ourselves to the case where the temperature is known; the method is generalizable to other types of thermodynamic calculations.

For mixtures, this Chebyshev proxy rootfinding results in a density calculation that is almost guaranteed to yield the right solution. These tools make multi-parameter equations of state nearly as reliable as cubic equations of state. The computational penalty from the use of Chebyshev expansions can be overcome through the exploitation of parallelism, though that is not further discussed here.

This method is implemented in C++11; the code is provided in the supplemental material.

1 Introduction

This paper synthesizes the work of two disparate fields: the applied mathematics domain of orthogonal polynomials and their use as proxies for non-linear rootfinding (see for instance Boyd [5] or the work of the `chebfun` [10] project), and the thermodynamic modeling of mixtures by highly-accurate multiparameter mixture models. As such, references are given, where appropriate, to orient readers coming from each field. The goal is that the mathematics be explained clearly enough that the user could implement these numerical tools themselves without any additional references. The C++ code in the supplemental information should also serve as a useful reference as it is practically a one-to-one translation of the equations presented in this work. In the case of discrepancies between the code and the equations here, the code should be relied upon in preference to the equations in this work.

A thermodynamic equation of state, in the most general sense, represents the link between disparate thermodynamic properties. The canonical example is that of van der Waals[33]

$$p = \frac{RT}{v-b} - \frac{a}{v^2}. \tag{1}$$

This equation of state is the ancestor of much of today's thermodynamic modeling. Cubic equations of state, of which van der Waals is but one [32, 1, 25], have the characteristic that all molar volume values that satisfy $p_{\rm EOS}(T,v) = p_{\rm target}$ can be obtained by solving a cubic equation explicit in molar volume; the cubic equation can be solved analytically. The roots of *all* polynomial equations can be obtained as matrix eigenvalues[14, p. 348]. At its core, this concept was the inspiration for this work.

The multiparameter equations of state employed in the state of the art libraries NIST REFPROP[20], CoolProp[3], or TREND[29] are all of the Helmholtz-energy-explicit formulation. The molar Helmholtz energy $a(T,v)$ is one of the fundamental thermodynamic potentials from which all other thermodynamic properties can be obtained from the appropriate derivatives. We further express the specific Helmholtz energy a as the sum of ideal-gas a^0 and residual $a^{\rm r}$ contributions. In practice, nondimensionalized Helmholtz energy terms are employed, where the nondimensional residual Helmholtz energy would be given by $\alpha^{\rm r} = a^{\rm r}/(RT)$. Other properties can then be obtained from derivatives of the nondimensionalized Helmholtz energy α. For instance the pressure can be obtained from

$$\frac{p}{\rho RT} = 1 + \delta\left(\frac{\partial \alpha^{\rm r}}{\partial \delta}\right)_{\tau}, \tag{2}$$

where $\delta = \rho/\rho_{\rm r}(\vec{x})$ and $\tau = T_{\rm r}(\vec{x})/T$. The reducing func-

*Commercial equipment, instruments, or materials are identified only in order to adequately specify certain procedures. In no case does such identification imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the products identified are necessarily the best available for the purpose.

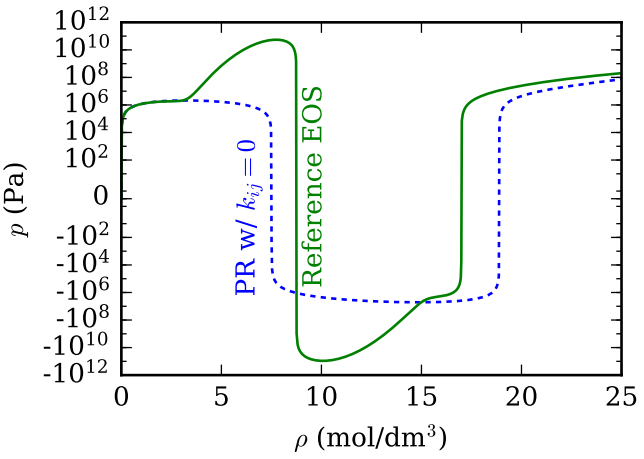

Figure 1: 180 K isotherm for an equimolar mixture of nitrogen + ethane, with Peng-Robinson (with $k_{ij} = 0$) and the multi-fluid GERG model[18] with reference equations of state for the pure fluids[30, 6]

tions $T_r(\vec{x})$ and $\rho_r(\vec{x})$ are a function of composition and are based on the GERG model. For more information on obtaining properties from this type of model, refer to the literature [19, 18, 11, 12, 2].

While the functional forms of these multiparameter models are much more complicated than that of van der Waals (as required by the need to develop equations of state that are able to replicate the most accurate experimental measurements to within their experimental uncertainties), it remains the case that the density rootfinding problem for multiparameter EoS can be expressed in the analogous form $p_{\rm EOS}(\tau, \delta) = p_{\rm target}$. Therefore, for a given value of τ, a 1-D rootfinding problem in δ is required.

In the state of the art literature on the topic of density rootfinding with multi-fluid models, that of Gernert *et al.* [12], a number of heuristics are used in order to ensure reliability of the density solver with multi-fluid mixture models. While these heuristics and guidelines help the density solver yield more reliable density roots, there are many degenerate cases that need special treatment. For instance, Fig. 1 demonstrates a particularly challenging density solving case for an equimolar mixture of nitrogen + ethane. In this case, the heuristics of Gernert *et al.* [12] of first finding the local pressure extrema and considering the density ranges outside the outermost local extrema becomes problematic because the local maximum at a pressure of 10^{10} Pa is difficult to locate without preknowledge of the shape of the isotherm. Similarly, the pressure minimum is at a very negative pressure of less than -10^{10} Pa, so common rules of thumb (e.g., consider only positive pressure) would cause failures of the calculation.

The key insight of this work is that you can obtain all the roots of $p_{\rm EOS}(\delta) - p_{\rm target} = 0$ when $p(\delta)$ is expressed for a given value of τ in the form of a polynomial in δ by solving for all the roots of the expansion. The expansion rootfinding represents an extremely reliable solution to the problem of rootfinding (though with some minor loss of accuracy of approximation of the underlying function), rather than an iterative solution with extensive user-assistance and handling of rare but important corner cases. This fact holds regardless of whether you consider monomial bases like $x^0, x^1, x^2, \ldots$, or orthogonal polynomials (e.g., Legendre polynomials or Chebyshev polynomials). The challenge of bridging the worlds of Chebyshev proxy rootfinding and equation of state modeling is the thrust of this work.

2 Background: Chebyshev expansions

Let us first begin with an exposition of the use of orthogonal polynomial expansions as a means of approximating continuous functions in the interval $[-1, 1]$. As will be shown later, any other interval of definition can be scaled to [-1,1]. Two functions f_i and f_j are said to be orthogonal on an interval I for a given weight function $w(x) \geq 0$ if the weighted integral of their product on I vanishes

$$\int_I f_i(x) f_j(x) w(x) dx = 0. \tag{3}$$

The Chebyshev polynomials are the family of orthogonal functions obtained with the weight function $w(x) = 1/\sqrt{1-x^2}$ on the interval I=[-1,1].

The approximation of a function f by a polynomial p represented as a sum of monomials, e.g.,

$$f(x) \approx p(x) = \sum_{i=0}^{n} c_i x^i \tag{4}$$

becomes very ill-conditioned for an order n that is not small and may yield catastrophically incorrect values on finite-precision processors. As is described throughout the numerical analysis literature, the use of orthogonal polynomials as basis functions for the expansion (instead of monomial terms like x^i) is numerically preferable. Orthogonal polynomials have many fascinating properties, but the most important ones for our purposes here are two-fold:

1. With a sufficient number of terms in the orthogonal polynomial expansion, a continuous function can be approximated to any desired accuracy over a closed domain $[x_{\rm min}, x_{\rm max}]$, and

2. All roots of an orthogonal polynomial expansion of a function can be obtained as the eigenvalues of a companion matrix. Alternatively, bounded iterative solvers can be used to find the roots of the expansion in a reliable manner.

In the case of Chebyshev expansions (a specific case of orthogonal polynomial expansion), the function is approximated by the expansion

$$f(x) \approx p(x) = \sum_{i=0}^{n} c_i T_i(x) \tag{5}$$

where $T_i(x)$ is the Chebyshev polynomial of the first kind of degree i.

2.1 Chebyshev polynomials

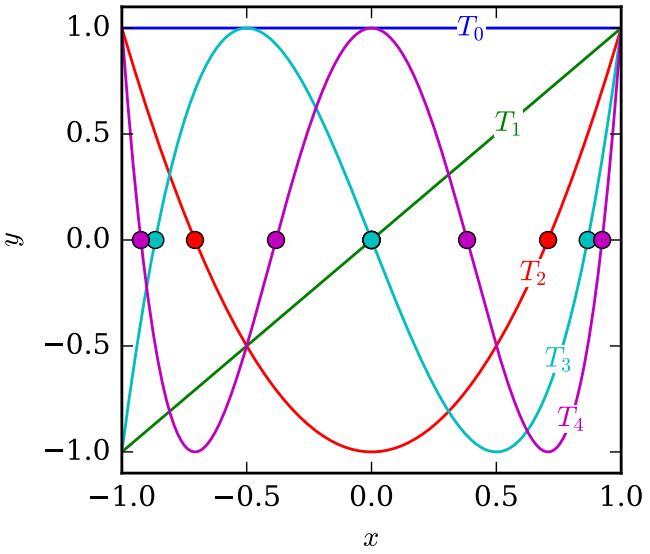

Figure 2: Chebyshev polynomials up to order 4

The Chebyshev polynomials of the first kind of the first few orders are shown in Fig. 2. They can be obtained from the three-term recurrence relationship (see for instance DLMF [8, (3.11.7)])

$$\begin{aligned} T_0(x) &= 1 && (6)\\ T_1(x) &= x && (7)\\ T_{n+1}(x) &= 2xT_n(x) - T_{n-1}(x), \quad n \geq 1 && (8) \end{aligned}$$

or for $x \in [-1, 1]$, given in explicit form as

$$T_n(x) = \cos(n \arccos x). \tag{9}$$

The n roots of the n-th order Chebyshev polynomial $T_n(x)$ are given by (see for instance DLMF [8, (18.3.2)])

$$x_k^n = \cos\left(\frac{(k-\frac{1}{2})\pi}{n}\right), \quad k = 1, \ldots, n, \tag{10}$$

where the roots are symmetric around $x = 0$.

2.2 Evaluation of Chebyshev expansion

The naïve treatment of the evaluation of the Chebyshev expansion given by Eq. (5) is to simply evaluate each Chebyshev basis function T_i at x, multiply by its corresponding coefficient c_i, and sum up the contributions; more computationally efficient alternatives are available. When a single value of the Chebyshev expansion is desired, all $n+1$ values of $T_0(x), \ldots, T_n(x)$ can be obtained from the recurrence relationship given by Eqs. (6) to (8) as step 1, and each polynomial value is then multiplied by its coefficient and summed, as step 2. Furthermore, when only a single function evaluation is needed for one value of x, Clenshaw's method can be used to evaluate the expansion with fewer steps than are used for the summation of recurrence terms (see for instance DLMF [8, (3.11.15)]).

Equivalently, for a vector $\vec{x}$ of abscissae,

$$f_n(\vec{x}) = \mathbf{A}\,\vec{c} \tag{11}$$

where $\vec{c}$ is the column vector of coefficients of the expansion from c_0 to c_n, with degree increasing going down the column, and $\mathbf{A}$ is a matrix with as many rows as entries in $\vec{x}$ and $n+1$ columns, with the form

$$\mathbf{A}(\vec{x}) = \begin{bmatrix} | & | & & | \\ T_0 & T_1 & \ldots & T_n \\ | & | & & | \end{bmatrix}, \tag{12}$$

where the recurrence relationship allows us to fill in the $\mathbf{A}$ matrix by columns:

$$\begin{aligned} T_0 &= 1 && (13)\\ T_1 &= \vec{x} && (14)\\ T_i &= 2\vec{x}T_{i-1} - T_{i-2}, \quad i \geq 2. && (15) \end{aligned}$$

The same paradigm can be employed when several expansions $(C_0, \ldots, C_n)$ are to be evaluated for a single value of x. In this way, the "matrix" $\mathbf{A}$ becomes a row vector, which is then multiplied by the matrix with the coefficients of each expansion $\mathbf{c}_0$ to $\mathbf{c}_n$ in each column, with degree increasing going down the column:

$$f_n(\vec{x}) = \mathbf{A} \begin{bmatrix} | & | & & | \\ \vec{c}_0 & \vec{c}_1 & \ldots & \vec{c}_n \\ | & | & & | \end{bmatrix}. \tag{16}$$

Evaluation of this vector-matrix product results in a row vector with the values of each expansion in each column.

The expansion Eq. (5) containing $n+1$ terms is well-represented by its values at the $n+1$ roots of T_{n+1}. It is even more convenient to evaluate it at the Chebyshev-Lobatto nodes, or "practical Chebyshev nodes," given by the values

$$x_{n,j} = \cos\left(\frac{j\pi}{n}\right) \tag{17}$$

for j between 0 and n, where the first and last nodes are the values 1 and -1, respectively. Expansion values at these nodes can be obtained through multiplication with the $(n+1) \times (n+1)$ matrix $\mathbf{U}$ with elements u_{jk} given by

$$u_{jk} = \cos\left(\frac{\pi jk}{n}\right) \tag{18}$$

where j and k range from 0 to n, inclusive. By inspection, this matrix is symmetric, and thus can be constructed by populating half of the matrix and filling the other half of the matrix by symmetry. The values of the function at these Chebyshev-Lobatto nodes can be obtained from the matrix-vector product

$$\vec{f_n} = \mathbf{U}\vec{c} \tag{19}$$

where $\vec{c}$ is the column vector of coefficients of the Chebyshev expansion. The inverse transformation (from nodal functional values to coefficients) can be carried out through the use of the $(n+1) \times (n+1)$ matrix $\mathbf{V}$ with elements v_{jk}:

$$v_{jk} = \frac{2}{np_n(j)p_n(k)} \cos\left(\frac{\pi jk}{n}\right) \tag{20}$$

where $p_n(i)$ is 2 if $i = 0$ or $i = n$ and $p_n(i) = 1$ otherwise. The transformation from expansion values at the Chebyshev-Lobatto nodes $\vec{f_n}$ given by Eq. (17) to coefficients $\vec{c}$ is given by

$$\vec{c} = \mathbf{V}\vec{f_n}. \tag{21}$$

2.3 Special Chebyshev expansions

While the expansion coefficients for most functions are determined from their values at the Chebyshev-Lobatto nodes following Eq. (21), certain functions have analytically known expansion coefficients. In the case of monomial terms (the basis functions in Eq. (4)), each monomial term can be converted to a Chebyshev expansion through the use of (see Mason and Handscomb [24, p. 22] or Gil et al. [13, p. 59])

$$x^n = 2^{1-n} {\sum_{k=0}^{\lfloor n/2 \rfloor}}' \binom{n}{k} T_{n-2k}(x) \tag{22}$$

for $n \geq 0$, where $\lfloor p \rfloor$ is the largest integer not exceeding p. For instance $\lfloor 3/2 \rfloor = 1$ and $\lfloor 4/2 \rfloor = 2$. ${}_nC_k = \binom{n}{k} = \frac{n!}{k!(n-k)!}$ is the binomial coefficient. The single prime $'$ means that the term containing $T_0(x)$ is to be halved. For instance, coefficients of the Chebyshev expansions of the first few monomials are:

n	c_0	c_1	c_2	c_3	c_4	c_5
0	1					
1	0	1				
2	$\frac{1}{2}$	0	$\frac{1}{2}$			
3	0	$\frac{3}{4}$	0	$\frac{1}{4}$		
4	$\frac{3}{8}$	0	$\frac{1}{2}$	0	$\frac{1}{8}$	
5	0	$\frac{5}{8}$	0	$\frac{5}{16}$	0	$\frac{1}{16}$

2.4 Roots of a Chebyshev expansion

For an arbitrary nonlinear function of one variable $f(x)$, there are no readily available insights about the existence, quantity, or location of roots of the function (values of x where $f(x) = 0$). This unfortunate reality has caused many sleepless nights for those professionally engaged in taming beastly nonlinear functions.

There are innumerable numerical methods that can be used to find roots of a nonlinear function, but in general they require reasonable estimates for the location of the root and/or bounds on the location of the root. In almost all cases they find roots one at a time, even if many roots may be present. Algorithms in this family would be Newton, Secant, Halley's method, Brent, etc.

2.4.1 Via eigenvalues

One of the advantages of the use of orthogonal polynomial expansions to approximate nonlinear functions is that *all* roots of an orthogonal polynomial expansion can be obtained as eigenvalues of the companion matrix of the expansion, thereby hiding root-finding iterations in a mature black box routine for eigenvalue solving. With few exceptions, this process is highly reliable. The companion matrix of a Chebyshev expansion of degree n is defined as the matrix $\mathbf{A}$ with entries [5, Appendix B]

$$A_{jk} = \begin{cases} K_{2,k} & j = 1, k = 1 \ldots n \\ \frac{1}{2}K_{j,k+1} + \frac{1}{2}K_{j,k-1} & j = 2 \ldots n-1, k = 1 \ldots n \\ -\frac{a_{j-1}}{2a_n} + \frac{1}{2}K_{k,n-1} & j = n, k = 1 \ldots n \end{cases} \tag{23}$$

where $K_{i,j}$ is the Kronecker delta, given to be 1 if $i = j$, and 0 otherwise. The Kronecker delta is typically given the symbol $\delta_{i,j}$, but we use the symbol $K_{i,j}$ here to avoid confusion with the reduced density of the equation of state.

The primary challenge of this direct method for finding the roots of the Chebyshev expansion is that it is (relatively) quite slow. The determination of the eigenvalues of a 50×50 matrix with randomly generated values between 0 and 1 takes approximately 700 μs[1]. While an evaluation time of all the roots of a 50^{th} degree polynomial of less than a millisecond might sound quite rapid, and for many problems is entirely adequate, here, higher throughput is needed. The companion matrix is lower Hessenberg (all values above the first superdiagonal are zero); this allows for some potential computational optimization because the first step of most eigenvalue solving algorithms is a reduction to Hessenberg form.

As explained elsewhere [5], adequate speed of the eigenvalue solver can be obtained by splitting the overall interval into multiple intervals, and then for each interval, solving a smaller eigenvalue problem. The real roots are then collected over each of the intervals. Due to the expense of evaluating the eigenvalues of the companion matrix, it can be beneficial to apply heuristics to the subinterval to determine the likelihood of a root being

[1] From numpy 0.11 linked with Intel MKL library, 64-bit Windows 7 PC with 16 GB of RAM and Intel Core i7-3770 CPU @ 3.4GHz

present. For instance, if all the values obtained from the matrix product given in Eq. (19) at the Chebyshev-Lobatto nodes given by Eq. (17) are "far" away from zero ("far" being a user-specified tolerance), the non-existence of a root in that interval can be assumed. Evaluation of Eq. (19) requires a matrix-vector product, but the **U** matrix can be cached, as it is only a function of the expansion degree and not the expansion itself. The matrix-vector product in Eq. (19) is far less computationally costly than the evaluation of the eigenvalues. In many practical cases, most of the intervals will not have roots, and the amount of computational effort that can be avoided through the elimination of the eigenvalue solver is very significant, well worth the cost of a number of matrix-vector products.

Furthermore, although we will not discuss this point further here, the process of dividing the overall interval and obtaining the eigenvalues for each subinterval is an embarrassingly parallel problem, extremely well suited to multi-threading or multi-processor paradigms. This offers the potential for additional computational throughput benefits. Of course, the same parallelization methods could be used in the evaluation of the full equation of state, so for a fair comparison we limit ourselves to single-threaded computational efficiency comparisons.

2.4.2 Iterative Chebyshev rootfinding

While the eigenvalue solver for the roots of the Chebyshev expansion, is with few exceptions (see Section 4.5), extremely reliable, its fundamental limitation is its speed. As the computational cost of the rootfinding method is of the highest importance, it is therefore also crucial to obtain the roots of the Chebyshev expansion as quickly as possible.

According to the intermediate value theorem, if a continuous function $f(x)$ has two abscissa a and b for which $f(a)f(b) < 0$ (the values of a and b bracket the root), there must be at least one root in the interval. Therefore, a bounded method may be employed that progressively reduces the width of the bracket around the root until sufficient convergence is reached. At each of the Chebyshev-Lobatto nodes, the representation of the function is "perfect" (to numerical precision). The Chebyshev-Lobatto nodes are reliable determinants of whether the functional value is positive or negative; if the functional values at two neighboring Chebyshev-Lobatto nodes are of opposite signs, then a bounded iteration can be used to locate the root(s) between them. In theory, it is possible that there are arbitrarily many roots between the nodes; the intermediate value theorem just tells us that that there is *at least* one root in the domain.

In our treatment, the first step is to oversample the expansion at Chebyshev-Lobatto nodes with a degree twice that of the underlying expansion (injecting a new Chebyshev-Lobatto node between each adjacent pair of existing Chebyshev-Lobatto nodes), and then at each set of three nodes for the oversampled expansion, a quadratic is fit. If the quadratic has no real roots in the domain, the domain is ignored and the next set of three nodes is tried. If the sign at the midpoint of the quadratic differs from those at the endpoints, there are two real roots, and each half of the quadratic needs to be considered. If the quadratic does have real roots, then the secant method is used to polish the root(s). As an alternative, improved regula falsi methods [9, 17] could be used to find the bracketed root, but that proved unnecessary in this case because the computational time is not dominated by the rootfinding.

We selected the iterative rootfinding algorithm because it proved to be as reliable as the eigenvalue solver in practice, but also significantly faster. In this case, we also apply the rootfinding avoidance heuristic of skipping intervals where the functional values at all the Chebyshev-Lobatto nodes are "far away" from zero. For more information on the method employed in this work, see the C++ source code, in particular the `real_roots2` function.

2.5 Operations on Chebyshev expansions

In this section we assume that we have two Chebyshev expansions A (of degree n) and B (of degree m) given as expansions in terms of the basis functions $T_i(x)$ by

$$A(x) = \sum_{i=0}^{n} a_i T_i(x) \tag{24}$$

$$B(x) = \sum_{i=0}^{m} b_i T_i(x) \tag{25}$$

The operations detailed in this section will be carried out on expansions A and B, yielding the expansion C:

$$C(x) = \sum_{i=0}^{l} c_i T_i(x). \tag{26}$$

2.5.1 Addition and multiplication by a constant

The sum of A and B is a Chebyshev expansion of degree equal to the maximum of the degrees of A and B. The coefficients of the Chebyshev expansion $C = A + B$ are equal to $\vec{c} = \vec{a} + \vec{b}$, where the coefficients of the lower degree expansion are right padded with zeros prior to the element-wise sum. This would yield something like the following pseudocode:

```
a = [1,2,5]
b = [0,1] # right-padded to [0,1,0]
c = [1,3,5] # c = a+b
```

Multiplication of a Chebyshev expansion by a constant simply multiplies each coefficient by the constant, so $C = kA$ would yield $\vec{c} = k\vec{a}$

```
a = [1,2,5]
k = 2
c = [2,4,10] # c = k*a
```

2.5.2 Multiplication of two expansions

Multiplication of two Chebyshev expansions can be attacked by a number of alternative methods, each with tradeoffs in terms of complexity and/or computational efficiency. One of the simplest, though perhaps not the most efficient, means of multiplying two expansions is to convert each expansion to nodal functional values, multiply the nodal values together, and then carry out the inverse transformation from nodal functional values to coefficients for the product. The product of two Chebyshev expansions A (of degree n) and B (of degree m) is of degree $l = n + m$. To be more specific, we carry out the following procedure to multiply two Chebyshev expansions together:

1. Pad out the coefficients for $\vec{a}$ and $\vec{b}$ to the degree of $n+m$ with zeros (each vector of coefficients now has $n + m + 1$ elements in it).
2. Calculate the functional values $\vec{f}_{l,a}$ and $\vec{f}_{l,b}$ at the Chebyshev-Lobatto nodes from Eq. (19) for padded expansions A and B respectively.
3. Multiply the vector of functional values at the Chebyshev-Lobatto nodes together (in an element-wise sense) to yield $\vec{f}_{l,ab} = \vec{f}_{l,a}\vec{f}_{l,b}$.
4. Pre-multiply $\vec{f}_{l,ab}$ by $\mathbf{V}$ of degree $l = n + m$ from Eq. (20) to yield the coefficient values for the product from Eq. (21).

The generalized product treatment is unnecessarily involved for the specific case of a Chebyshev expansion multiplied by its independent variable x. The expansion of $C = xA$, where A is an n-th degree expansion, can be given by

$$C = \sum_{i=0}^{n+1} c_i T_i(x) = xA = \sum_{i=0}^{n} a_i (xT_i(x)) \tag{27}$$

where the general equation for $xT_i(x)$ can be given by Mason and Handscomb[p. 31][24]

$$xT_i(x) = \frac{1}{2}\left(T_{i+1}(x) + T_{|i-1|}(x)\right) \tag{28}$$

which yields the form of the coefficients c_i:

$$c_i = \begin{cases} \frac{1}{2}a_1 & i = 0 \\ a_0 + \frac{1}{2}a_2 & i = 1 \\ \frac{1}{2}a_{i-1} + \frac{1}{2}a_{i+1} & 2 \le i \le n-1 \\ \frac{1}{2}a_{i-1} & n \le i \le n+1. \end{cases} \tag{29}$$

A more efficient implementation of this method avoiding re-allocation of arrays of coefficients can be found in the C++ code in the supplemental material, in particular the `times_x_inplace` function.

2.5.3 Derivatives

A Chebyshev expansion A can be differentiated with respect to the independent variable x and expressed in the form

$$\frac{\mathrm{d}A}{\mathrm{d}x} = \sum_i a_i \frac{\mathrm{d}T_i(x)}{\mathrm{d}x}, \tag{30}$$

where the derivative of the Chebyshev basis functions are given by

$$\frac{\mathrm{d}T_i(x)}{\mathrm{d}x} = \begin{cases} 2i \sum_{j=0}^{(i-2)/2} T_{i-1-2j}(x) & i \text{ even}, \\ iT_0(x) + 2i \sum_{j=0}^{(i-3)/2} T_{i-1-2j}(x) & i \text{ odd}, \end{cases} \tag{31}$$

where $T_0(x) = 1$ and $T_1(x) = x$. The same procedure can be applied to higher derivatives, and the vectorized evaluation proposed above can also be applied to the evaluation of the derivative(s). This recurrence relationship yields each of the terms in Eq. (30).

It is possible to express the entire derivative in an explicit form in terms of the original, undifferentiated basis functions according to the method proposed in Mason and Handscomb[p. 34][24]. The derivative operation for the expansion A of degree n yields a derivative of degree $n - 1$. The derivative is given by [p. 34][24]:

$$\frac{\mathrm{d}A}{\mathrm{d}x} = \sum_{i=0}^{n-1}{}' d_i T_i(x). \tag{32}$$

where the $'$ on the summation indicates that the $i = 0$ term is to be divided by two, and where the coefficients d_i are given by

$$d_i = \sum_{k=i+1}^{n} \begin{cases} 2ka_k & k - i \text{ odd}, \\ 0 & k - i \text{ even}. \end{cases} \tag{33}$$

One of the unfortunate numerical features of derivatives, which holds true for Chebyshev expansions as well, is that differentiation results in a lower accuracy derivative. As a trivial but illustrative example, a 10^{th} degree expansion of $\exp(x)$ is developed; the derivative of $\exp(x)$ is equal to itself and therefore, the coefficients of the expansion should in theory be equal to those of the derivative. Taking 3, 6, and 9 derivatives of $\exp(x)$ yields the values in Table 1. As each derivative is taken, the expansion deviates more and more from its analytic solution. This demonstrates the destructive influence of differentiation; in order to retain the maximum accuracy, differentiation should be avoided if at all possible.

2.6 Other intervals

As described at the beginning of this section, Chebyshev polynomials, and therefore, Chebyshev expansions, are defined in the domain [-1, 1]. In practice, many functions

Table 1: The coefficients of an expansion of $\exp(x)$ in [-1,1] and the coefficients of the 3^{rd}, 6^{th}, and 9^{th} order derivatives. Only a few significant digits are retained for display purposes.

i	c_i	$c_i^{(3)}$	$c_i^{(6)}$	$c_i^{(9)}$
0	$1.266{\cdot}10^{0}$	$1.266{\cdot}10^{0}$	$1.266{\cdot}10^{0}$	$1.028{\cdot}10^{0}$
1	$1.130{\cdot}10^{0}$	$1.130{\cdot}10^{0}$	$1.128{\cdot}10^{0}$	$1.023{\cdot}10^{0}$
2	$2.715{\cdot}10^{-1}$	$2.715{\cdot}10^{-1}$	$2.712{\cdot}10^{-1}$	
3	$4.434{\cdot}10^{-2}$	$4.434{\cdot}10^{-2}$	$4.282{\cdot}10^{-2}$	
4	$5.474{\cdot}10^{-3}$	$5.474{\cdot}10^{-3}$	$5.328{\cdot}10^{-3}$	
5	$5.429{\cdot}10^{-4}$	$5.429{\cdot}10^{-4}$		
6	$4.498{\cdot}10^{-5}$	$4.460{\cdot}10^{-5}$		
7	$3.198{\cdot}10^{-6}$	$3.171{\cdot}10^{-6}$		
8	$1.992{\cdot}10^{-7}$			
9	$1.106{\cdot}10^{-8}$			
10	$5.506{\cdot}10^{-10}$			

are defined on other closed intervals $[x_{\min}, x_{\max}]$. It can therefore be of interest to scale from real-world coordinates in $[x_{\min}, x_{\max}]$ to scaled coordinates in [-1,1]. This transformation is a linear mapping between the real-world $x_{\rm rw}$ and the scaled coordinate x:

$$x = \frac{2x_{\rm rw} - (x_{\max} + x_{\min})}{x_{\max} - x_{\min}} \tag{34}$$

and the inverse transformation is given by

$$x_{\rm rw} = \frac{x_{\max} - x_{\min}}{2}x + \frac{x_{\max} + x_{\min}}{2}. \tag{35}$$

For all of the operations given in this section, the interval of applicability $[x_{\min}, x_{\max}]$ must be the same for both expansions involved in the transformation. In some cases, the transformation must be invoked as part of the operation, for instance, in the product $x_{\rm rw} \cdot A(x_{\rm rw})$. The reader is directed to the C++ code in the supplemental material, in particular the `times_x_inplace` function, for a description of how that transformation should be carried out. In most other cases, no special treatment is required.

3 Chebyshev deconstruction

In the GERG framework [19, 18], mixtures are formed as the sum of the pure fluids in a corresponding states formulation $\alpha^{\rm r}_{\rm C}$, along with a departure term $\alpha^{\rm r}_{\rm D}$ to correct the mixture thermodynamics. In this framework, the residual Helmholtz energy is given by

$$\alpha^{\rm r}(\tau, \delta, \vec{z}) = \alpha^{\rm r}_{\rm C}(\tau, \delta, \vec{z}) + \alpha^{\rm r}_{\rm D}(\tau, \delta, \vec{z}), \tag{36}$$

where the corresponding states contribution (in which each of the pure fluids is evaluated at the same *reduced* state) is given by

$$\alpha^{\rm r}_{\rm C} = \sum_{i=0}^{n_{\rm c}} z_i \alpha^{\rm r}_{{\rm o},i}(\tau, \delta), \tag{37}$$

and the departure term is given by the form

$$\alpha^{\rm r}_{\rm D} = \sum_{i=0}^{n_{\rm c}} \sum_{j=i+1}^{n_{\rm c}} F_{ij} z_i z_j \alpha^{\rm r}_{ij}(\tau, \delta). \tag{38}$$

Each of the individual contributions $\alpha^{\rm r}_{{\rm o},i}$ and $\alpha^{\rm r}_{ij}$ are entirely empirical in nature, having been fit to (hopefully) large and comprehensive sets of experimental data.

The residual Helmholtz contribution for all but a few pure fluids (some exceptions being water, carbon dioxide, refrigerant R-125, and methanol) can be expressed in the form

$$\alpha^{\rm r}_{{\rm o},i} = \sum_{k=0}^{k_{\max}} n_k \delta^{d_k} \tau^{t_k} \exp(u_k) \tag{39}$$

where u_k is given in a general form by

$$u_k = -c_{\delta,k}\delta^{l_{\delta,k}} - \eta_{2,k}(\delta - \varepsilon_{2,k})^2 - \beta_{2,k}(\tau - \gamma_{2,k})^2. \tag{40}$$

During the initialization of the mixture, the Chebyshev proxy for the mixture must be obtained. This proxy Chebyshev data structure is obtained as the deconstruction of the mixture model into its constituent pieces. The beauty is that while the Chebyshev deconstruction process is slow (compared to a single evaluation of the equation of state, though in absolute time it is fast), this deconstruction process must only be carried out a single time.

In this section we describe how to carry out the deconstruction of a generic term in the model, and then the overall algorithm that is used to deconstruct the entire mixture model. The result is a C++ data structure that stores the mixture model as a set of matrices of Chebyshev approximants to the multi-fluid model.

3.1 Generalized deconstruction

The non-dimensionalized Helmholtz energy contributions for the pure fluids and the departure term (with only a few exceptions) can be expressed in the separated analytic form

$$\alpha^{\rm r} = \sum_{k=0}^{k_{\max}} n_k F_k(\tau) G_k(\delta). \tag{41}$$

Each of the functions F_k and G_k are a function only of τ or δ, respectively.

The Chebyshev approximation to $\alpha^{\rm r}$

$$\tilde{\alpha}^{\rm r} \approx \alpha^{\rm r} \tag{42}$$

can therefore be expressed as the combination of the Chebyshev expansions of the F_k and G_k terms, or,

$$\tilde{\alpha}^{\rm r} = \sum_{k=0}^{k_{\max}} n_k \tilde{F}_k(\tau) \tilde{G}_k(\delta). \tag{43}$$

In principle, the Chebyshev expansion of $\alpha^{\rm r}$ results in a multivariate function of τ and δ, but in practice, this

expansion is used in the case either τ or δ is known and the goal is to be able to obtain all values of the other one that meet some condition (e.g., for a specified temperature, find all densities for which the pressure calculated from the equation of state is equal to a specified value).

The first partial derivatives of $\alpha^{\rm r}$ with respect to τ and δ, each multiplied by τ or δ, respectively, are given by

$$\tilde{A}_{10} = \tau \left(\frac{\partial \tilde{\alpha}^{\rm r}}{\partial \tau} \right)_{\delta} = \sum_{k=0}^{k_{\max}} n_k (\tau \tilde{F}'_k(\tau)) \tilde{G}_k(\delta) \tag{44}$$

and

$$\tilde{A}_{01} = \delta \left(\frac{\partial \tilde{\alpha}^{\rm r}}{\partial \delta} \right)_{\tau} = \sum_{k=0}^{k_{\max}} n_k \tilde{F}_k(\tau) [\delta \tilde{G}'_k(\delta)]. \tag{45}$$

Higher-order partial derivatives are evaluated in a similar fashion. Here we also define the syntax

$$\tilde{A}_{nm} = \tau^n \delta^m \left(\frac{\partial^{n+m}(\tilde{\alpha}^{\rm r})}{\partial \tau^n \partial \delta^m} \right) \tag{46}$$

for concision.

When τ is specified and δ is the independent variable to be detemined, the expansion of the derivative of $\alpha^{\rm r}$ with respect to δ can be given by

$$\tilde{A}_{01} = \begin{bmatrix} | & ... & | \\ \delta \tilde{G}'_0 & ... & \delta \tilde{G}'_{k_{\max}} \\ | & ... & | \end{bmatrix} \cdot \begin{bmatrix} n_0 \tilde{F}_0(\tau) \\ \vdots \\ n_{k_{\max}} \tilde{F}_{k_{\max}}(\tau) \end{bmatrix} \tag{47}$$

where $\delta \tilde{G}'_k$ is the column of coefficients for the Chebyshev expansion of the analytic term $\delta G'_k(\delta)$, with the degree increasing moving down the column. The right-most column vector is obtained by evaluating each Chebyshev expansion for $\tilde{F}$ at the specified value of τ. This matrix product results in the expansion for $\tilde{A}_{01}$ as a Chebyshev expansion in δ, as given by the form in Eq. (16).

3.1.1 Pure fluid terms

For pure fluids (except for water, carbon dioxide and a few others), F_k can be expressed in the form

$$F_k(\tau) = \tau^{t_k} \exp[-\beta_k(\tau - \gamma_k)^2], \tag{48}$$

and G_k can be expressed in the form

$$G_k(\delta) = \delta^{d_k} \exp[-c_{\delta,k}\delta^{l_{\delta,k}} - \eta_k(\delta - \varepsilon_k)^2], \tag{49}$$

where the derivative $\delta G'_k(\delta)$ is given by

$$\delta G'_k(\delta) = G_k(\delta) \left[d_k - 2\delta\eta_k(\delta - \varepsilon_k) - c_{\delta,k} l_{\delta,k} \delta^{l_{\delta,k}} \right]. \tag{50}$$

The particular form of the term is not important; the key point is that this method is applicable to any term that can be separated into two contributions, one that is a function of τ and the other that is a function of δ.

3.1.2 Mixture terms

For mixtures, the functional forms that have thus far been used in the literature are relatively homogeneous. For instance, the terms available in the GERG-2004 [19] and GERG-2008 [18] models can all be expressed in the form

$$F_k(\delta) = \tau^{t_k} \tag{51}$$

and

$$G_k(\delta) = \delta^{d_k} \exp[-\eta_k(\delta - \varepsilon_k)^2 - \beta_k(\delta - \gamma_k)], \tag{52}$$

and where the derivative $\delta G'_k(\delta)$ is given by

$$\delta G'_k(\delta) = G_k(\delta) \left[d_k - 2\delta\eta_k(\delta - \varepsilon_k) - \delta\beta_k \right]. \tag{53}$$

3.2 Degree versus error

The terms forming the pure fluid equations of state vary wildly in their complexity. Figure 3 demonstrates a few of the contributions from the nitrogen equation of state of Span et al. [30] formed from a set of 36 contributions. In this example, the entire domains of τ in [0.2, 4] and δ in $[1 \times 10^{-10}, 5]$ have been used (a single interval). The worst case error is determined over a few thousand data points linearly spaced over the domain. Figure 4 shows the estimation error of the Chebyshev expansions of each term as a function of the degree of the expansion. As expected, the Chebyshev expansions can yield high fidelity representations of each term given sufficient degree of the expansion. For the sharp Gaussian peak in τ in term 34 (F_{34}), even a 100^{th} degree Chebyshev expansion is not sufficient to yield a representation approximating numerical precision. For the other terms, there is a threshold, depending on the shape of the function fitted, beyond which further increasing the degree of expansion yields no further improvement in the fidelity of the expansion to the function it is representing.

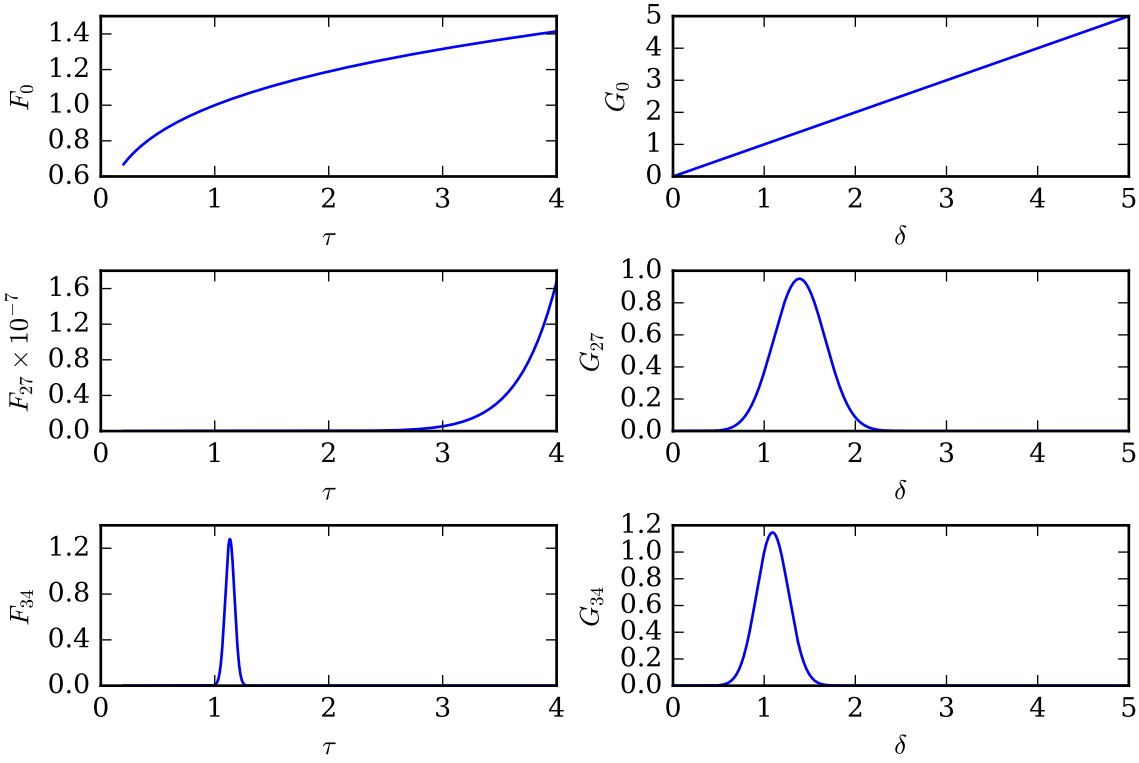

Figure 3: Functional values for selected terms (with indices 0, 27, and 34) from the reference equation of state of nitrogen [30]

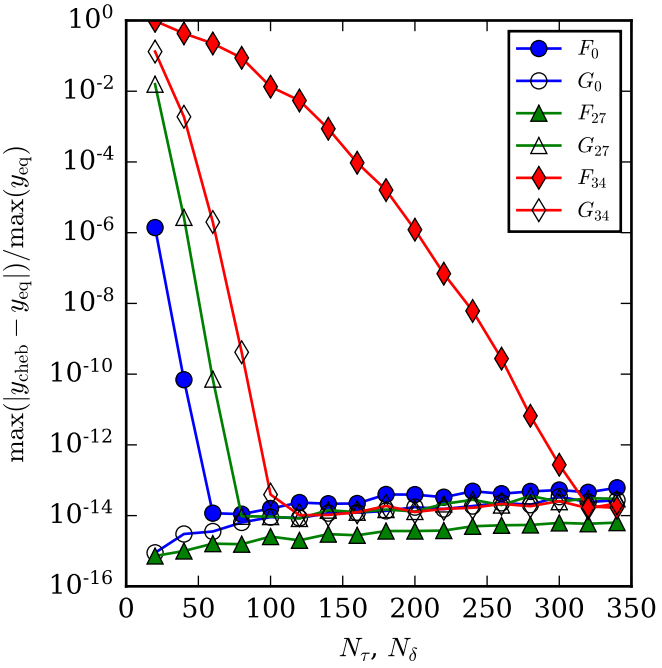

Figure 4: Expansion error as a function of expansion degree for selected terms from the reference equation of state of nitrogen [30]. The variable y_{cheb} is the value given by the Chebyshev expansion; y_{eq} is the value from the equation.

While ultimately numerical precision in the approximation of each of the terms with Chebyshev expansions can be approached, high degree expansions may be required to yield adequate error ("adequate" being problem dependent). From the standpoint of implementation, high degree expansions do not pose a major problem; the use of high degree expansions only results in more work for the computer so long as the code is structured appropriately. The next section demonstrates that a better way forward is to subdivide the domain of applicability of the Chebyshev expansion into multiple intervals, and in each one, a Chebyshev approximation is constructed.

3.3 Interval subdivision

As the previous section demonstrates, dramatically increasing the degree of the expansion will, asymptotically, yield a "perfect" representation of the function. The problem is that the eigenvalue solve required to do rootfinding takes on the order of N^3 operations, therefore, doubling the degree of the expansion yields a massive increase in computational time to find the eigenvalues. For a 10^{th} degree expansion, the eigenvalue rootfinding problem becomes the dominant contribution to the overall run time. Even if we use the iterative Chebyshev rootfinding algorithm, there is still a tradeoff between accuracy and speed.

Alternatively, the domain can be subdivided into inter-

vals, and in each interval, a Chebyshev expansion is built; the Chebyshev expansion in each interval is of much lower degree than would be required to yield the same accuracy with one interval. The optimal tradeoff between degree of expansion and number of intervals is a balance of a number of factors:

- For a larger number of intervals, more interval constructions must be carried out. While the construction of the set of F_k and G_k expansions can be carried out at initialization and cached internally, the computational effort required for creation of the combined expansions for the fluid and the mixture is non-trivial.

- For a smaller number of intervals, fewer interval constructions must be carried out, but at the price of increased degree of expansion in the intervals in order to retain the same level of accuracy.

- The higher the degree of expansion in a given interval, the larger the companion matrix becomes when the density solver is executed, resulting in a non-trivial amount of computational effort, as stated above.

- If the fidelity of the expansions to the equation of state is insufficient, it may prove impossible to yield sufficiently accurate roots of the equation, or all roots may not be found.

One approach, though certainly not the only one, that could be used to determine when to split an interval into two smaller intervals is to consider the magnitude of the coefficients of the Chebyshev expansion. The magnitude of the coefficients of the high order terms decay rapidly. Figure 5 demonstrates the decay of the magnitude of the coefficients for the Chebyshev expansion of $\exp(x)$ in the domain $[-1, 1]$. In this case, there is no additional improvement for degrees of expansion above approximately 15. Once a degree of expansion of 15 is reached, the relative magnitude of the high degree coefficients is at the level of double precision ($\approx 2.2 \times 10^{-16}$), and can be neglected.

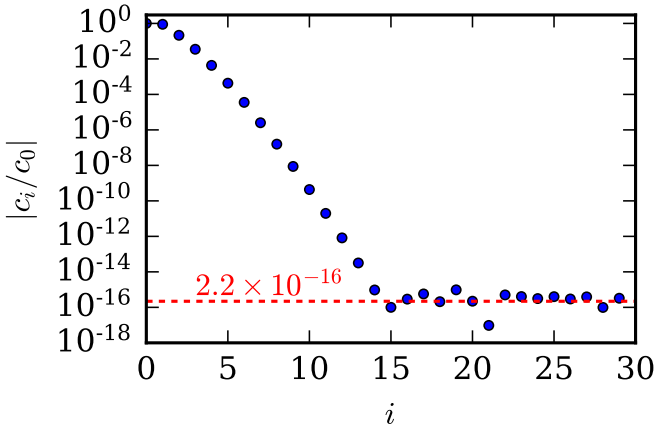

Figure 5: The relative magnitude of the coefficients of a Chebyshev expansion of $\exp(x)$ in the closed domain $[-1, 1]$.

In this work, we have decided to adopt the strategy of subdividing an interval if the norm of the highest degree M coefficients is greater than some fraction of the lowest degree M coefficients. As long as M is small relative to the degree of Chebyshev expansion, this strategy should yield a reasonable interval splitting metric. In this case, the M-term splitting metric is defined as

$$\varepsilon_{\text{split},M} = \frac{\sqrt{c_{N-M}^2 + \cdots + c_{N-1}^2 + c_N^2}}{\sqrt{c_0^2 + c_1^2 + \cdots + c_M^2}}. \tag{54}$$

Dyadic interval splitting is used; that is, the interval is split at its midpoint into two intervals of equal width.

Figure 6 demonstrates the results of the "automatic" splitting methodology, as compared with a naïve splitting methodology based upon linearly splitting the overall interval into a number of intervals, each of equal width. This figure is for an equimolar mixture of methane and hydrogen sulfide, a commonly considered mixture in mixture thermodynamics due to the wide range of fascinating thermodynamic behaviors it can demonstrate, even with quite simple thermodynamic models (e.g., for cubic equations of state). In this figure we define the worst M-element norm as the largest norm (defined by Eq. (54)) for any term from any of the equations of state in a given interval. The norms are considered based on the $\tilde{A}_{01}$ matrix for the mixture. This figure demonstrates that the automatically subdivided solution is more efficient than pure linear spacing; for 1000 intervals, the highest worst-case norm ratio is higher than the splitting tolerance. In the end, experience shows that in order to yield sufficient fidelity to the equation of state, it is necessary to use *quite a few* intervals of moderate degree expansions.

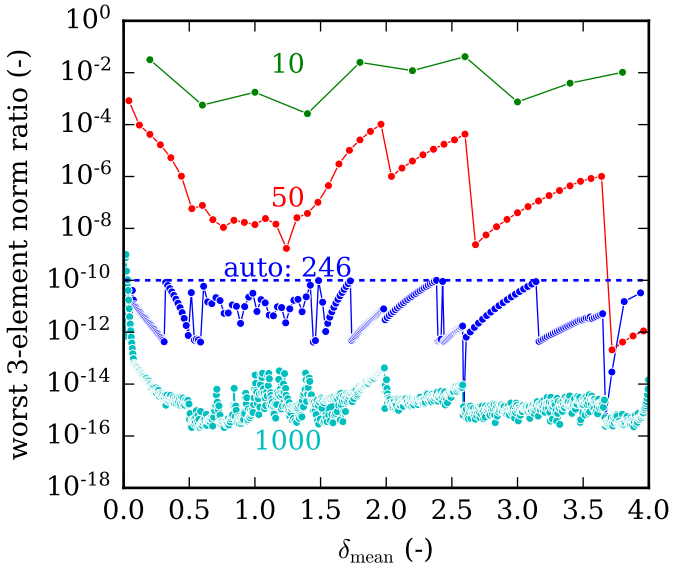

Figure 6: Worst 3-element norm ratio for $\tilde{A}_{01}$ in each interval with 10 degree expansions in δ versus mean δ in the interval for several linearly spaced intervals (with 10, 50, and 1000 intervals) as well as one automatically subdivided solution with a splitting tolerance of $\varepsilon_{\text{split},3} = 1\times10^{-10}$. The mixture is that of methane[26] + hydrogen sulfide [23]

.

4 Implementation

4.1 Architecture

An open-source C++11 library, `ChebTools`[4], was written by the authors in order to tackle the numerics involved in working with Chebyshev expansions of functions. The library `ChebTools` was integrated into the mixture rootfinding library developed in this work, here called `ChebMix`. The source code for `ChebMix` is available in the supplemental information. As with other open-source C++11 based projects, wrappers are developed for high-level languages like Python. The `pybind11` [15] library was used to develop a thin wrapper around the C++ objects. The architectural paradigm of a C++ core with high-level wrappers in Python and other high-level languages has been adopted by a number of open-source scientific packages (e.g., numpy, Psi4, CoolProp, etc.). The routines in `ChebMix` and `ChebTools` make heavy use of the header-only C++ library `Eigen` for carrying out the requisite linear algebra operations in C++.

4.2 Deconstruction algorithm

After having worked out the individual parts of the deconstruction, we arrive at the final algorithm for deconstructing a mixture model of a form that can be cast into the Helmholtz-energy-explicit formulation of the GERG[19, 18] model. Figure 7 graphically depicts how the deconstruction is carried out.

To begin, the coefficients of the departure terms and the pure fluid equations of state are read in; the terms are stored according to the javascript object notation (JSON) format of CoolProp [3]. After formatting the JSON data to a C++ data structure, the interval subdivision begins. The user must specify the bounds for the domain $[\tau_{\min}, \tau_{\max}] \times [\delta_{\min}, \delta_{\max}]$. For most equations of state, $\delta_{\max}$ can be set to a large value (often a value of 7 is reasonable, but it depends on the particular mixture model), and $\delta_{\min}$ should be set to a value very close to zero. For temperature, the limits on τ should be selected to match the limits of applicability of the equation of state (or the limits of reasonable extrapolation). Once the limits of the overall expansion have been selected, the expansions for each interval are constructed. Initially, a single interval is used for both δ and τ, and dyadic interval subdivision is used to split the interval into two equal parts. Further interval subdivisions follow the methods described in this work.

4.3 Chebyshev flattening

In order to obtain pressure from the overall expansion, the pressure Chebyshev expansion in terms of δ, for a given interval, is obtained by flattening the data structure from a set of expansions for each of the pure fluids and the departure terms into a single expression for the total mixture in terms of δ. The expansion of the pressure for the mixture, for a specified temperature and mixture composition, is obtained from

$$\tilde{p} = \rho_{\text{r}} RT \cdot \delta \left(1 + \tilde{A}_{01}\right). \tag{55}$$

In each step of the evaluation of Eq. (55), vectorized in-place operations can and should be used in order to minimize the amount of copying of array coefficients that is required. For more information on the implementation of this expansion, see the C++ code in the supplemental material; in particular, the `get_p` function.

There are two stages to the flattening: 1) For the specified value of τ, which is implicitly a function of the mixture composition via $\tau = T_{\text{r}}(x)/T$, flatten the τ dependence away into vectors containing the temperature dependence rather than matrices containing the temperature dependence; 2) For the given composition, flatten away the composition dependence as well, leaving a set of Chebyshev expansions for each interval.

There are many situations in the evaluation of mixture phase equilibrium problems in which the temperature and composition of the bulk phase is fixed, for instance in the calculation of bubble-point or dew-point calculations. In these cases, the temperature flattening needs only be carried out once for the bulk phase; the temperature flattening proves to be the most computationally expensive part of the evaluation of the density roots from Chebyshev

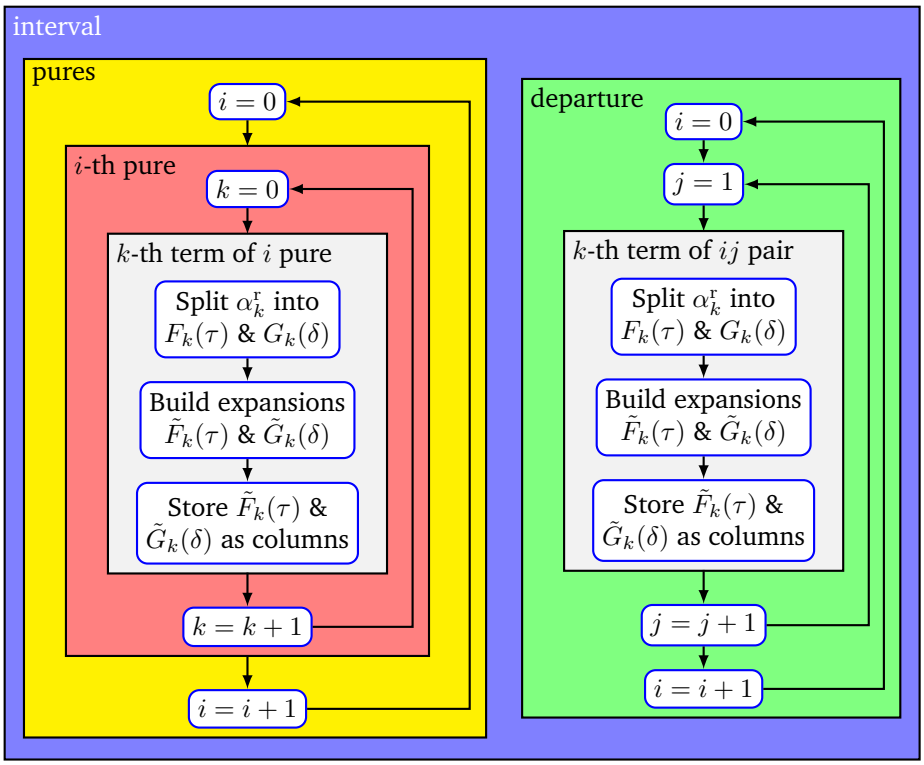

Figure 7: Flowchart for deconstruction of the mixture model into Chebyshev expansions for a given interval

.

expansions. In other cases (the general case), it is necessary to carry out all the flattening (both temperature and composition), which results in a rather significant computational penalty.

4.3.1 Corresponding state contributions

Flattening for the corresponding states contribution of $\tilde{A}_{01}$ is carried out by summing the expansions for the pure fluids $\tilde{A}_{01,i}$ times their mole fractions. This is motivated by the form of the underlying corresponding states contribution given by Eq. (37). The flattening operation for the corresponding states contribution can be expressed as the matrix-vector product

$$\tilde{A}_{01,\mathrm{C}} = \begin{bmatrix} | & \dots & | \\ \tilde{A}_{01,0} & \dots & \tilde{A}_{01,N_\mathrm{c}-1} \\ | & \dots & | \end{bmatrix} \begin{bmatrix} z_0 \\ \vdots \\ z_{N_\mathrm{c}-1} \end{bmatrix} \tag{56}$$

in which each column of the matrix is obtained from Eq. (47) (with coefficients in increasing degree going down the column), and the z_i are the molar fractions of the components 0 to $N_\mathrm{c} - 1$, and where N_c is the number of components.

4.3.2 Departure contributions

For the departure term, the flattening process is more complicated. The complexity arises from the fact that the product $z_i z_j$ multiplies each of the contributions in the double summation in Eq. (38). As a result, the coefficients of the α^{r}_{ij} terms must be stored carefully in the matrix. For instance, in the case of a three component mixture, the coefficients would be stored in the form

$$\tilde{A}_{01,\mathrm{D}} = \begin{bmatrix} | & | & | \\ \tilde{A}_{01,01} & \tilde{A}_{01,02} & \tilde{A}_{01,12} \\ | & | & | \end{bmatrix} \begin{bmatrix} z_0 z_1 F_{01} \\ z_0 z_2 F_{02} \\ z_1 z_2 F_{12} \end{bmatrix} \tag{57}$$

The point is that we can express the operations in terms of matrix-vector products; linear algebra libraries are optimized for this task.

The coefficients F_{ij} of the upper triangular matrix could be more efficiently stored in a vector that is appropriately indexed by the index l. The mapping between i, j and an index l that starts with the value of 0 at $0, 1$ and increases along the rows in the upper-triangular part of the i, j matrix can be given by:

$$l = iN_\mathrm{c} - i(i+1)/2 + j - i - 1. \tag{58}$$

For instance, the value of l can be used to index into the vector of F for a given i, j pair. To make this mapping more concrete, the indices l for the ij entries of a three-

component mixture are given by

$$\begin{aligned} ij &\to l \\ 01 &\to 0 \\ 02 &\to 1 \\ 12 &\to 2 \end{aligned}$$

4.4 Density solver

Once the overall Chebyshev expansion has been flattened to a single expansion of the form $\tilde{p} = f(\delta)$, it is then possible to determine all the real values of δ in the domain $[\delta_{\min}, \delta_{\max}]$ that satisfy $\tilde{p}(\delta) = p_{\text{target}}$ by creating a new residue expansion of the form $\tilde{r}(\delta) = \tilde{p}(\delta) - p_{\text{target}}$ and determining all values of δ that satisfy $\tilde{r}(\delta) = 0$. The roots of $\tilde{r}(\delta)$ are obtained according to the method proposed in Section 2.4 based on the secant solver of the Chebyshev expansion; the rootfinding avoidance heuristic proposed in the same section allows for rootfinding to be avoided in nearly all of the intervals, dramatically improving the speed of the rootfinding. For instance, if there are 300 intervals, there may be only 4 or 5 intervals in which roots *might* be found. In the candidate intervals, the rootfinding is carried out, and all real roots are collected.

4.4.1 Solution polishing

As will be shown below, although the roots obtained by the density solver are usually obtained exceptionally accurately, it can be advantageous to yield even more precise density roots by polishing the obtained density roots. The full equation of state is used to polish each of the obtained roots, starting at each of the density roots obtained from the Chebyshev expansion.

Newton's method of rootfinding can be given as the root updating formula

$$\delta_{\text{new}} = \delta_{\text{old}} - \frac{r}{r'} \tag{59}$$

where r is given by

$$r = p - p_{\text{target}}, \tag{60}$$

where r' is given by

$$r' = \left(\frac{\partial p}{\partial \delta}\right)_T = \rho_{\text{r}} RT \left[1 + 2A_{01} + A_{02}\right], \tag{61}$$

and the concise derivatives are as given in Eq. (46) (but for the full mixture model, not the Chebyshev expansion).

Halley's method is an extension of Newton's method of convergence order 3:

$$\delta_{\text{new}} = \delta_{\text{old}} - \frac{2rr'}{2(r')^2 - rr''} \tag{62}$$

where

$$r'' = \left(\frac{\partial^2 p}{\partial \delta^2}\right)_T = \frac{\rho_{\text{r}} RT}{\delta} \left[2A_{01} + 4A_{02} + A_{03}\right] \tag{63}$$

In general, one or two steps of Halley's method, applied to the entire EoS, is sufficient to yield density roots consistent with the equation of state for which the error in pressure calculated from the equation of state with the polished root is less than $10^{-10}\%$. The classical rootfinding algorithms usually have a threshold on the same order. There are cases, however, where the polishing may not be completely successful. For instance, the unstable two-phase density roots near $\delta = 1$ may sometimes have a derivative r' that is exceptionally large in magnitude (see for instance Fig. 1). As a result, it may be simply impossible to yield a further polishing of the root because $\left|\frac{r}{r'}\right|$ goes to zero as r' becomes very large in magnitude.

4.5 Numerical considerations

There are a number of numerical challenges that must be overcome when working with Chebyshev rootfinding for multi-fluid mixture models. One of the most serious numerical problems is the issue of the balancing of the companion matrix when the density solve is based on the eigenvalues. The core problem is that the companion matrices are often very ill-conditioned, and as a result, if the obtained companion matrix is not balanced, the eigenvalues of the companion matrix (the density roots are the real eigenvalues) can be significantly in error. In this work we have adopted the matrix balancing of James et al. [16, Algorithm #3]. This approach proves to be quite effective at balancing the companion matrices and yields reliable eigenvalues. Without it, the eigenvalues calulated by `Eigen` were at times significantly in error.

While it is possible to develop expansions of F_k and G_k directly and then take derivatives of the expansion of G_k with respect to δ, it is numerically preferable to develop the expansion of the derivative directly, as is described in this work. Otherwise, differentiation of the expansion can cause a reduction in fidelity to the underlying model, as is highlighted in Section 2.5.3.

In this work, the eigenvalue solver employed was the baseline eigenvalue solver of the `Eigen` library. While in principle, some modest increase in computational speed should be possible through the use of Schur decompositions (because the companion matrices are all Hessenberg, and the Schur decomposition exposes the eigenvalues along its block diagonal), the Schur decomposition did not, in practice, yield significant improvements in the overall speed. Several implementations of the Schur decomposition were compared. It was decided to consistently use the eigenvalue solver of `Eigen` with the appropriate balancing, which proved to be a satisfactory solution. It is still hoped that a faster eigenvalue solver might be possible, but that seems unlikely at this juncture.

5 Results

The results presented here are obtained with the following system configuration:

- 64-bit Windows 7 PC with 16 GB of RAM and Intel Core i7-3770 CPU @ 3.4GHz
- Python 3.6
- NIST REFPROP version 10.0 [21]
- EoS data from CoolProp version 6.1[3].

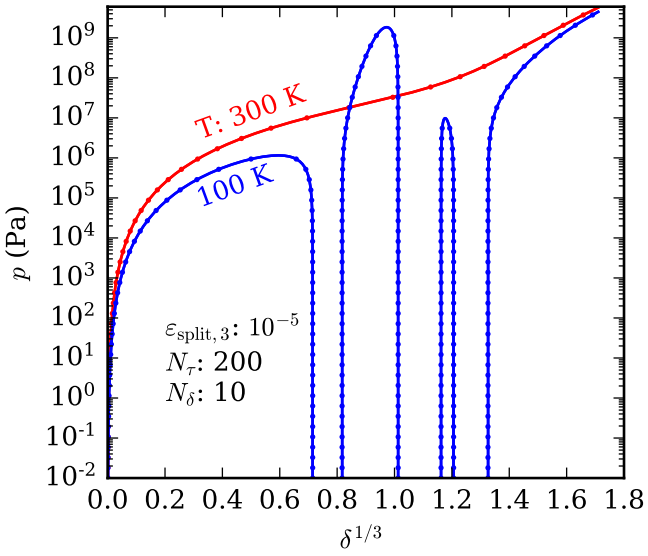

Figure 8: Isotherms at 100 K and 300 K for an equimolar mixture of nitrogen[30] + argon [31] with the mixing parameters from Gernert *et al.* [11]. Solid curves correspond to the mixture model, and markers correspond to the densities obtained from the Chebyshev rootfinding for specified values of temperature and pressure. There were no failures of the density rootfinder.

As a demonstration of the potential challenges with the state-of-the-art algorithms for density rootfinding, we present two isotherms for the mixture of nitrogen + argon from the GERG 2008 model. For the 100 K isotherm K at low pressures (e.g., at 100 Pa), *seven* density roots can be obtained for a given pressure, and at the higher temperature of 300 K, only one density root is obtained because the mixture is supercritical. Along the 100 K isotherm, it can be very difficult to find the liquid-like density solution by conventional means. The first step of the conventional rootfinder is to find the pressure extrema, in this case, where $(\partial p/\partial \rho)_T = 0$. Calculation of the pressure extrema along an isotherm is non-trivial because of the steepness of the pressure along an isotherm. It is quite easy to begin with a guess for the location of the pressure minimum that results in the extrema rootfinder jumping to another minimum than desired because the pressure extrema are very tightly packed. It is not easy to start off at a safe density value that can guarantee that the extrema search will succeed. A very small change in density (or δ) corresponds to an enormous change in pressure along the near-vertical parts of the isotherm. The Chebyshev rootfinder does not suffer from this limitation, and can be considered effectively as a direct rootfinding method (though there are very stable iterations involved in solving for the roots of the expansions, whether via eigenvalue solving or modified secant method).

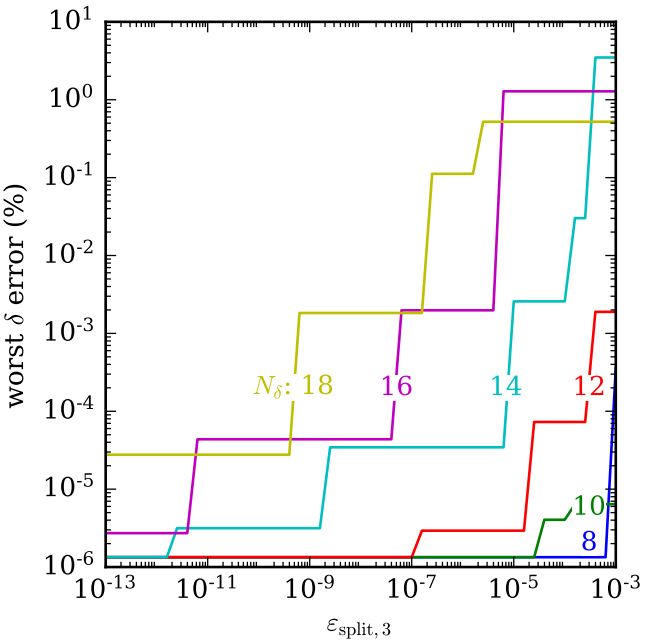

Figure 9: Accuracy of the density roots obtained from the expansion as a function of the degree of the expansion in δ in each interval N_δ as well as the splitting tolerance $\varepsilon_{\text{split},3}$. Results are for a mixture of methane[26] + ethane[6] with the departure function from GERG 2008 [18] with a 200 degree (201 term) expansion in τ.

Figure 9 provides a summary of the tradeoff between the interval degree in δ and the splitting tolerance $\varepsilon_{\text{split},3}$. In this figure, the worst-case relative error in δ is presented over a wide range of temperature, pressure and composition for a mixture of methane and ethane. The worst-case error is the largest relative error between the density obtained from the Chebyshev expansion and the density obtained after polishing the density root with the full equation of state as described above. This figure demonstrates that for a given interval degree in δ, as the splitting tolerance tightens, the worst-case error decreases. The asymptotic limit for exceptionally tight values of δ appears to be a worst-case density error for this mixture of approximately 1.4×10^{-6}%. The utility of this figure is to identify the necessary splitting tolerance for a specified level of accuracy of the density roots. For instance, calculation of density roots that are consistent with the full equation of state to within one part per mil-

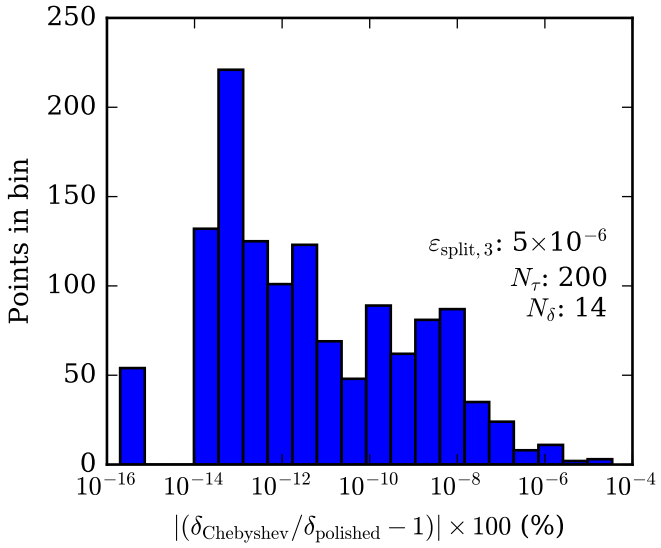

Figure 10: Error in density from the Chebyshev rootfinding problem. Values of 0% error (correct density root to numerical precision) were entered as 2×10^{-16}. The mixture is that of methane[26] + ethane[6], τ values in [0.25, 2.25], p in [0.1 Pa, 10^9 Pa], z_0 in [0, 1].

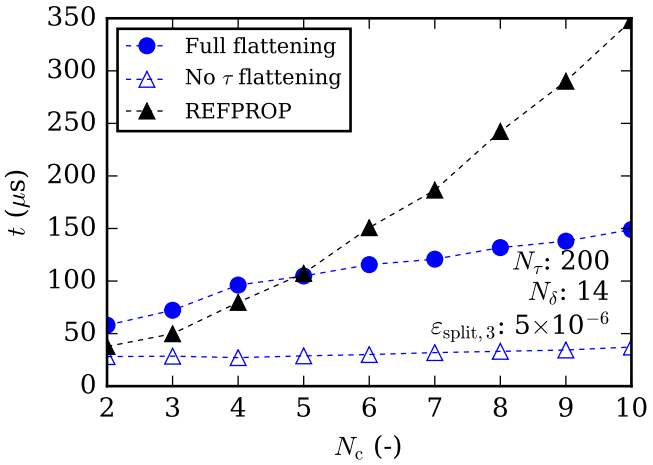

Figure 11: Timing of density calculations with REFPROP and the Chebyshev rootfinding for multicomponent n-alkane mixtures at a temperature of 200 K and a pressure of 1 MPa. Lines are included to guide the eye.

lion requires that the worst-case error be less than 10^{-4}%. Therefore, drawing a horizontal line through the figure, we can determine the smallest value of the splitting tolerance that will yield all density roots to within the specified level of accuracy. The more strict the requirements on the error in the density roots, the tighter the splitting tolerance that would be required, and therefore, the more computational work that would be expended.

Figure 10 shows the distribution of the error of the density roots obtained from the Chebyshev expansion for a mixture of methane + ethane with τ, composition, and p values exhaustively spanning the ranges of validity. The error is given as the comparison between the values obtained from polishing the density root with the full equation of state and the value obtained from the Chebyshev rootfinding problem without any additional polishing. In this case, the interval subdivision is quite conservative, and the error in density has a peak in the distribution around an error of 10^{-12}%. The worst case values are errors in density of approximately 10^{-4}%. These results demonstrate that the density roots obtained by this method are exceptionally reliable, and that for all inputs investigated, the returned densities were consistent with the equation of state to within a part per million.

From the standpoint of performance, the ultimate comparison is against the computational throughput of the full equation of state. Figure 11 shows a comparison of the computational efficiency of density rootfinding with the Chebyshev rootfinding as compared with the density rootfinder of REFPROP, in particular the `TPRHOdll` function in the liquid phase at 200 K and 1000 kPa without guess values. The fluids considered are the linear alkanes methane[26], ethane[6], propane[22], butane[7], pentane[28], hexane[28], heptane[28], octane[28], nonane[23], decane[23], etc.. Each mixture is formed of an equimolar mixture of the first N_c linear alkanes; for $N_c = 4$ the mixture would be an equimolar mixture of methane, ethane, propane, and butane. The expansion parameters were selected to yield a worst-case density error of less than one part per million according to Fig. 9. This figure indicates that the computational efficiency breakeven point occurs for a mixture with approximately five components. For mixtures with fewer components, the Chebyshev rootfinding routines are somewhat slower, but considering the guarantees that can be made about the accuracy of the density roots obtained by the Chebyshev rootfinding method, that computational penalty may be be worth the cost, depending on the application. The comparison here should only be considered as a rule-of-thumb; many other considerations must be handled in a more rigorous fashion (e.g., calling overhead, precise values obtained, root polishing, etc.) to ensure a fair comparison. While the Chebyshev rootfinding is slower than the "full" density solver for mixtures with a few components, it is not orders of magnitude slower. The penalty is quite modest and all density roots are obtained at the same time, which is not true for the conventional density rootfinder. When the value of τ is fixed, the temperature flattening can be avoided. In that case, the density rootfinder is significantly faster than REFPROP for all mixtures.

6 Conclusions

The method developed in this work is a novel approach to the rootfinding problem with multiparameter equations of state. It permits a nearly-guaranteed solution for *all* the density roots ρ that satisfy $p(T, \rho) = p_{\text{target}}$. This method should allow for a radical improvement in the reliability and stability of phase equilibrium calculations; the density solver is one of the weak links in the algorithms required for mixture thermodynamics. Additionally, this approach can be used to make other mixture calculations much more reliable when one of temperature or density are known.

No numerical algorithm is entirely without flaws. There are a few limitations to this method of density rootfinding:

- It is not currently possible to model terms that cannot be separated into independent contributions that are uniquely functions of either τ or δ. Examples of this category would be the non-analytic[2] terms used for water[34] and carbon dioxide[27].
- Conservative limits for τ and δ in the Chebyshev expansions must be specified as an input to the mixture model deconstruction. Therefore, the user must have some idea about the limits of physically-reasonable extrapolation of the pure-fluid equations of state and the mixture model. In principle this knowledge can be gleaned from the metadata associated with the equation of state, but there are subtle features of the mixture model that may require more conservative bounds than expected: *caveat emptor*.
- For mixtures with very few components, this method is not as fast as the conventional density rootfinding methods, although it does avoid a number of the common pitfalls associated with some of the state-of-the-art rootfinding methods.
- Once *all* the roots have been found, it is still necessary to decide what to do with the roots. Are the density roots stable (in a thermodynamic sense)? Should they be rejected due to other thermodynamic considerations?

In spite of these manageable issues, the method presented in this work seems well-poised to result in major improvements in the reliability of calculations with multi-fluid mixture models.

Acknowledgments

The authors thank Lucas Bouck of George Mason University for many fruitful mathematical discussions and assistance with some derivations.

[2] in the sense that their value is undefined at the critical point

References

[1] Ian H. Bell and Andreas Jäger. Helmholtz energy transformations of common cubic equations of state for use with pure fluids and mixtures. *J. Res. NIST*, 2016. doi: 10.6028/jres.121.011.

[2] Ian H. Bell and Andreas Jäger. Calculation of critical points from Helmholtz-energy-explicit mixture models. *Fluid Phase Equilib.*, 433:159 – 173, 2017. ISSN 0378-3812. doi: 10.1016/j.fluid.2016.10.030.

[3] Ian H. Bell, Jorrit Wronski, Sylvain Quoilin, and Vincent Lemort. Pure and Pseudo-pure Fluid Thermophysical Property Evaluation and the Open-Source Thermophysical Property Library CoolProp. *Ind. Eng. Chem. Res.*, 53(6):2498–2508, 2014. doi: 10.1021/ie4033999.

[4] Ian H. Bell, Lucas Bouck, and Bradley K. Alpert. ChebTools: C++11 (and Python) tools for working with Chebyshev expansions. *Journal of Open Source Software*, 2018. doi: 10.21105/joss.00569.

[5] John P. Boyd. Finding the Zeros of a Univariate Equation: Proxy Rootfinders, Chebyshev Interpolation, and the Companion Matrix. *SIAM Review*, 55 (2):375–396, 2013. doi: 10.1137/110838297.

[6] D. Buecker and W. Wagner. A Reference Equation of State for the Thermodynamic Properties of Ethane for Temperatures from the Melting Line to 675 K and Pressures up to 900 MPa. *J. Phys. Chem. Ref. Data*, 35(1):205–266, 2006. doi: 10.1063/1.1859286.

[7] D. Buecker and W. Wagner. Reference Equations of State for the Thermodynamic Properties of Fluid Phase n-Butane and Isobutane. *J. Phys. Chem. Ref. Data*, 35(2):929–1019, 2006. doi: 10.1063/1.1901687.

[8] DLMF. *NIST Digital Library of Mathematical Functions*. http://dlmf.nist.gov/, Release 1.0.14 of 2016-12-21, 2016. URL `http://dlmf.nist.gov/`. F. W. J. Olver, A. B. Olde Daalhuis, D. W. Lozier, B. I. Schneider, R. F. Boisvert, C. W. Clark, B. R. Miller and B. V. Saunders, eds.

[9] M. Dowell and P. Jarratt. The "Pegasus" method for computing the root of an equation. *BIT*, 12(4): 503–508, 1972. doi: 10.1007/bf01932959.

[10] T. A. Driscoll, N. Hale, and L. N. Trefethen. Chebfun Guide, 2014.

[11] Johannes Gernert and Roland Span. EOS-CG: A Helmholtz energy mixture model for humid gases and CCS mixtures. *J. Chem. Thermodyn.*, 93:274–293, 2016. doi: 10.1016/j.jct.2015.05.015.

[12] Johannes Gernert, Andreas Jäger, and Roland Span. Calculation of phase equilibria for multi-component

mixtures using highly accurate Helmholtz energy equations of state. *Fluid Phase Equilib.*, 375:209–218, 2014. doi: 10.1016/j.fluid.2014.05.012.

[13] Amparo Gil, Javier Segura, and Nico M. Temme. *Numerical Methods for Special Functions*. Society for Industrial and Applied Mathematics, 2007.

[14] Gene H. Golub and Charles F. van Loan. *Matrix Computations, Fourth Edition*. The John Hopkins University Press, 2013. ISBN 978-1-4214-0794-4.

[15] Wenzel Jakob, Jason Rhinelander, and Dean Moldovan. pybind11 – seamless operability between c++11 and python, 2016. https://github.com/pybind/pybind11.

[16] Rodney James, Julien Langou, and Bradley R. Lowery. On matrix balancing and eigenvector computation. *arXiv*, 2014. URL `https://arxiv.org/abs/1401.5766`.

[17] Richard F. King. An improved pegasus method for root finding. *BIT*, 13(4):423–427, 1973. doi: 10.1007/bf01933405.

[18] O. Kunz and W. Wagner. The GERG-2008 Wide-Range Equation of State for Natural Gases and Other Mixtures: An Expansion of GERG-2004. *J. Chem. Eng. Data*, 57:3032–3091, 2012. doi: 10.1021/je300655b.

[19] O. Kunz, R. Klimeck, W. Wagner, and M. Jaeschke. *The GERG-2004 Wide-Range Equation of State for Natural Gases and Other Mixtures*. VDI Verlag GmbH, 2007.

[20] E. W. Lemmon, M. L. Huber, and M. O. McLinden. NIST Standard Reference Database 23: Reference Fluid Thermodynamic and Transport Properties-REFPROP, Version 9.1, National Institute of Standards and Technology, 2013. URL `http://www.nist.gov/srd/nist23.cfm`.

[21] E. W. Lemmon, Ian H. Bell, M. L. Huber, and M. O. McLinden. NIST Standard Reference Database 23: Reference Fluid Thermodynamic and Transport Properties-REFPROP, Version 10.0, National Institute of Standards and Technology, 2017. URL `http://www.nist.gov/srd/nist23.cfm`.

[22] Eric W. Lemmon, Mark O. McLinden, and Wolfgang Wagner. Thermodynamic Properties of Propane. III. A Reference Equation of State for Temperatures from the Melting Line to 650 K and Pressures up to 1000 MPa. *J. Chem. Eng. Data*, 54:3141–3180, 2009. doi: 10.1021/je900217v.

[23] E.W. Lemmon and R. Span. Short Fundamental Equations of State for 20 Industrial Fluids. *J. Chem. Eng. Data*, 51:785–850, 2006. doi: 10.1021/je050186n.

[24] John C. Mason and David Christoper Handscomb. *Chebyshev Polynomials*. Chapman & Hall, 2003.

[25] Michael L. Michelsen and Jørgen M. Mollerup. *Thermodynamic Models: Fundamentals & Computational Aspects*. Tie-Line Publications, 2007.

[26] U. Setzmann and W. Wagner. A New Equation of State and Tables of Thermodynamic Properties for Methane Covering the Range from the Melting Line to 625 K at Pressures up to 1000 MPa. *J. Phys. Chem. Ref. Data*, 20(6):1061–1151, 1991. doi: 10.1063/1.555898.

[27] R. Span and W. Wagner. A New Equation of State for Carbon Dioxide Covering the Fluid Region from the Triple Point Temperature to 1100 K at Pressures up to 800 MPa. *J. Phys. Chem. Ref. Data*, 25:1509–1596, 1996. doi: 10.1063/1.555991.

[28] R. Span and W. Wagner. Equations of State for Technical Applications. II. Results for Nonpolar Fluids. *Int. J. Thermophys.*, 24:41–109, 2003. doi: 10.1023/A:1022310214958.

[29] R. Span, T. Eckermann, S. Herrig, S. Hielscher, A. Jäger, and M. Thol. TREND. Thermodynamic Reference and Engineering Data 3.0, 2016. URL `http://www.thermo.ruhr-uni-bochum.de/`.

[30] Roland Span, Eric W. Lemmon, Richard T. Jacobsen, Wolfgang Wagner, and Akimichi Yokozeki. A Reference Equation of State for the Thermodynamic Properties of Nitrogen for Temperatures from 63.151 to 1000 K and Pressures to 2200 MPa. *J. Phys. Chem. Ref. Data*, 29:1361–1433, 2000. doi: 10.1063/1.1349047.

[31] Ch. Tegeler, R. Span, and W. Wagner. A New Equation of State for Argon Covering the Fluid Region for Temperatures From the Melting Line to 700 K at Pressures up to 1000 MPa. *J. Phys. Chem. Ref. Data*, 28:779–850, 1999. doi: 10.1063/1.556037.

[32] José O. Valderramma. The State of the Cubic Equations of State. *Ind. Eng. Chem. Res.*, 42:1603–1618, 2003. doi: 10.1021/ie020447b.

[33] Johannes Diderik van der Waals. *Over de Continuiteit van den Gas- en Vloeistoftoestand*. PhD thesis, University of Leiden, 1873.

[34] W. Wagner and A. Pruβ. The IAPWS Formulation 1995 for the Thermodynamic Properties of Ordinary Water Substance for General and Scientific Use. *J. Phys. Chem. Ref. Data*, 31:387–535, 2002. doi: 10.1063/1.1461829.