Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 369 (2020) 113207
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

A self-adaptive deep learning algorithm for accelerating multi-component flash calculation☆

Tao Zhang[a], Yu Li[b], Yiteng Li[a], Shuyu Sun[a,*], Xin Gao[b,*]

[a] *Computational Transport Phenomena Laboratory, Division of Physical Science and Engineering, King Abdullah University of Science and Technology, Thuwal 23955-6900, Saudi Arabia*

[b] *King Abdullah University of Science and Technology (KAUST), Computational Bioscience Research Center (CBRC), Computer, Electrical and Mathematical Sciences and Engineering (CEMSE) Division, Saudi Arabia*

Available online 11 June 2020

Abstract

In this paper, the first self-adaptive deep learning algorithm is proposed in details to accelerate flash calculations, which can quantitatively predict the total number of phases in the mixture and related thermodynamic properties at equilibrium for realistic reservoir fluids with a large number of components under various environmental conditions. A thermodynamically consistent scheme for phase equilibrium calculation is adopted and implemented at specified moles, volume and temperature, and the flash results are used as the ground truth for training and testing the deep neural network. The critical properties of each component are considered as the input features of the neural network and the final output is the total number of phases at equilibrium and the molar compositions in each phase. Two network structures are well designed, one of which transforms the input of various numbers of components in the training and the objective fluid mixture into a unified space before entering the productive neural network. "Ghost components" are defined and introduced to process the data padding work in order to modify the dimension of input flash calculation data to meet the training and testing requirements of the target fluid mixture. Hyperparameters on both two neural networks are carefully tuned in order to ensure the physical correlations underneath the input parameters are preserved properly through the learning process. This combined structure can make our deep learning algorithm to be self-adaptive to the change of input components and dimensions. Furthermore, two Softmax functions are used in the last layer to enforce the constraint that the summation of mole fractions in each phase is equal to 1. An example is presented that the flash calculation results of a 8-component Eagle Ford oil is used as input to estimate the phase equilibrium state of a 14-component Eagle Ford oil. The results are satisfactory with very small estimation errors. The capability of the proposed deep learning algorithm is also verified that simultaneously completes phase stability test and phase splitting calculation. Remarks are concluded at the end to provide some guidance for further research in this direction, especially the potential application of newly developed neural network models.

☆ The work of Tao Zhang, Yiteng Li and Shuyu Sun was supported by funding from the NSF of China (Grants No. 51874262 and 51904031) and King Abdullah University of Science and Technology (KAUST), Saudi Arabia through the grants BAS/1/1351-01-01. The work of Yu Li and Xin Gao was supported by funding from King Abdullah University of Science and Technology, Saudi Arabia (KAUST) through the grants BAS/1/1624-01-01 and URF/1/4077-01-01.

* Corresponding authors.
E-mail addresses: tao.zhang.1@kaust.edu.sa (T. Zhang), yu.li@kaust.edu.sa (Y. Li), yiteng.li@kaust.edu.sa (Y. Li), shuyu.sun@kaust.edu.sa (S. Sun), xin.gao@kaust.edu.sa (X. Gao).

https://doi.org/10.1016/j.cma.2020.113207

MSC: 68T05; 82B30

Keywords: Deep learning algorithm; Flash calculation; Self-adaptive neural network; Multi-component reservoir fluid

1. Introduction

Multiphase multicomponent flows in geological formation are often needed to be solved numerically to understand complicated flow behaviors, since the subsurface oil and gas typically contain numerous chemical species and exhibit complex phase behaviors [1]. Flash calculation, a computational tool to predict phase partitioning of a given fluid mixture, is a key step in multiphase multicomponent flow simulation as the thermodynamic properties of the investigated mixture, such as composition, densities and the number of phases, play a critical role in accurate description of reservoir fluid transport and migration [2]. Nowadays, phase behavior modeling is always conducted based on realistic equation of states (EOS), such as Peng–Robinson (PR) EOS [3] and Soave–Redlich–Kwong (SRK) EOS [4] which are commonly used in reservoir simulators. Iterative algorithms are popular in solving the nonlinear phase equilibrium problems, but phase stability test and phase split calculations are computationally expensive. A hydrocarbon mixture can comprise up to hundreds of species in reality, which could consume enormous computational cost for phase equilibria modeling by iterative flash schemes. Thus, the petroleum industry has been increasingly seeking for a reliable and efficient flash calculation method to handle the realistic reservoir fluid mixture containing a large number of components.

A simple and direct acceleration strategy is to reduce the number of components in the fluid mixture and meanwhile preserve as much as accuracy of the EOS model [5]. However, the balance between the remaining number of components and the computational accuracy often yields unsatisfactory acceleration performance, and thus a large batch of scholars have been working on various flash calculation acceleration approaches. The first reduction model was proposed in [6] to decrease the number of nonlinear equations by taking advantage of the low-rank binary interaction coefficient (BIC) matrix. With the van der Waals mixing rule, only three equations need to be solved. However, the zero binary interaction coefficient assumption is not suitable for realistic reservoir fluids. Another reduction model was proposed in [7,8] based on the spectral theory of linear algebra to accelerate the phase stability tests as well as phase equilibrium calculations. A smoother tangent plane distance (TPD) function can be obtained in the reduced space and robustness and efficiency of their reduction model is verified through numerical tests. In addition, efforts have also been paid on decomposing the BICs, and a generalized formulation was proposed in [9]. At most six reduced parameters are included in their scheme, which could be degenerated to the reduced model in [6] with three parameters or the model in [10] with five parameters considering the non-zero BICs between a single non-hydrocarbon component and hydrocarbon components. A significant contribution in this scheme stands on retaining the calculation accuracy at the same time of avoiding solving the nonlinear Rachford–Rice equation. Besides, the energy parameter was approximated with a minimized error in [11] with a new decomposition method and fewer reduced variables are needed in this approach. However, the efficiency of these acceleration methods with reduced models has not been agreed and recognized widely for general cases.

Another class of solutions is to avoid the iterative calculations in order to reduce the computational efforts. A non-iterative algorithm was developed in [12], assuming an instantaneously established thermodynamic equilibrium throughout the whole simulation process and the decoupled flash calculation from compositional flow simulation can be approximated based on the previous and current flow solution. The difficulty to solve Rachford–Rice equation is also relieved in this scheme, which can further accelerate the flash calculations. Repeated phase stability tests were reduced in [13] by using a shadow region method. It was assumed that temperature, pressure or compositions have small changes in two consecutive time steps located in a certain grid block, and the determination of whether stability tests are needed or not can be made by checking the distance to the critical point. In addition, the sparse grid surrogate model was introduced in [14] to approximate phase equilibrium properties by a interpolation approach and a subsequent study further improved the approximation efficiency by constructing the layer and array structure. However, the detection of single-phase region is not always accurate and reliable, while the saturation can even exceed the physical meaningful boundaries.

During the last two decades, a large number of acceleration strategies for flash calculations sprang out, but the wide recognition, high reliability and general application are still not resolved well. Meanwhile, the fast development of reservoir simulation is calling for phase equilibrium calculation at extremely high resolutions,

even up to millions of grid blocks. Conventional acceleration approaches cannot handle this task and attentions have been transferred to machine learning techniques as a speedup tool. The use of artificial neural networks (ANN) was reported in [15], successfully estimating the density of refrigerants and shape factors, which can be further extended to model the corresponding thermodynamic state. The breakthrough of AlexNet in 2012 [16] announced the start of deep learning era. This technique has revolutionized various industry and academia fields, including computer vision [17], natural language translation and processing [18], electronic entertainments [19] and many others [20–22]. Recently, numerous efforts have been made to apply this promising technique to related researches in the oil and gas industry. A deep neural network was constructed in [23] to learn the mapping between stochastic fields and effective properties for fast calculation of the effective properties in subsurface reservoirs for a coarse grid approximation of the problem and a more remarkable progress was reported in [24] using convolutional neural network (CNN) for high-dimensional problems. A Multilayer Neural Network (ML-NN) artificial intelligent technique was utilized in [25] in a robust Equation-of-State reservoir simulator fully coupled with geochemistry to mechanistically model Hybrid Low Salinity Chemical Flooding with the capability of uncertainty assessment of the LSW process at the field scale. A regression model was developed in [26] to predict the equilibrium coefficients using ANN, and both relevance vector machines (RVM) and ANN were applied in [27] to reduce the cost of phase-equilibrium calculations for compositional models. A significant improvement can be detected on deep learning algorithms compared with traditional machine learning methods, mainly attributed to the multiple hidden non-linear layers that enable the model to represent the complex correlations among the input features. It has been pointed out in [28] that any continuous function can be approximated by a feed-forward network consisting of a finite number of neurons on one single hidden layer, which is also known as the universal approximation theorem, but the success of deep learning has shown a more promising potential to predict the thermodynamic equilibrium properties using multi-layer deep neural networks. It can be stated that the task environment in general flash calculation is static, so that a trained model is capable to capture the distribution at a specific time point. This property is intrinsic in the underlying physical process, which makes the deep learning model to be practically useful. Besides, there is a large amount of data in the flash calculation field, from both physical experiments and computational simulations (easily up to billions). Such a huge amount of data provides expected opportunities for training deep learning models. However, despite the suitability and necessity of developing deep learning-based methods in this field and their potential applications, limited attempts have been made on developing deep learning methods for accelerating flash calculation. A fully connected deep neural network was designed and tuned carefully in [29], in which the critical thermodynamic properties are selected as input features to represent each component implicitly. Instead of using conventional EOS, the correlation rules among the input features are described by the trained model, and a highly enhanced efficiency can be obtained with acceptable accuracy. In order to overcome the limited number of expensive experimental data as training ground truth, the flash calculation results at given moles, volume and temperature (NVT flash) are used as the ground truth for training and testing, so that the deep learning algorithm in [29] has been extended from binary component cases to multi-component cases and a larger range of environmental conditions can be considered [30]. A significant progress has been reported in [31,32] that the trained deep learning model to accelerate both phase stability test and phase splitting calculations was successfully integrated in practical compositional reservoir simulators. The steps have forwarded closer to realistic reservoir complex fluid, but all of these existing deep learning methods assume a fixed and given number of components in the mixture, which makes such models to be practically useless in other engineering cases with various numbers of components in the fluid mixture.

In this paper, an optimized deep learning algorithm is proposed based on a two-network structure as the first trial to overcome the previous limitations on the number of components of the target fluid mixture. A thermodynamically consistent NVT flash calculation scheme is generated first to prepare the data for training and testing. Two fluid mixtures, an 8-component Eagle Ford oil and a 14-component Eagle Ford oil, are selected for flash calculation as the input training data and output test data, respectively. Several practical techniques are involved in our deep learning algorithm, including Xavier initializer, dropout, batch normalization, to overcome overfitting problem so as to improve the prediction accuracy, and the Softmax function is applied to meet the constraint of the summed mole fractions of each phase equal to 1. Features on the network structure is tuned carefully to chase for a better performance, and the reliable estimation is an encouraging example to seize the possibility of using the reservoir fluid data of a small number of components to predict the equilibrium phase compositions of complex reservoir fluids with a larger number of components. This satisfactory robustness and accuracy can promise a good potential in this direction to finally investigate realistic reservoir fluids by acceptable computation causes.

The rest of the paper is organized as follows. In Section 2, a thermodynamically consistent phase equilibrium calculation scheme is generated at specified moles, volume and temperature as constraints. The flash calculation results will be used as the ground truth for training and testing in the deep neural network. In Section 3, a two-network structure is designed and related techniques are employed to improve the estimation accuracy and efficiency, including Xavier initializer, dropout, batch normalization and Softmax. In Section 4, An example of using an 8-component Eagle Ford oil flash calculation results as training inputs to predict the phase equilibrium properties of a 14-component Eagle Ford fluid mixture. Network tuning are conducted for better performance and phase stability tests and splitting calculations are incorporated together. At the end, we provide a conclusion and certain remarks are provided in Section 5 to suggest future researches in this direction.

2. Thermodynamically consistent flash calculation scheme

The long history of developing flash calculation schemes have established two main types of algorithms with different specifications: either at given chemical compositions ($\boldsymbol{N}$), pressure (P) and temperature (T), the so-called NPT flash, or at given chemical compositions ($\boldsymbol{N}$), volume (V) and temperature (T), the so-called NVT flash [33]. NPT flash calculation methods are popular in academia studies and widely accepted in industrial applications mainly due to its easy implementation and mature techniques. However, in conventional NPT flash frameworks, selection of the best root from the solutions of a cubic equation (e.g. PR EOS) is inevitable, which may result into slow convergence or even divergence if improper root is selected at the early stage. Besides, pressure is not always known as a priori in realistic conditions. Furthermore, a distinct and nonnegligible difference will occur on the pressure of each phase if capillarity is considered, especially for phase behavior estimation of unconventional reservoir fluids in nanopores. Compared to NVT flash, pressure has to be specified for a certain phase in the NPT flash framework. Such a procedure can be avoided in NVT flash calculation. Recently, a rapid development of NVT flash calculation schemes have attracted attentions of the petroleum academia and industry. As there is no more need to invert the EOS, a unique pressure–volume relation is obtained so that the root selection procedure can be avoided. A robust NVT flash calculation scheme was proposed in [34] for two-phase equilibrium calculations and subsequently it was extended for multiphase equilibrium [35]. In this section, we will first go through the general statements for phase equilibria at given moles, volume and temperature and introduce a dynamic model for the equilibrium calculation. This approach is used to obtain the ground truth data for further training and testing in deep learning.

2.1. General phase equilibria statements

The mole and volume constraints in NVT flash schemes read as

$$\boldsymbol{N}^G + \boldsymbol{N}^L = \boldsymbol{N}^t, \quad V^G + V^L = V^t, \tag{2.1}$$

where $\boldsymbol{N}$ denotes the vector of mole numbers, V denote the volume. The superscripts G and L represent the gas phase and liquid phase respectively. The phase equilibria is considered at specified total mole numbers $\boldsymbol{N}^t = \left[N_1^t, \ldots, N_M^t\right]^T$, total volume V^t and temperature T, with denoted by the subscript M the number of components in the mixture. A difference between the NVT flash scheme and NPT flash scheme is the minimization of Helmholtz free energy instead of Gibbs free energy, and the Helmholtz free energy of a vapor–liquid two-phase system is given by

$$F = f\left(\boldsymbol{n}^G\right) V^G + f\left(\boldsymbol{n}^L\right) V^L, \tag{2.2}$$

where F denotes the total Helmholtz free energy and $f(\boldsymbol{n})$ denotes the Helmholtz free energy density, which can be calculated as follows

$$f(\boldsymbol{n}) = f^{\text{ideal}}(\boldsymbol{n}) + f^{\text{repulsion}}(\boldsymbol{n}) + f^{\text{attraction}}(\boldsymbol{n}), \tag{2.3}$$

$$f^{\text{ideal}}(\boldsymbol{n}) = RT \sum_{i=1}^{M} n_i (\ln n_i - 1), \tag{2.4}$$

$$f^{\text{repulsion}}(\boldsymbol{n}) = -nRT \ \ln(1 - bn), \tag{2.5}$$

$$f^{\text{attraction}}(\boldsymbol{n}) = \frac{a(T)\,n}{2\sqrt{2}b} \ln\left(\frac{1+\left(1-\sqrt{2}\right)bn}{1+\left(1+\sqrt{2}\right)bn}\right). \tag{2.6}$$

The molar density vector $\boldsymbol{n} = [n_1, n_2, \ldots, n_M]^T$ in the above expressions are defined by

$$\boldsymbol{n}^G = \frac{\boldsymbol{N}^G}{V^G}, \quad \boldsymbol{n}^L = \frac{\boldsymbol{N}^L}{V^L}, \tag{2.7}$$

and parameter $a(T)$ and b is the energy parameter and the co-volume of PR EOS, which can be computed based on the classical Van der Waals mixing rule

$$a(T) = \sum_{i=1}^{M}\sum_{j=1}^{M} x_i x_j \left(a_i a_j\right)^{1/2}\left(1-k_{ij}\right), \quad b = \sum_{i=1}^{M} x_i b_i, \tag{2.8}$$

with

$$a_i = 0.45724\frac{R^2 T_{c,i}^2}{P_{c,i}}\left[1+m_i\left(1-\sqrt{T_{r,i}}\right)\right]^2, \tag{2.9}$$

$$b_i = 0.07780\frac{RT_{c_i}}{P_{c,i}}, \tag{2.10}$$

where x_i is the mole fraction of component i, P_c and T_c represent the critical pressure and critical temperature, and k_{ij} is the binary interaction coefficient between ith component and jth component The coefficient m_i is often calculated by an empirical correlation with respect to the acentric factor, ω_i [33].

2.2. Dynamic model

Successive Substitution Iteration method, also known as SSI method, is commonly used in conventional iterative NPT flash calculations [29], and this numerical algorithm has been extended to NVT flash calculations as well by introducing an appropriately defined thermodynamic function [36]. Another popular algorithm in conventional NPT flash schemes, the Newton's method, has also been extensively applied in solving NVT flash problems with modified Cholesky factorization to constantly minimize the Helmholtz free energy. A dynamic evolution scheme based on diffuse interface model, also known as DI model, is developed in [33] with an energy stable numerical algorithm to describe the dynamic process from non-equilibrium state to the equilibrium state.

For a two-phase system, the volume and mole compositions of either phase can be selected arbitrarily as the primary variables, and the counterparts of the other phase can be calculated by the aforementioned constraints as shown in Eq. (2.1). For example, if we choose the vapor phase mole composition ($\boldsymbol{N}^G$) and volume (V^G) as the primary variables, the Helmholtz free energy can be reduced to $F\left(\boldsymbol{N}^G, V^G\right)$. The partial derivative of $F\left(\boldsymbol{N}^G, V^G\right)$ with respect to N_i^G and V^G can be derived as

$$\frac{\partial F\left(\boldsymbol{N}^G, V^G\right)}{\partial N_i^G} = \mu_i\left(\boldsymbol{n}^G\right) - \mu_i\left(\boldsymbol{n}^L\right) = \mu_i^G - \mu_i^L, \tag{2.11}$$

$$\frac{\partial F\left(\boldsymbol{N}^G, V^G\right)}{\partial V^G} = P\left(\boldsymbol{n}^L\right) - P\left(\boldsymbol{n}^G\right) = P^L - P^G, \tag{2.12}$$

where μ_i denotes the chemical potential of component i and P denotes the phase pressure. The time derivative of the total Helmholtz free energy $F\left(\boldsymbol{N}^G, V^G\right)$ are given by the chain rule

$$\begin{aligned}\frac{\partial F}{\partial t} &= \sum_{i=1}^{M}\frac{\partial F}{\partial N_i^G}\frac{\partial N_i^G}{\partial t} + \frac{\partial F}{\partial V^G}\frac{\partial V^G}{\partial t} \\ &= \sum_{i=1}^{M}\left(\mu_i^G - \mu_i^L\right)\frac{\partial N_i^G}{\partial t} + \left(p^L - p^G\right)\frac{\partial V^G}{\partial t}.\end{aligned} \tag{2.13}$$

In terms of Eq. (2.13), the evolution equations of mole and volume can be formulated based on the Onsager's reciprocal principle

$$\frac{\partial N_i^G}{\partial t} = \sum_{j=1}^{M} \psi_{i,j} \left(\mu_j^G - \mu_j^L\right) + \psi_{i,M+1} \left(p^L - p^G\right) , \tag{2.14}$$

$$\frac{\partial V^G}{\partial t} = \sum_{j=1}^{M} \psi_{M+1,j} \left(\mu_j^G - \mu_j^L\right) + \psi_{M+1,M+1} \left(p^L - p^G\right) . \tag{2.15}$$

The computational efficiency and reliability require an energy stable system consistent with the second law of thermodynamics. In other words, the dissipation of the total Helmholtz free energy is expected, which implies the Onsager coefficient matrix $\boldsymbol{\Psi} = \left(\psi_{i,j}\right)_{i,j=1}^{M+1}$ inclusive in Eqs. (2.14) and (2.15) shall be negative definite. For this purpose, a modified Cholesky factorization [37] is employed to ensure this negative definiteness if necessary. At the equilibrium state, Helmholtz free energy is minimized, indicating the pressure equilibrium ($p^G = p^L$) and the chemical equilibrium ($\mu_i^G = \mu_i^L$). It should be mentioned that this pressure equilibrium holds only if no capillary pressure is considered. Various mechanisms found in reservoir fluids, including partial miscibility, compressibility and capillarity, have been incorporated well in this dynamic model. Such extensions, as well as the proof of holding the first and second law of thermodynamics, can be seen in details in [33,38].

2.3. Numerical experiments

Phase stability test is usually needed before phase splitting calculation in general isothermal–isochoric phase equilibrium calculations. Different implementation methods were proposed in [39,40] and an energy-stable evolution scheme has been created to ensure dissipation of the Helmholtz free energy. It is stated in [40] that the fully implicit Euler's method and fully explicit method will both fail to preserve the energy dissipation properties in the time-marching scheme unless a small time step is applied. In other words, such evolution schemes are not unconditionally stable with strict restrictions on the time step. To generate an energy stable numerical scheme, the convex splitting technique is used with a semi-implicit evolution scheme to relax the restriction on timestep and facilitate the decline of Helmholtz free energy. This convex–concave splitting of the Helmholtz free energy density is given by

$$f\left(\boldsymbol{n}\right) = f^{\text{convex}}\left(\boldsymbol{n}\right) + f^{\text{concave}}\left(\boldsymbol{n}\right) , \tag{2.16}$$

where,

$$f^{\text{convex}}\left(\boldsymbol{n}\right) = \left(1+\lambda\right) f^{\text{ideal}}\left(\boldsymbol{n}\right) + f^{\text{repulsion}}\left(\boldsymbol{n}\right) , \tag{2.17}$$

$$f^{\text{concave}}\left(\boldsymbol{n}\right) = f^{\text{attraction}}\left(\boldsymbol{n}\right) - \lambda f^{\text{ideal}}\left(\boldsymbol{n}\right) . \tag{2.18}$$

Correspondingly, the chemical potential can also be split into two parts. In practice, the concave parts of the Helmholtz free energy density and chemical potential are treated explicitly while the convex parts are treated implicitly. Newton–Raphson method is applied to solve this resultant nonlinear system with line search.

In this paper, two data sets are generated using this thermodynamic-consistent flash calculation scheme for an 8-component Eagle Ford oil and a 14-component Eagle Ford oil respectively. Molar compositions and compositional properties needed for flash calculations are listed in Tables 1 and 2. The 2D computational domain of overall concentration and temperature is discretized into a uniform grid with size of 201 $\times$ 201 to obtain a smooth phase envelope. The investigated overall molar concentration for these two mixtures vary from $[10, 12\,000]$ and $[10, 10\,000]$ mol/m^3 respectively, and the temperature interval is same for both fluid samples with $T \in [260, 850]$ K. The computed phase envelop and TPD plot of the 8-component Eagle Ford oil are illustrated in Fig. 2.1, which meet well with the published results in [38]. These reliable flash calculation data will be used as the ground-truth training and testing data in the following deep learning algorithm.

3. Deep learning algorithm

The details of our neural network structure and deep learning techniques are explained in this section. The design of a self-adaptive two-network scheme can handle the task of predicting complex reservoir fluids with a large number of components using limited iterative flash calculation data for a smaller number of components, by

Table 1
Compositional parameters for the 8-component Eagle Ford oil.

Component	z_i	$T_{c,i}$ (K)	$P_{c,i}$ (MPa)	$M_{w,i}$ (g/mol)	ω_i
CO_2	0.0232	304.11	7.3739	44.01	0.2250
C_1	0.5816	190.56	4.5988	16.04	0.0110
C_2	0.0744	305.33	4.87180	30.07	0.0990
C_3	0.0417	369.83	4.2479	44.10	0.1520
nC_4	0.0259	418.71	3.7383	58.12	0.1948
C_5–C_6	0.0269	485.60	3.3767	76.502	0.2398
C_{7+}	0.1321	606.69	2.7083	122.96	0.3548
C_{13+}	0.0942	778.31	1.5598	255.28	0.7408

Table 2
Compositional parameters for the 14-component Eagle Ford oil.

Component	z_i	$T_{c,i}$ (K)	$P_{c,i}$ (MPa)	$M_{w,i}$ (g/mol)	ω_i
CO_2	0.01282	304.22	7.3864	44.01	0.2250
C_1	0.31231	190.72	4.6409	16.04	0.0130
N_2	0.00073	126.22	3.3943	28.01	0.0400
C_2	0.04314	305.44	4.8842	30.07	0.0986
C_3	0.04148	369.89	4.2568	44.10	0.1524
iC_4	0.01350	408.11	3.6480	58.12	0.1848
nC_4	0.03382	425.22	3.7969	58.12	0.2010
iC_5	0.01805	460.39	3.3336	72.15	0.2223
nC_5	0.02141	469.78	3.3750	72.15	0.2539
nC_6	0.04623	507.89	3.0316	86.18	0.3007
C_{7+}	0.16297	589.17	2.7772	114.40	0.3739
C_{11+}	0.12004	679.78	2.1215	166.60	0.5260
C_{15+}	0.10044	760.22	1.6644	230.10	0.6979
C_{20+}	0.07306	896.78	1.0418	409.20	1.0456

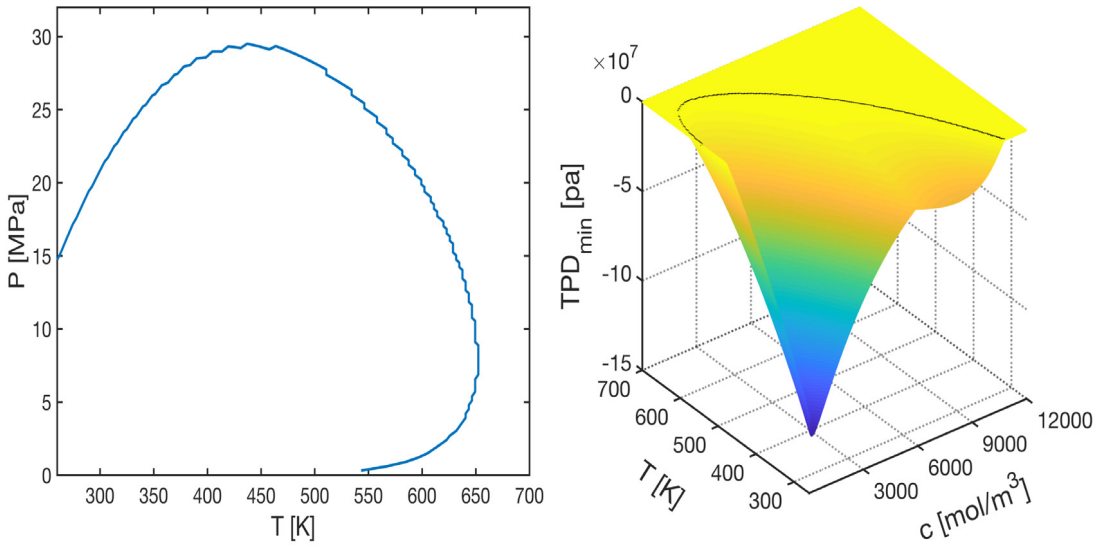

Fig. 2.1. Left: Phase envelop of the 8-component Eagle Ford oil; Right: TPD plot of the 8-component Eagle Ford oil.

uniformizing the dimension of training and target fluid mixtures. Ghost components are defined and introduced for the data padding, and the efficiency and reliability of this approach are verified in Section 4. Certain popular techniques in deep learning studies are involved in this work to achieve a better performance of our algorithm.

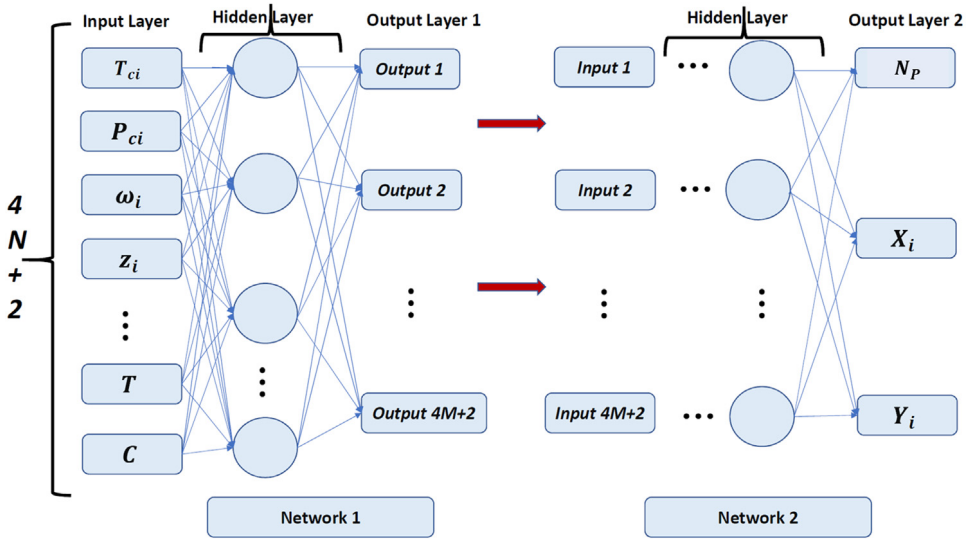

Fig. 3.1. The two-network structure.

3.1. Network structure

Artificial neural networks are designed and constructed to extract the key features of the input data and discover the underneath correlations between the input and output [41,42]. A two-network structure is proposed in this paper to accelerate the flash calculation in general cases without limitation on the number of components. As shown in Fig. 3.1, we first extract the input features from the training data, which is the flash calculation results of a reservoir fluid mixture with N components. The input features are selected based on investigating the numerical algorithms described in Section 2 to find the critical parameters underlying and controlling the thermodynamic equilibrium. The selection of temperature (T), overall molar concentration (C), and the critical temperature $T_{c,i}$, critical pressure $P_{c,i}$, acentric factor ω_i and mole fraction z_i as input features is verified in [30]. Thus, the input dimension is $4N+2$. In the meantime, the network will read the information of target fluid mixture with M components. Secondly, the first network is constructed with the output layer as a dimension of $4M+2$, and each output refer to an index controlling the input of the second network. If $N < M$, all the information in the training data will be reserved and a padding technique is performed to fulfill the training data with $M - N$ "ghost components". The "ghost components" are defined with all the critical properties to be zero, and the molar fraction is zero as well. If $N > M$, the algorithm will set $M = N$ in the first network to output an index with the dimension of $4N + 2$, and the testing data (target fluid mixture) in Network 2 will be padded with certain $N - M_0$ "ghost components", where M_0 refers to the original number of components in the target fluid mixture. Thus, the training and testing data are uniformized to one certain dimension, and the Network 2 will be benefited. Afterwards, similar techniques in [30] are performed to train and test the Network 2 and finally output the flash calculation data of the target fluid mixture. If $N < M$, the output will be of the dimension as $2M + 1$, including the mole fraction of each component in vapor and liquid phase and the total number of phases existing at the equilibrium state. If $N > M$, there will be information of the "ghost components" in the output data, and that will be eliminated automatically in our algorithm. It should be pointed out that phase stability tests are incorporated with the phase splitting calculations together in this deep learning algorithm, as the total number of phases existing in the fluid mixture at equilibrium state will be determined directly after our estimation using the trained model.

The flow chart of our deep learning algorithm in each node is shown in Fig. 3.2, which can be concluded by the following equation,

$$\mathbf{y}_i = f_i(\mathbf{W}_i * \mathbf{a}_i + \mathbf{b}_i), \tag{3.1}$$

where $\mathbf{a}_i$ denotes the input of the layer i, $\mathbf{y}_i$ denotes the output of this layer, $\mathbf{W}_i$ denotes the weight, f_i denotes the activation function and $\mathbf{b}_i$ denotes the bias. The output of one hidden layer is the input of the next one, and this

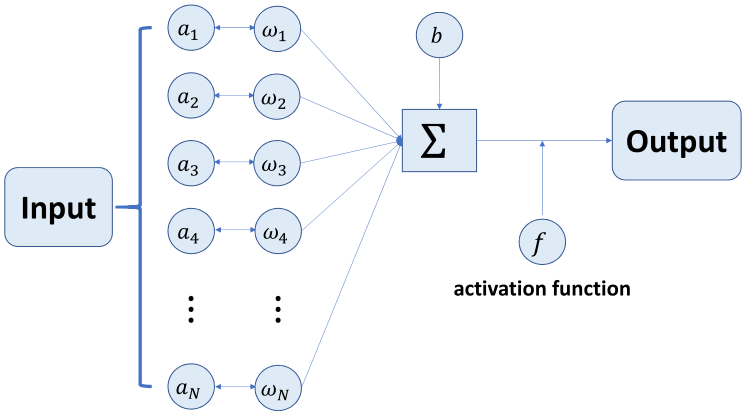

Fig. 3.2. The simulation process in each node.

relation can be formulated as (e.g. using three activation layers)

$$\mathbf{o} = f_3(\mathbf{W}_3 * f_2(\mathbf{W}_2 * f_1(\mathbf{W}_1 * \mathbf{x}_1 + \mathbf{b}_1) + \mathbf{b}_2) + \mathbf{b}_3), \tag{3.2}$$

where $\mathbf{o}$ is the final output, $\mathbf{x}_1$ is the input of the first layer, f_1, f_2, f_3 are the activation functions in Layer 1, 2, 3 respectively, $\mathbf{W}_1$, $\mathbf{W}_2$, $\mathbf{W}_3$ are the weights of each layer and $\mathbf{b}_1$, $\mathbf{b}_2$, $\mathbf{b}_3$ are the bias terms of each layer.

In our self-adaptive deep neural network, the trained model can be automatically adjusted to pad the "ghost components" in either input data or output data. Specifically, if $N < M$, $M - N$ "ghost components" will be padded into the input data of the training network, so that $\mathbf{x}_1$ in Eq. (3.2) is written as

$$\begin{aligned}\mathbf{x}_1 = \big\{&T_{c,1}, P_{c,1}, \omega_1, z_1, \ldots, T_{c,N}, P_{c,N}, \omega_N, z_N, T_{c,g_1}, P_{c,g_1}, \omega_{g_1}, z_{g_1}, \ldots, T_{c,g_{(M-N)}},\\ &P_{c,g_{(M-N)}}, \omega_{g_{(M-N)}}, z_{g_{(M-N)}}, T, C\big\},\end{aligned} \tag{3.3}$$

and the output $\mathbf{o}$ remains as

$$\mathbf{o} = \{P, X_1, X_2, \ldots, X_M, Y_1, Y_2, \ldots, Y_M\}. \tag{3.4}$$

On the other hand, if $N > M$, $N - M$ "ghost components" will be padded into the testing data as

$$\mathbf{o} = \big\{P, X_1, X_2, \ldots, X_M, X_{g_1}, X_{g_2}, \ldots, X_{g_{(N-M)}}, Y_1, Y_2, \ldots, Y_M, Y_{g_1}, Y_{g_2}, \ldots, Y_{g_{(N-M)}}\big\}, \tag{3.5}$$

while the input data $\mathbf{x}_1$ remain as

$$\mathbf{x}_1 = \big\{T_{c,1}, P_{c,1}, \omega_1, z_1, \ldots, T_{c,N}, P_{c,N}, \omega_N, z_N, T, C\big\}. \tag{3.6}$$

The parameters in the above formulations with subscript g denote the compositional variables of the "ghost components", which are all set as 0 in our paper.

Overfitting is a common issue in deep learning studies, which means that a perfect performance of the trained model has been achieved for the training data but a poor performance can only be obtained for the validation of the testing data. Generally, this problem occurs if too many parameters are involved in the deep learning model, which is also known as an over-parameterized model. Usually, an additional constraint is applied to reduce the freedom of our model to prevent the overfitting problem from damaging the performance of our deep learning algorithm. A significant feature of overfitting is a very large norm of the weight parameters, and this could be one way to add this additional constraint. In practice, the large weights can be penalized by modifying the loss function with an additional regularization term proportional to the L2 norm of the weights, as formulated in Eq. (3.7)

$$L = \frac{1}{N}\sum_{n=1}^{N} \|\mathbf{o} - \widehat{\mathbf{o}}\|^2 + \lambda \|\mathbf{W}\|_2^2, \tag{3.7}$$

where L denotes the loss function, $\widehat{\mathbf{o}}$ denotes the observation value, $\mathbf{o}$ denotes the model output, N denotes the number of training data, λ denotes the regularization coefficient of the L2 weight decay term and $\mathbf{W}$ denotes the weight parameters. The selection of λ depends on how much penalization we want to perform on the large weights.

3.2. Deep learning techniques

An efficient and robust deep neural network is expected with proper initialization of the weight parameters, which show significant impacts on the performance of the deep learning algorithm, especially on the convergence rate. If the weight parameters are overestimated at the initialization, a rapid increase of the input variance will be observed, which can further cause the gradient exploding or vanishing, and in that case the training will never converge. If the weight parameters are underestimated at the initialization, on the contrary, the input variance may drop rapidly to a very small value, which results in a damaged model complexity and even bad performance of final estimation. A common-used initialization approach is to follow the Gaussian distribution, and the weights are initialized with certain variance to ensure that the input and output variance of one layer will keep the same. This approach can easily control the signal variance through the networks, and it is known as Xavier initialization. For example, in Fig. 3.2, the variance of the node output should follow the below equations

$$\begin{aligned} var(y) &= var(w_1 * a_1 + w_2 * a_2 + \cdots + w_n * a_n + b) \\ &= var(w_1) * var(a_1) + var(w_2) * var(a_2) + \cdots + var(w_n) * var(a_n) \\ &\overset{(1)}{=} n * var(w_i) * var(a_i), \end{aligned} \tag{3.8}$$

where the equivalence (1) is meaningful only if all the w_i and a_i are assumed to be identity distributed. To meet the requirement of the same variance of output (y) and input (a_i), it is easy to get

$$n * var(w_i) = 1, \tag{3.9}$$

which indicates that the initial weights should follow a Gaussian distribution with a determined variance $1/n$. The parameter n is the number of weights in that layer, as illustrated in Fig. 3.2.

Dropout technique is also of vital importance to efficiently improve the deep learning performance. Due to the high complexity of the neural network representing the thermodynamic correlations, overfitting is hard to avoid and always seriously affects the final estimation. By discarding certain nodes in the network, as shown in Fig. 3.3, the corresponding connections will be dropped out as well and the network freedom is reduced significantly in this manner. After dropout, the output of the layer can be expressed as

$$\mathbf{y}_i = f_i * (\mathbf{W}_i * \mathbf{r}_i * \mathbf{a}_i + \mathbf{b}_i), \tag{3.10}$$

where $\mathbf{r}_i$ is a boolean vector. Each node j on layer i will be evaluated independently with the probability of p to be remained or the probability of $1 - p$ to be discarded. If it is remained, the corresponding r_j will be one and if it is discarded, r_j will be zero. All the nodes and connections are evaluated after this process, and the training stage will be performed on the reduced network. It should be pointed out that the discarded nodes will be inserted back to the model with the initial weights after the training stage before the next training cycle begins in order to keep the robustness and consistency.

Due to the large number of parameters involved in model training and ultra-large scale of input flash calculation data, especially for the complex network structure representing the thermodynamic correlations among large number of components in a mixture, training the deep learning model is notoriously time-consuming and challenging the final efficiency performance of this algorithm. Furthermore, there are always certain undesirable properties underneath the deep hidden layers that slow down the training convergence rate, making it difficult to efficiently optimize this large scale system. One typical property is the “internal covariate shift”, which means the change of each layer’s input due to the changing of parameters on the previous layer. A simple solution to handle this problem is “whitening”, by which the mean and variance of input features can remain the same. However, this technique is time consuming and some of the parameter information may be discarded by changing the distribution of each layer. As a result, Batch normalization, a technique to normalize the original input of the first layer as well as on hidden layers, has been used and proved to accelerate the convergence rate significantly. Mean and variance of each batch will be calculated and then be used to normalize the batch data by introducing a small value to avoid zero-variance cases.

The non-linear correlations underneath the input features (represented by EOS in conventional flash calculation methods) are held by activation functions. As a key factor to represent the network nonlinearity and problem-solving capability, a variety of activation functions have been proposed to fit various problems. It has been reported in

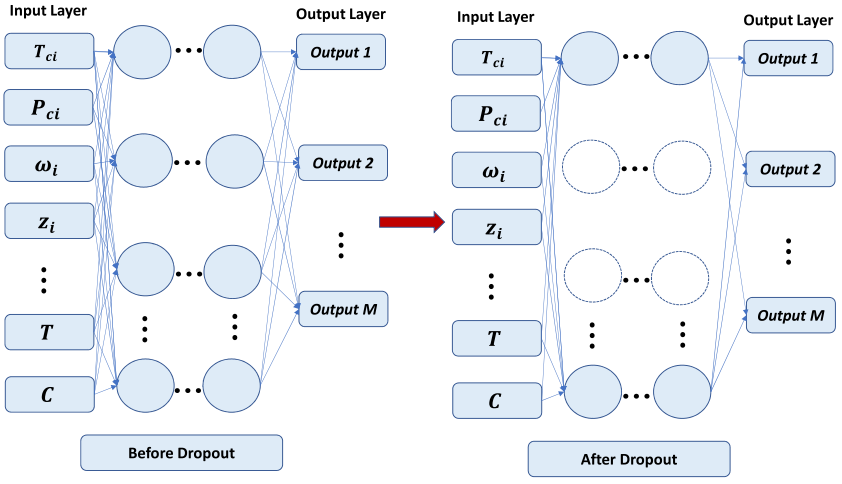

Fig. 3.3. Dropout Network.

previous studies [29,30] that both activation functions, "ReLU" and "Sigmoid", exhibit good performance in deep learning algorithms for accelerating flash calculations. ReLU function reads as

$$f(x) = \begin{cases} 0, & \text{if } x < 0 \\ x, & \text{if } x \geq 0 \end{cases}, \tag{3.11}$$

while Sigmoid function has the following form

$$\sigma(x) = \frac{1}{1 + \exp(-x)}. \tag{3.12}$$

Due to the padding process in our two-network structure and changing of input dimensions, the underlying constraint, the summation of mole fractions of all phases of each phase being one, is challenged in the output results. Softmax function, which was proposed to normalize a set of value to follow a probability distribution summing up to 1, has attracted our attention and applied in the final processing on the network output. This function can be formulated as

$$a_j^L = \frac{e^{z_j^L}}{\sum_k e^{z_k^L}}, \tag{3.13}$$

where a_j^L denotes the Softmax value of the jth data in layer L, z_j^L and z_k^L denote all the data in this layer. By computing the ratio of the exponential of the mole fraction of component j in phase α (α could represent either gas or liquid phase) and the sum of exponential parameters of all the mole fractions in that phase, the final output of our deep neural network can meet the total constraint and thus physically meaningful.

4. Results and discussion

In this section an example will be provided, with an 8-component Eagle Ford oil flash calculation data as input to train the model to predict the thermodynamic equilibrium properties of a 14-component Eagle Ford oil. This example can verify the idea of using our well-designed two-network deep learning algorithm to predict the thermodynamic equilibrium properties of realistic complex reservoir fluid mixtures with a large number of components, using an acceptable iterative flash results of a smaller number of components for model training. Totally 90601 data points are included in the 8-component training data and 14-component testing data, and in the testing process we randomly select a batch of 9061 data points in the 14-component data. This data ratio as 0.9 : 0.1 for training and testing is verified in [29].

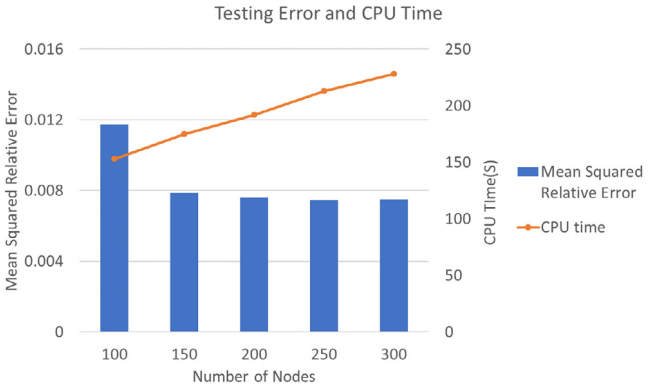

Fig. 4.1. Mean squared relative error of testing data using different number of nodes in the hidden layers of Network 1 with the corresponding CPU time(s).

4.1. Network tuning

As the input training data is of a smaller dimension compared with the output flash calculation results, the first network should be tuned carefully before optimizing the second network hyperparameters to determine a reasonable and meaningful data padding($N < M$ case explained in Section 3.1). It can be easily referred from Section 3.2 that the "ReLU" activation function is suitable for outputting such index to control the data padding. Afterwards, the number of nodes in the hidden layers in Network 1 is optimized to achieve a better balance between prediction accuracy and efficiency. As shown in Fig. 4.1, the CPU time used for the deep learning algorithm increases significantly with the increasing number of nodes used in each hidden layer. However, the mean squared relative error decreases sharply with the number of nodes changing from 100 to 150 and then keeps a very slow decrease as the number nodes increases. The error differences between the usage of 200, 250 and 300 nodes are so small that the choice of 200 nodes on hidden layers in Network 1 is reasonable promising. In this tuning process, we keep 150 nodes for the hidden layers in Network 2.

After tuning the Network 1, hyperparameters in Network 2 are optimized. The first comparison is conducted using different activation functions in the hidden layers in Network 2. Five commonly used activation functions, Sigmoid, Softplus, Softsign, tanh and ReLU are used in the network and we compare the performance indicated by the mean squared absolute error and mean squared relative error. As shown in Fig. 4.2, the network with four activation functions, Sigmoid, Softplus, Softsign and tanh, exhibits similar prediction errors in the testing, while the error of using ReLU is a little bit higher. Furthermore, as testing data are randomly selected from the iterative flash results, repeated testing using different batches is performed to improve the reliability of our tuning on the activation function. It can be referred from Fig. 4.3 that the averaged mean squared relative errors in 200 trials of using the four "good" activation functions in Fig. 4.2 will finally converge to a similar value. This repeated testing is convincible, because the averaged CPU time for each testing is only 2.71 s. Obviously the deep learning algorithm can indeed accelerate a lot the flash calculations compared with the CPU time used for iterative methods in [30]. However, it is interesting to see that the error of using Softplus is much more stable compared with the other three, so that this activation function is selected in Network 2 to better improve the stability and robustness of our deep learning algorithm.

The number of nodes in hidden layers in Network 2 is optimized in a similar way to achieve a better balance between accuracy and efficiency. It can be seen from Fig. 4.4 that an obvious decrease of mean squared relative error takes place with the increase of number of nodes from 50 to 200 and the change is negligible with more nodes. The CPU time increases much faster, compared with Fig. 4.1, so that the final choice is 200 nodes in order to obtain a good prediction accuracy (mean squared relative error lower than 0.008) in an acceptable computational time.

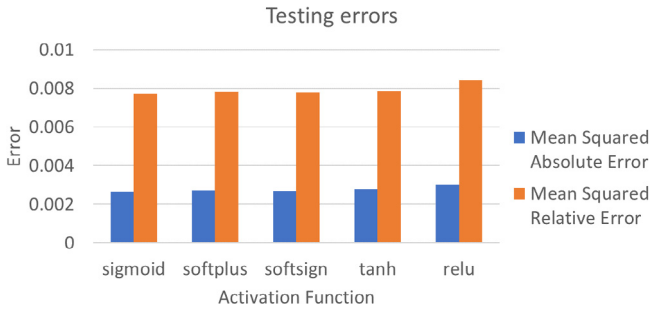

Fig. 4.2. Mean squared absolute error and mean squared relative error of testing data using different activation functions.

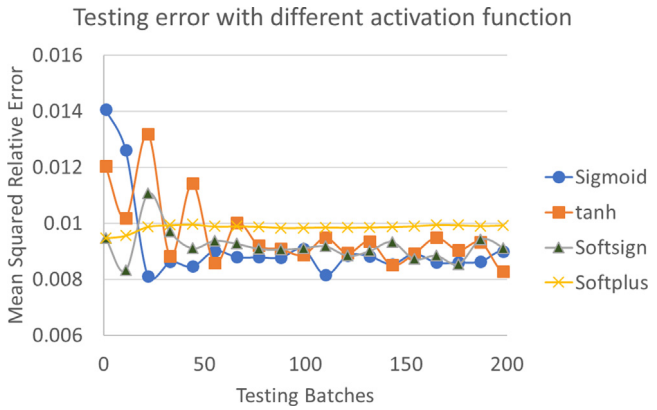

Fig. 4.3. The averaged mean squared relative error of testing data using different activation functions (blue points for Sigmoid, orange squares for tanh, green triangles for Softsign and yellow crosses for Softplus) in a number of testing batches.

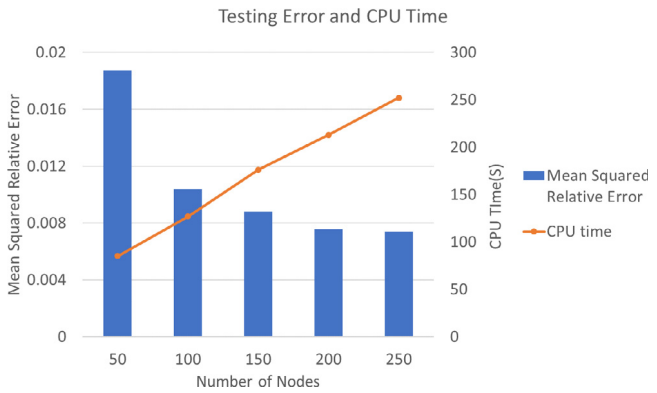

Fig. 4.4. Mean squared relative error of testing data using different number of nodes in the hidden layers of Network 2 with the corresponding CPU time(s).

4.2. Prediction evaluation

A good prediction accuracy can be obtained using our deep learning algorithm with a very small error as illustrated in Section 4.1, and visual comparisons are provided in this section to illustrate and evaluate our deep learning prediction with optimized network configurations. Conventional flash calculation frameworks are often consisting of two stages: phase stability test first and then phase splitting calculation. One significant feature of our deep learning algorithm is to automatically determine the total number of phases existing in the mixture at

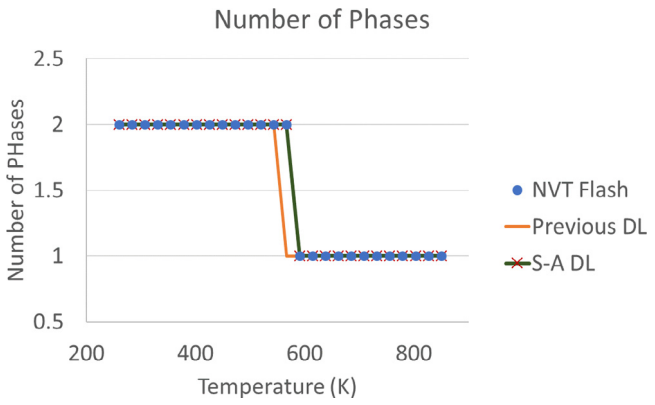

Fig. 4.5. Number of phases existing at equilibrium, predicted by the NVT flash calculation (blue points), previous deep learning algorithms (orange line) and the new self-adaptive deep learning algorithm (red cross), as a function of temperature under the specified overall concentration of 129.9 mol/m^3. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

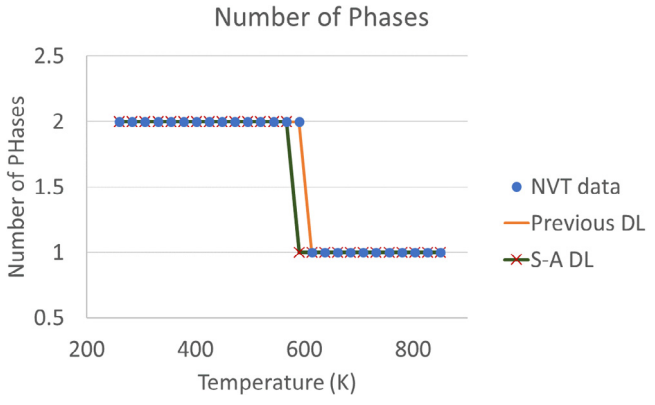

Fig. 4.6. Number of phases existing at equilibrium, predicted by the NVT flash calculation (blue points), previous deep learning algorithms (orange line) and the new self-adaptive deep learning algorithm (red cross), as a function of temperature under the specified overall concentration of 249.8 mol/m^3. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

equilibrium. No independent phase stability test is needed so that our flash calculation is accelerated further. As shown in Fig. 3.1, the molar compositions of each phase (X, Y) are predicted at the final output together with the total number of phases existing (N_P) by simultaneously completing the stability test and splitting calculation in the learning process. To validate this automatic characterization of total number of phases in our prediction, comparisons are performed among the iterative NVT flash calculation method, a previous deep learning algorithm (the training and testing data are for the same number of components) and our self-adaptive deep learning algorithm. The total number of phases existing in the mixture at equilibrium as a function of environmental temperature is shown in Fig. 4.5 and Fig. 4.6, with the overall concentrations are 129.9 mol/m^3 and 249.8 mol/m^3 respectively. It is indicated that although ghost components are introduced to pad data for training process, this new self-adaptive deep learning algorithm can yield a reliable phase stability prediction similar to the previous limited deep learning algorithms. Compared with the iterative NVT flash calculation scheme, which is used as ground truth in training and testing, only a slight difference can be detected at the temperature changing from two-phase region to single-phase region at overall concentration of 249.8 mol/m^3 in Fig. 4.6, but our deep learning model can perfectly capture the phase transition process from the vapor–liquid region to the single vapor region in Fig. 4.5. In all, the prediction on the total number of phases at equilibrium using our optimized deep learning algorithm is reliable and can be taken into consideration in further multi-phase flow simulations.

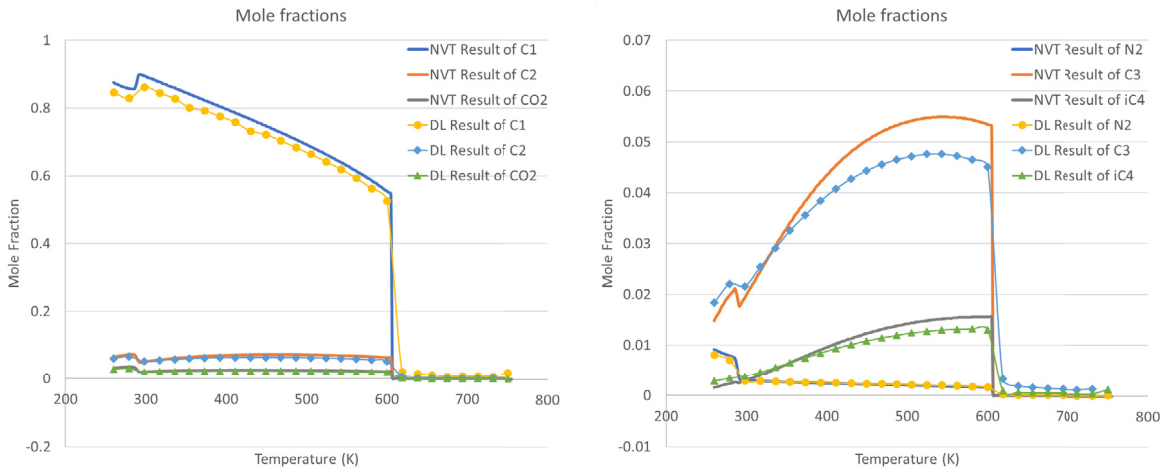

Fig. 4.7. Mole fractions of C_1, C_2, CO_2 in the left figure, and N_2, C_3, iC_4 in the right figure, in the liquid phase of the 14-component Eagle Ford oil as a function of temperature under the specified overall concentration of 3810 mol/m^3. The line and solid symbols represent the NVT flash results and self-adaptive deep learning results, respectively.

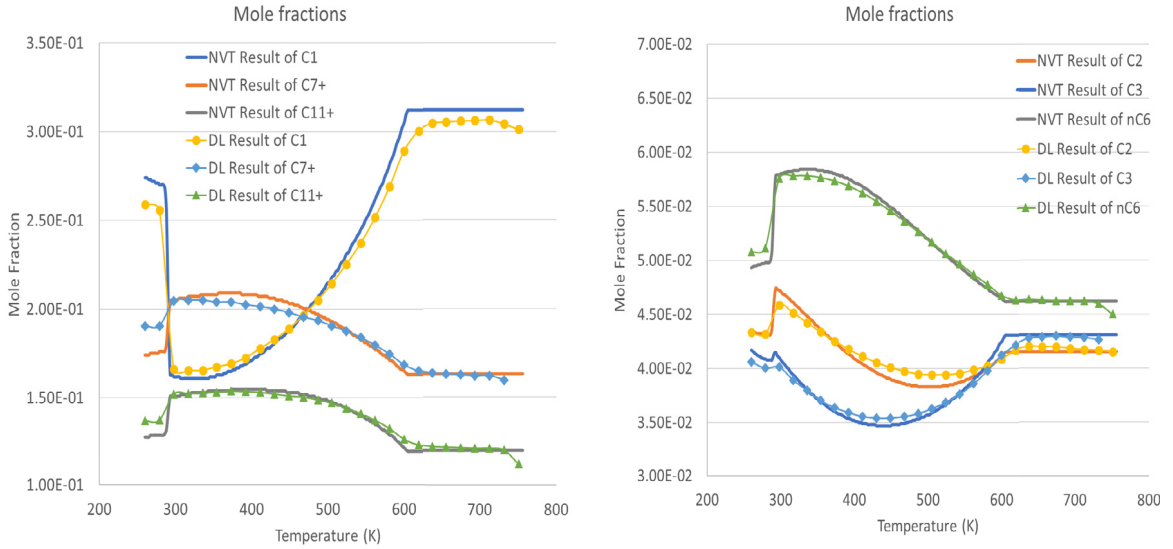

Fig. 4.8. Mole fractions of C_1, C_{7+}, C_{11+} in the left figure, and C_2, C_3, nC_6 in the right figure, in the vapor phase of the 14-component Eagle Ford oil as a function of temperature under the specified overall concentration of 3810 mol/m^3. The line and solid symbols represent the NVT flash calculation results and self-adaptive deep learning results, respectively.

Molar compositions of each phase at equilibrium is the key information required by multi-phase multi-component flow simulations. A comparison of molar compositions computed by iterative NVT flash calculations and the self-adaptive deep learning algorithm is illustrated in Figs. 4.7 and 4.8 for the liquid and vapor phase of the 14-component Eagle Ford oil at an overall concentration of 3810 mol/m^3. The mole fractions of C_1, C_2, CO_2, N_2, C_3, iC_4 in the liquid phase and that of C_1, C_{7+}, C_{11+}, C_2, C_3, nC_6 in the vapor phase are presented, and our deep learning prediction results meet well with the iterative NVT Flash data. It can be further observed that phase transitions occur when temperature increases, from vapor–liquid two phase region to single vapor region, so that the mole fractions of the liquid phase are all vanished at around 600 K. The small variance between the results of deep learning prediction and iterative flash calculation represents a good robustness and reliability of this approach.

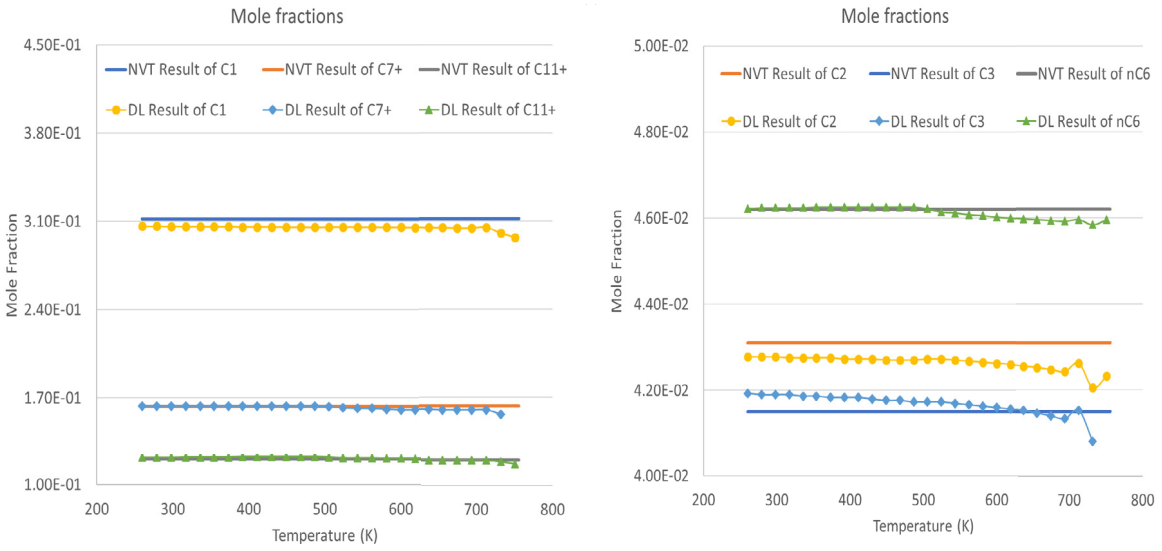

Fig. 4.9. Mole fractions of C_1, C_{7+}, C_{11+} in the left figure, and C_2, C_3, nC_6 in the right figure, in the vapor phase of the 14-component Eagle Ford oil as a function of temperature under the specified overall concentration of 7200 mol/m^3. The line and solid symbols represent the NVT flash calculation results and self-adaptive deep learning results, respectively.

Another example is tested to further verify the reliability of our new deep learning algorithm. As indicated by Fig. 4.9, the 14-component Eagle Ford oil is in the single vapor phase state at the given overall concentration of 7200 mol/m^3, and this feature is captured perfectly by our deep learning model. Mole fractions of the selected components remain the same at this temperature interval. The capability to complete both phase stability test and phase splitting calculations in our deep learning algorithm is proved again.

5. Conclusion and remarks

A novel two-network deep learning algorithm is proposed in this paper, to meet the demand of estimating the thermodynamic equilibrium states of realistic reservoir fluids with a large number of components, by using an iterative flash calculation data of certain reservoir fluid with a relatively smaller number of components. Using our algorithm, the computational time for iterative flash calculation can be restricted to an acceptable range, while the large scale flash calculation can be accelerated with deep learning algorithms with a robust and reliable approach. A thermodynamically consistent NVT flash calculation scheme is generated with an energy stable evolution scheme, so that we can prepare as much flash calculation data for the network training as possible in an acceptable CPU time cost to overcome the overfitting problem in deep learning. This network structure is self-adaptive, as it can automatically detect the dimension of training and testing data and further construct the input and output dimensions of the two networks correspondingly. The data dimension is correlated with the number of components existing in each fluid mixture, and certain thermodynamic properties of each component is selected as the input features controlling the equilibrium process. By using padding techniques in the first network and the definition of "ghost components", the training and testing data can be uniformed to a certain dimension, which can benefit the model training in the second network. Hyperparameters are carefully tuned to obtain a good balance between efficiency and accuracy, and Softmax function is introduced to enforce the constraint that the summation of phase mole fractions should be one.

This satisfactory robustness and accurate thermodynamic equilibrium predictions can promise a good potential in this direction for future studies on accelerating flash calculation for realistic complex reservoir fluids with as much components as needed. Our selection of compositional properties as input features is verified again with the padding of ghost components, and this labeling technique can be easily extended to characterize a large number of components in various cases if needed. Following remarks are concluded to help future researches in this field:

1. Optimal Network Hyperparameters are tuned carefully in this paper as a similar work of [29,30]. This set of network hyperparameters, either on Network 1 or Network 2, has shown a good performance for this 8-component to 14-component case. However, it is interesting to investigate further the network features in more extreme cases. For example, if we want to predict the thermodynamic equilibrium states of a reservoir mixture with up to 50 components using a trained model from iterative flash calculation data of a mixture with 8 components, maybe the optimized network features are different with our examples. Thus, a lot more tests and tuning work should be conducted to find the changing rule of the optimized network features for various proportions between the number of training and target components. Of course, the iterative flash calculation schemes are also expected to be optimized deeper to prepare the data of such large scales to feed the network and overcome potential overfitting problems caused by data shortage.

2. The training and testing data are generated using an iterative NVT flash calculation method. This method is proved to be thermodynamically consistent and energy stable in previous publications, and the flash calculation results are verified with previous literature. However, experimental data are always expected as the EOS used in iterative flash schemes are empirical expressions regressed from experimental data. Thus, if the flash calculation data was used for training in the deep learning algorithm, the reliability might also be questioned that we are not using the realistic ground truth. It can be referred from [29] that the performance of deep learning models trained from experimental data can sometimes be better than the iterative flash calculation results. Unfortunately, getting sufficient experimental data for phase equilibrium is extremely difficult and expensive, especially for complex reservoir fluids with a large number of components in a wide range of environmental conditions. If the data are limited, overfitting is difficult to avoid in deep learning so that the trained models are not practically useful. As a result, the current approach to use iterative flash calculation data remains the best option, and we are also looking for a better iterative flash scheme to better approximate the realistic experimental data. Furthermore, the application of trained self-adaptive deep learning model on practical compositional reservoir simulators [32] is also expected, especially on well-designed algorithms considering complex mechanisms [43], strictly preserving physical conservations [44] or high-performance schemes [45].

3. Newly developed translation models in deep neural networks, including RNN, LSTM and the attention model, are expected to be incorporated to further improve the performance of our deep learning algorithm. Among them, the attention model is the most promising one since it can model the pair-wise interaction between two components which is consistent with the real-world case. Further investigations on the customization and tailor of these new and strong models may be another solution to handle the large scale flash calculation tasks, as we can benefit a lot from their applications on other areas with similar problems.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Guangpu Zhu, Aifen Li, Interfacial dynamics with soluble surfactants: A phase-field two-phase flow model with variable densities, Adv. Geo-Energy Res. 4 (1) (2020) 86–98.
[2] Shuyu Sun, Darcy-scale phase equilibrium modeling with gravity and capillarity, J. Comput. Phys. 399 (2019) 108908.
[3] Ding-Yu Peng, Donald B. Robinson, A new two-constant equation of state, Ind. Eng. Chem. Fundam. 15 (1) (1976) 59–64.
[4] Giorgio Soave, Equilibrium constants from a modified Redlich-Kwong equation of state, Chem. Eng. Sci. 27 (6) (1972) 1197–1203.
[5] Curtis H. Whitson, Michael R. Brulë, Phase Behavior, Vol. 20, Henry L. Doherty Memorial Fund of AIME, Society of Petroleum Engineers, Richardson, TX, 2000.
[6] Michael L. Michelsen, Simplified flash calculations for cubic equations of state, Ind. Eng. Chem. Process Des. Dev. 25 (1) (1986) 184–188.
[7] Abbas Firoozabadi, Huanquan Pan, Fast and robust algorithm for compositional modeling: Part i-stability analysis testing, in: SPE Annual Technical Conference and Exhibition, Society of Petroleum Engineers, 2000.
[8] Huanquan Pan, Abbas Firoozabadi, Fast and robust algorithm for compositional modeling: Part ii-two-phase flash computations, in: SPE Annual Technical Conference and Exhibition, Society of Petroleum Engineers, 2001.
[9] Yinghui Li, Russell T. Johns, Rapid flash calculations for compositional simulation, SPE Reservoir Eval. Eng. 9 (05) (2006) 521–529.
[10] Bjarne H. Jensen, Aage Fredenslund, A simplified flash procedure for multicomponent mixtures containing hydrocarbons and one non-hydrocarbon using two-parameter cubic equations of state, Ind. Eng. Chem. Res. 26 (10) (1987) 2129–2134.

[11] Vassilis Gaganis, Nikos Varotsis, An improved BIP matrix decomposition method for reduced flash calculations, Fluid Phase Equilib. 340 (2013) 63–76.
[12] Peng Wang, Erling H. Stenby, Non-iterative flash calculation algorithm in compositional reservoir simulation, Fluid Phase Equilib. 95 (1994) 93–108.
[13] Claus P. Rasmussen, Kristian Krejbjerg, Michael L. Michelsen, Kersti E. Bjurstrøm, Increasing computational speed of flash calculations with applications for compositional, transient simulations, in: SPE Annual Technical Conference and Exhibition, Society of Petroleum Engineers, 2003.
[14] Yuanqing Wu, Christoph Kowitz, Shuyu Sun, Speeding up the flash calculations in two-phase compositional flow simulations–The application of sparse grids, J. Comput. Phys. 285 (2015) 88–99.
[15] Viet D. Nguyen, Raymond R. Tan, Yolanda Brondial, Tetsuo Fuchino, Prediction of vapor–liquid equilibrium data for ternary systems using artificial neural networks, Fluid Phase Equilib. 254 (1–2) (2007) 188–197.
[16] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012.
[17] Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun, Faster r-cnn: Towards real-time object detection with region proposal networks, in: Advances in Neural Information Processing Systems, 2015.
[18] George E. Dahl, Dong Yu, Li Deng, Alex Acero, Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition, IEEE Trans. Audio Speech Lang. Process. 20 (1) (2011) 30–42.
[19] Jose Maria Font, Tobias Mahlmann, The Dota 2 bot competition, IEEE Trans. Games (2018).
[20] Yu Li, Sheng Wang, Ramzan Umarov, Bingqing Xie, Ming Fan, Lihua Li, Xin Gao, DEEPre: sequence-based enzyme EC number prediction by deep learning, Bioinformatics 34 (5) (2017) 760–769.
[21] Yu Li, Fan Xu, Fa Zhang, Pingyong Xu, Mingshu Zhang, Ming Fan, Lihua Li, Xin Gao, Dlbi: deep learning guided bayesian inference for structure reconstruction of super-resolution fluorescence microscopy, Bioinformatics 34 (13) (2018) i284–i294.
[22] Yu Li, Chao Huang, Lizhong Ding, Zhongxiao Li, Yijie Pan, Xin Gao, Deep learning in bioinformatics: Introduction, application, and perspective in the big data era, Methods 166 (2019) 4–21.
[23] Maria Vasilyeva, Aleksey Tyrylgin, Machine learning for accelerating effective property prediction for poroelasticity problem in stochastic media, 2018, arXiv preprint arXiv:1810.01586.
[24] Maria Vasilyeva, Aleksey Tyrylgin, Convolutional neural network for fast prediction of the effective properties of domains with random inclusions, J. Phys. Conf. Ser. 1158 (4) (2019) IOP Publishing.
[25] Cuong Dang, et al., AI based mechanistic modeling and probabilistic forecasting of hybrid low salinity chemical flooding, Fuel 261 (2020) 116445.
[26] Vassilis Gaganis, Nikos Varotsis, An integrated approach for rapid phase behavior calculations in compositional modeling, J. Pet. Sci. Eng. 118 (2014) 74–87.
[27] Abishek Kashinath, Michael L. Szulczewski, Ali H. Dogru, A fast algorithm for calculating isothermal phase behavior using machine learning, Fluid Phase Equilib. 465 (2018) 73–82.
[28] George Cybenko, Approximation by superpositions of a sigmoidal function, Math. Control Signals Syst. 2 (4) (1989) 303–314.
[29] Yu Li, Tao Zhang, Shuyu Sun, Xin Gao, Accelerating flash calculation through deep learning methods, J. Comput. Phys. 394 (2019) 153–165.
[30] Yiteng Li, Tao Zhang, Shuyu Sun, Acceleration of the NVT-flash calculation for multicomponent mixtures using deep neural network models, Ind. Eng. Chem. Res. 58 (27) (2019) 12312–12322.
[31] Kun Wang, Jia Luo, Wei Yizheng, Keliu Wu, Jing Li, Zhangxin Chen, Artificial neural network assisted two-phase flash calculations in isothermal and thermal compositional simulations, Fluid Phase Equilib. 486 (2019) 59–79.
[32] Kun Wang, Jia Luo, Wei Yizheng, Keliu Wu, Jing Li, Zhangxin Chen, Practical application of machine learning on fast phase equilibrium calculations in compositional reservoir simulations, J. Comput. Phys. 401 (2020) 109013.
[33] Jisheng Kou, Shuyu Sun, Xiuhua Wang, A novel energy factorization approach for the diffuse-interface model with Peng–Robinson equation of state, SIAM J. Sci. Comput. 42 (1) (2020) B30–B56.
[34] Tereza Jindrovä, Jiří Mikyška, Fast and robust algorithm for calculation of two-phase equilibria at given volume, temperature, and moles, Fluid Phase Equilib. 353 (2013) 101–114.
[35] Tereza Jindrovä, Jiří Mikyška, General algorithm for multiphase equilibria calculation at given volume, temperature, and moles, Fluid Phase Equilib. 393 (2015) 7–25.
[36] Jiří Mikyška, Abbas Firoozabadi, A new thermodynamic function for phase-splitting at constant temperature, moles, and volume, AIChE J. 57 (2011) 1897–1904.
[37] Robert B. Schnabel, Elizabeth Eskow, A revised modified Cholesky factorization algorithm, SIAM J. Optim. 9 (4) (1999) 1135–1148.
[38] Yiteng Li, Jisheng Kou, Shuyu Sun, Thermodynamically stable two-phase equilibrium calculation of hydrocarbon mixtures with capillary pressure, Ind. Eng. Chem. Res. 57 (50) (2018) 17276–17288.
[39] Jiří Mikyška, Abbas Firoozabadi, Investigation of mixture stability at given volume, temperature, and number of moles, Fluid Phase Equilib. 321 (2012) 1–9.
[40] Jisheng Kou, Shuyu Sun, Xiuhua Wang, Linearly decoupled energy-stable numerical methods for multicomponent two-phase compressible flow, SIAM J. Numer. Anal. 56 (6) (2018) 3219–3248.
[41] Jordy Homing Lam, Yu Li, Lizhe Zhu, Ramzan Umarov, Hanlun Jiang, Amëlie Hëliou, Fu Kit Sheong, Tianyun Liu, Yongkang Long, Yunfei Li, Liang Fang, Russ B. Altman, Wei Chen, Xuhui Huang, Xin Gao, A deep learning framework to predict binding preference of RNA constituents on protein surface, Nature Commun. 10 (2019) 1.
[42] Xinshi Chen, Yu Li, Ramzan Umarov, Xin Gao, Le Song, Rna secondary structure prediction by learning unrolled algorithms, 2020, arXiv preprint arXiv:2002.05810.

[43] Tao Zhang, Shuyu Sun, A coupled Lattice Boltzmann approach to simulate gas flow and transport in shale reservoirs with dynamic sorption, Fuel 246 (2019) 196–203.
[44] Huangxin Chen, et al., Fully mass-conservative IMPES schemes for incompressible two-phase flow in porous media, Comput. Methods Appl. Mech. Engrg. 350 (2019) 641–663.
[45] Haijian Yang, et al., A fully implicit constraint-preserving simulator for the black oil model of petroleum reservoirs, J. Comput. Phys. 396 (2019) 347–363.