Supplementary Information "Molecular Dynamics Properties without the Full Trajectory: A Denoising Autoencoder Network for Properties of Simple Liquids"

Alireza Moradzadeh and N. R. Aluru

aluru@illinois.edu

Department of Mechanical Science and Engineering, Beckman Institute for Advanced Science and Technology, University of Illinois at Urbana−Champaign, Urbana, Illinois, 61801 United States

S.1. Training and Selection of Denoising Autoencoder Network

In order to select the deep neural network (DNN) architecture, we have tried various architectures with different nonlinearities and depths. Based on initial experiments, we have pursued full training of only six DNNs for almost 6,000,000 steps. The six DNNs are shown in Table S1. The loss function for Net 4 is smaller compared with the remaining DNNs, and it has a smaller overfitting as shown in Figure S1. The overfitting is measured as the distance of loss function value on validation and training dataset. Note that during training of deterministic networks, data usually gets divided into three sets, namely training, validation, and testing.[1] However, in the case of generative networks, data is usually divided into only two sets, training and validation. In this study, we use data from simulation of other simple liquids[2] as the test dataset and generalizability assessment.

Table S1. The DNNs trained for selecting the optimal network.

	Encoder	Latent Space	Decoder	Regularization factor
Net 1	$250 \times 200 \times 125$	75	$125 \times 200 \times 250$	10^{-5}
Net 2	$250 \times 200 \times 125$	75	$125 \times 200 \times 250$	5×10^{-5}
Net 3	$250 \times 200 \times 125$	75	$125 \times 200 \times 250$	10^{-4}
Net 4	$250 \times 200 \times 150$	100	$150 \times 200 \times 250$	10^{-5}
Net 5	$250 \times 200 \times 150$	100	$150 \times 200 \times 250$	5×10^{-5}
Net 6	$250 \times 200 \times 150$	100	$150 \times 200 \times 250$	10^{-4}

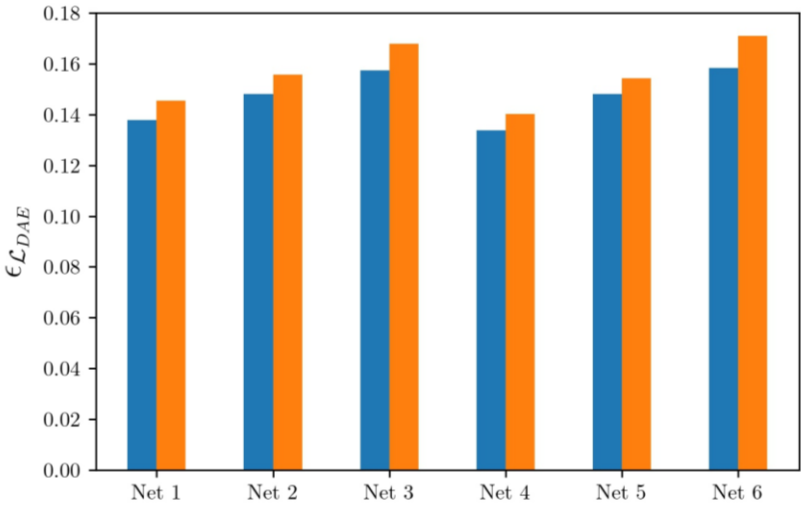


Figure S1. Loss function for six different DNNs trained for about 6,000,000 iterations. Net 4 is selected for obtaining the results in the main manuscript. Blue and orange boxes show loss function of training and validation data sets, respectively.

The loss functions decrease with training steps for the six DNNs as shown in Figure S2.

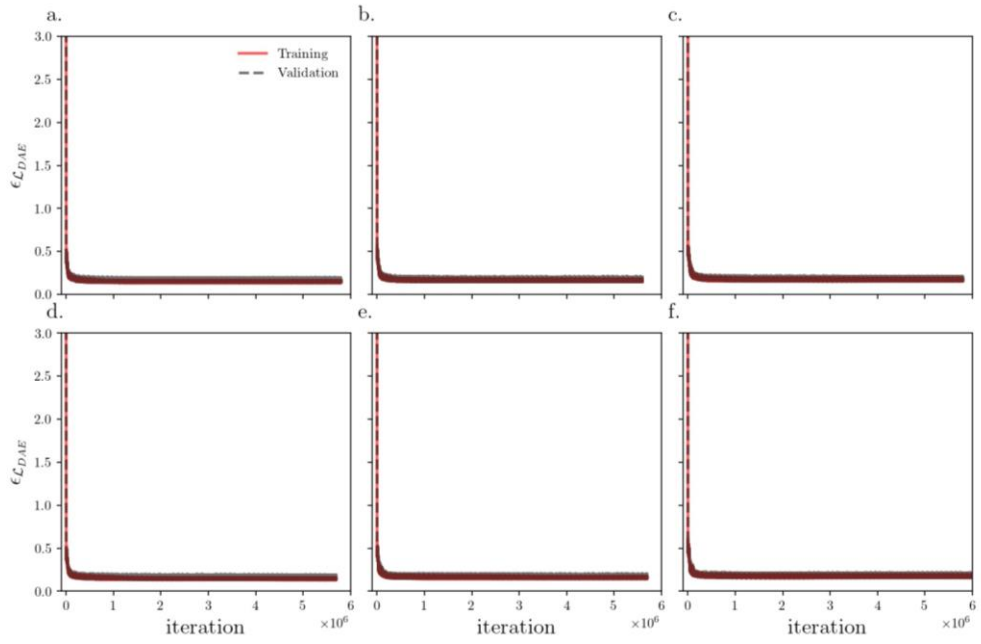


Figure S2. Loss function with training steps for six different DNNs. a. Net 1 b. Net 2 c. Net 3 d. Net 4 e. Net 5 f. Net 6.

We also invesigated the transferability of DAE network to unforeseen thermodynamic states. To do so, we feed RDF of various Lennard-Jones liquids into the frozen DAE network. We observe that the performance of DAE detoriates as we explore thermodynamic states far away from the dataset thermodynamic states (shown in Figure S3). However, we observe a clear correlation between peak radial distance between DAE prediction and MD ground truth RDFs.

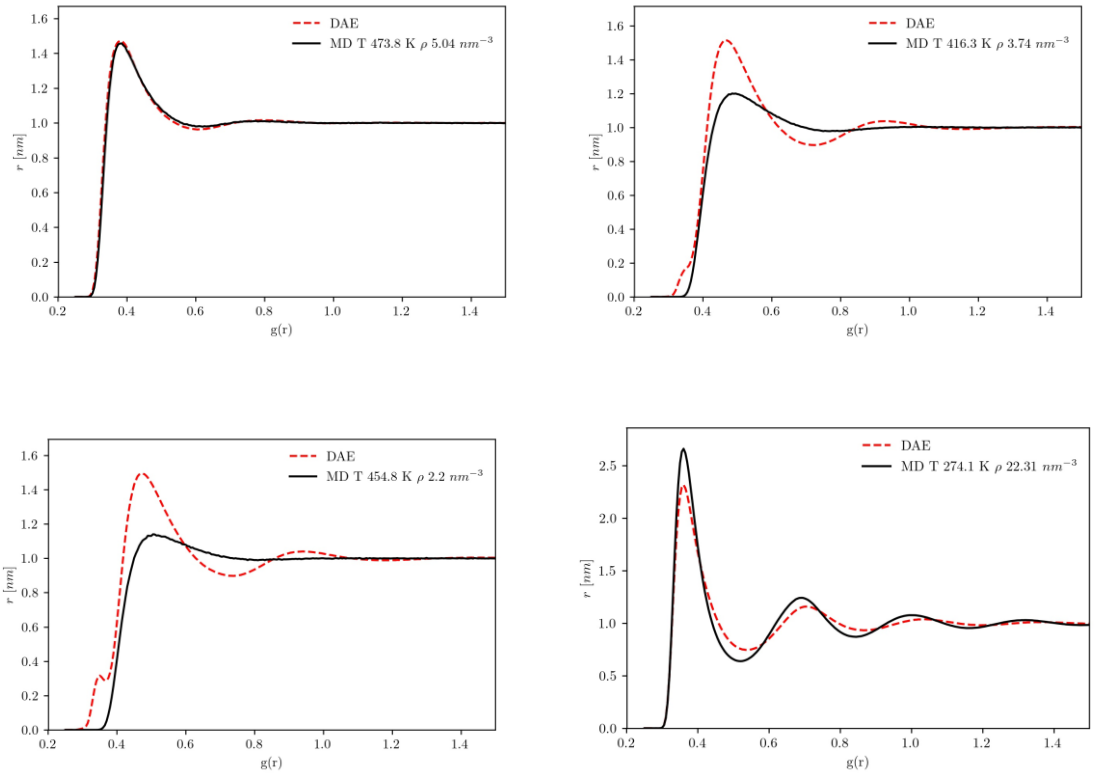


Figure S3. Transferability of DAE network for prediction of RDF for unforseen thermodynamic states.

S.2. Molecular Dynamics Details

The potential parameters and thermodynamic states of 6 randomly selected LJ systems studied in the main manuscirpt are given in Table S2.

Table S2. Potential parameters and thermodynamic states of LJ systems studied in Figures 2-3 of the main manuscript. a, b, c, d, e , and f refer to the results with the same alphabetic order.

System	Potential Parameters		Thermodynamic States	
	$C_{12}\ [\frac{kJ}{kmol\ nm^{12}}]$	$C_6\ [\frac{kJ}{kmol\ nm^{6}}]$	$T\ [K]$	$\rho\ [nm^{-3}]$
a	2.215×10^{-5}	3.06×10^{-3}	385.42	16.05
b	8.160×10^{-6}	7.26×10^{-3}	346.45	16.44
c	2.232×10^{-5}	2.03×10^{-3}	325.98	13.40
d	6.190×10^{-5}	7.59×10^{-3}	399.86	13.53
e	4.086×10^{-5}	3.27×10^{-3}	399.86	14.33
f	2.395×10^{-5}	6.74×10^{-3}	318.46	11.40

Potential parameters used in MD simulation of simple liquids are given in Table S3. The thermodynamic states of all the simple liquid systems are identical with temperature and density of 350 K and 12.23 nm^{-3}, respectively.

Table S3. Potential parameters of simple liquids studied in the main manuscript.

	Parameters	Values
Exponential Potential	$A\ [\frac{kJ}{kmol}]$	1.34×10^{6}
	$k_D\ [nm^{-1}]$	3.424×10^{1}
Yukawa Potential	$A\ [\frac{kJ}{kmol}]$	3.17×10^{7}
	$k_D\ [nm^{-1}]$	3.203×10^{1}
Inverse-Power-Law Potential (14-8)	$C_{14}\ [\frac{kJ}{kmol.nm^{14}}]$	4.142×10^{-6}
	$C_8\ [\frac{kJ}{kmol.nm^{8}}]$	6.077×10^{-5}
Inverse-Power-Law Potential (10-4)	$C_{10}\ [\frac{kJ}{kmol.nm^{10}}]$	2.500×10^{-4}
	$C_4[\frac{kJ}{kmol.nm^{8}}]$	8.521×10^{-2}

S.3. Comparison with PCA and ICA

Principal component analysis (PCA) and independent component analysis (ICA) are dimensionality reduction and denoising algorithms for high dimensional and large datasets.[3,4] In particular, PCA method is developed for data with Gaussian distribution, and recent work focuses on extending this for non-Gaussian data.[3] The denoising performance of PCA also requires independent and identically distributed (iid) Gaussian noise components. We assess validity of these conditions, namely Gaussian distribution and iid.[5] To do so, we test the departure from normality with calculation of p-value for null hypothesis of noise being drawn from Gaussian distribution with $\alpha = 0.05$ (the test combines skew and kurtosis to assess departure of normality; we use scipy stats package of python to perform this test).[6] We observe that at various radial distances the null hypothesis is rejected (shown below with red circles in Figure S4a). Figure S4b shows the noise distribution of argon Lennard-Jones liquids at a radial distance of 0.46 nm, which shows a non-Gaussian distribution, the non-Gaussian noise for MD data is also reported for other quantities in the literature[7].

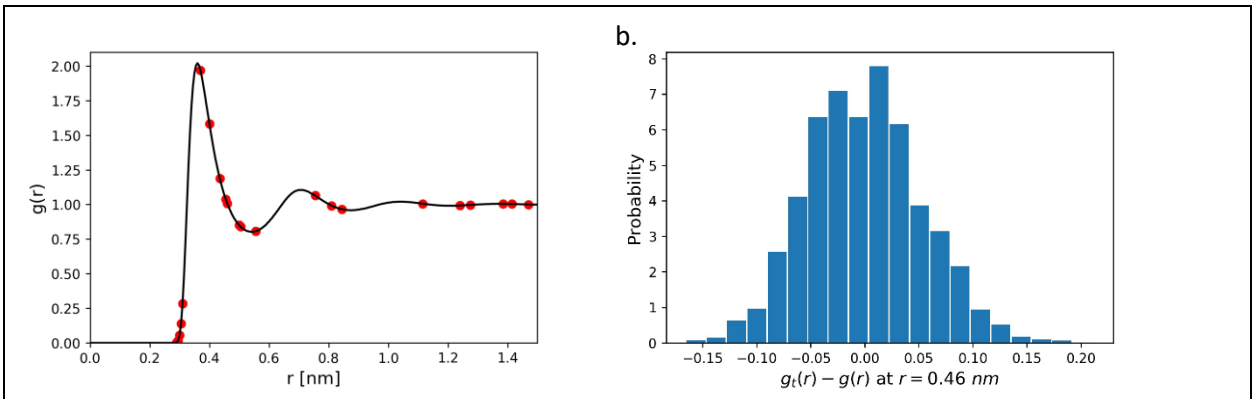


Figure S4. a. Non-Guassian noise at various radial distance b. noise distribution at a specific radial distance.

To check the iid condition of noise components, mutual information[8,9] between the noise component at each radial distance with the noise at other radial distance is calculated. The mutual information can be expressed as,

$$I(X,Y) = \sum_{x} \int \log\left(\frac{p(x,y)}{p(x)p(y)}\right) dy \qquad \text{S.1}$$

where (x, y) are the data points with underlying probability distribution $p(x, y)$. For iid data, mutual information is equal to zero, otherwise its non-zero. The maximum value of estimated mutual information at various radial distances is shown in Figure S5 (the test is performed using scikit-learn package of python).

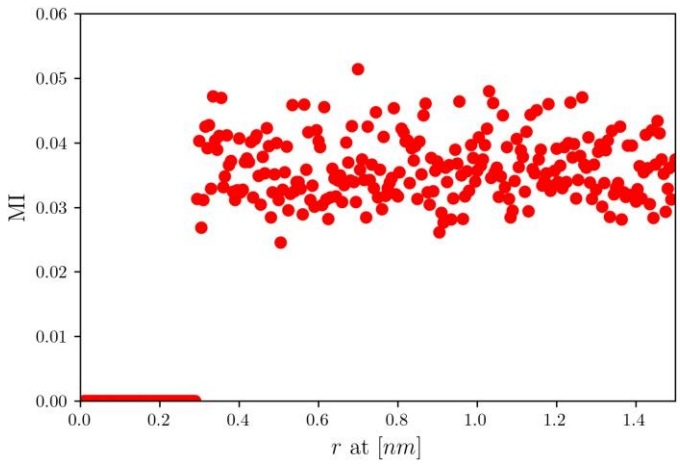


Figure S5. Mutual information at various radial distances. The maximum value of mutual information between noise at each radial distance with other radial distances is shown with circles. The non-zero circles show dependency with other points.

As shown in Figure S5 mutual information has a non-zero value, which indicates that the iid condition is not satisfied.

Even though the conditions for validity of PCA and ICA are not fully-satisfied, we compare performance of PCA and ICA with different number of components with the DAE method developed in the main manuscript. In general, we observe that the DAE method is better than both methods consistent with recent

studies reporting a better performance for denoising autoencoder compared with PCA for physical problems.[10,11] The main problem associated with PCA and ICA is unphysical negative values as these methods do not guarantee non-negative value for the non-negative dataset. We investigate both methods up to 19 components with dataset of 1500 RDFs. For any number of components, the DAE method outperforms both methods. RDFs of various systems reconstructed by PCA and ICA with number of components between 3 and 11 are shown in Figure S6-14. Similar trends are observed for the number of components between 12 and 18. Further increase in the number of components, in fact, deteriorates denoising performance of both PCA and ICA methods as shown in Figure S15 for 19 components.

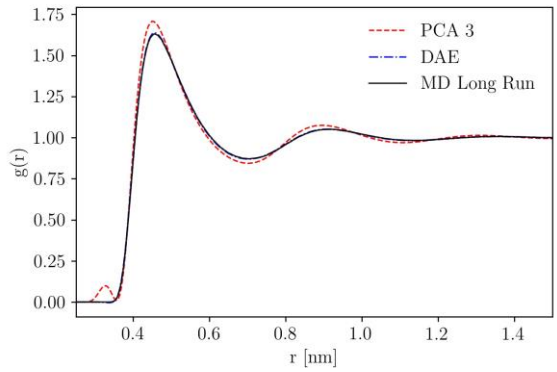
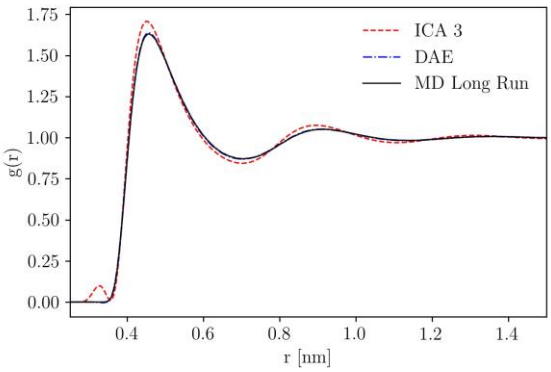
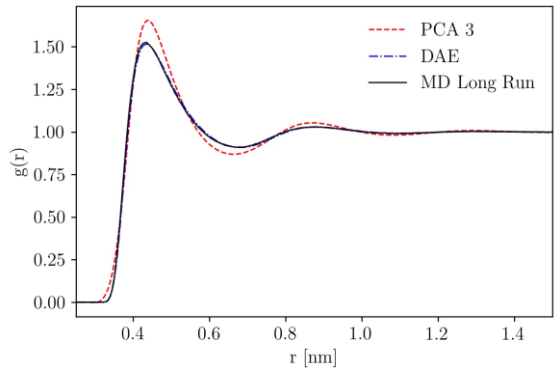
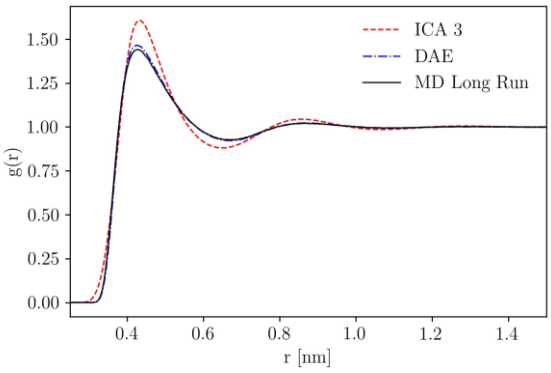
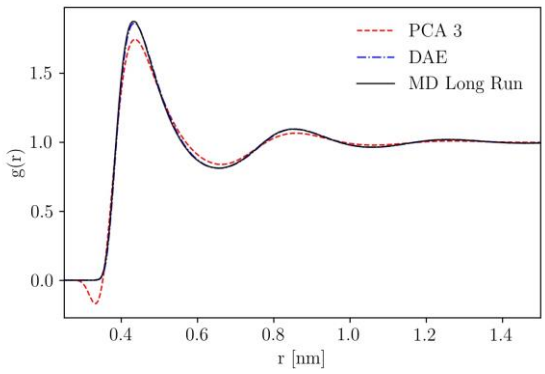
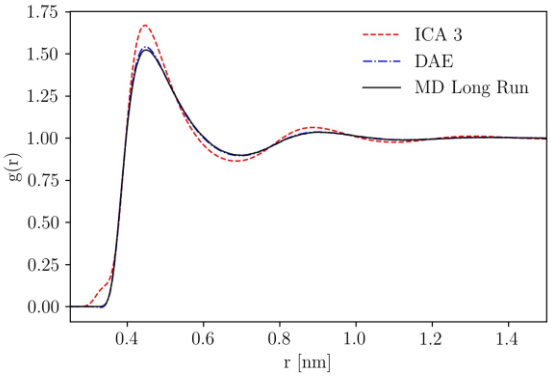
PCA with 3 components	ICA with 3 components
	
	
	

Figure S6. Comparison between DAE, PCA, and ICA (ICA and PCA with 3 components).

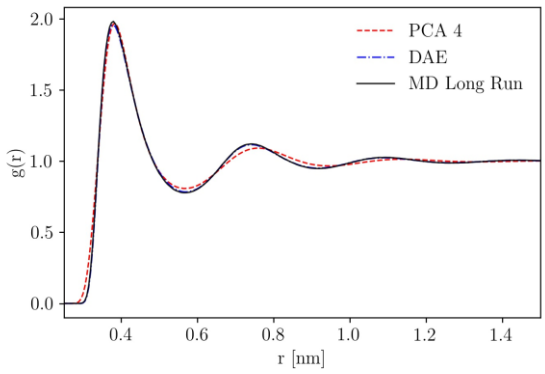
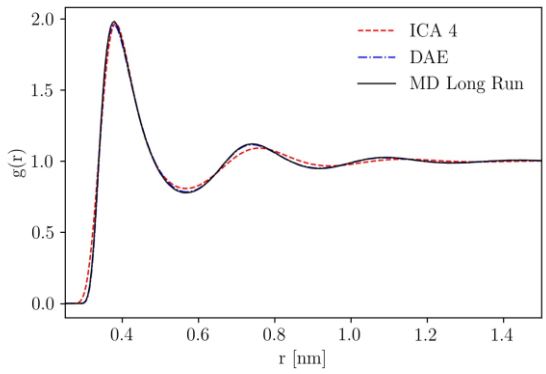
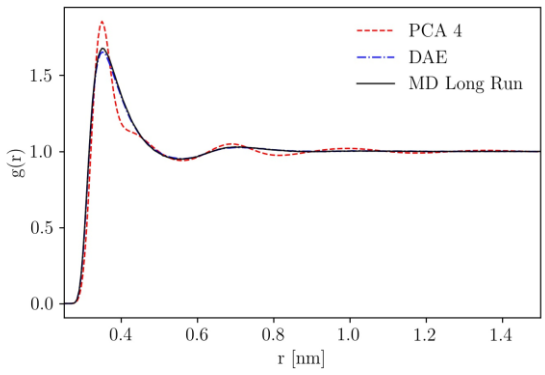
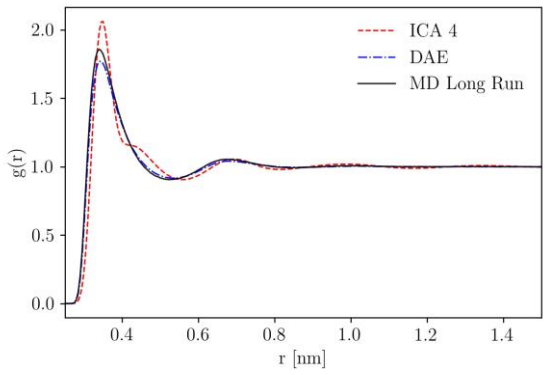
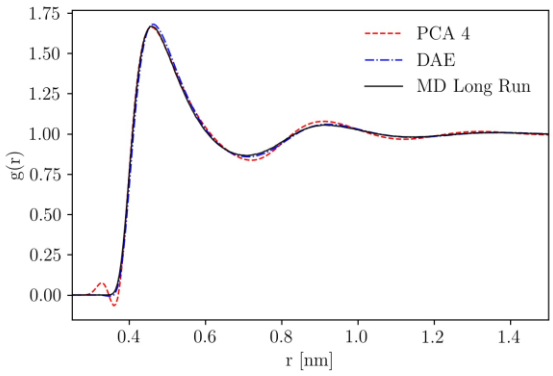
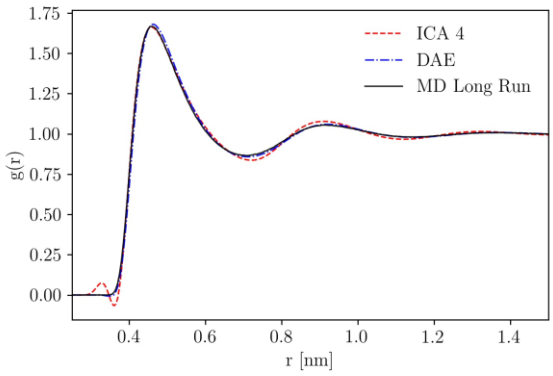
PCA with 4 components	ICA with 4 components
	
	
	

Figure S7. Comparison between DAE, PCA, and ICA (ICA and PCA with 4 components).

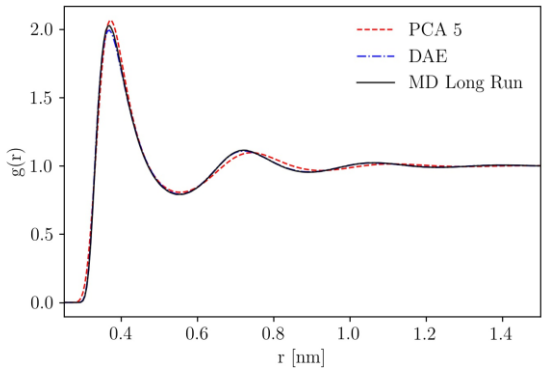
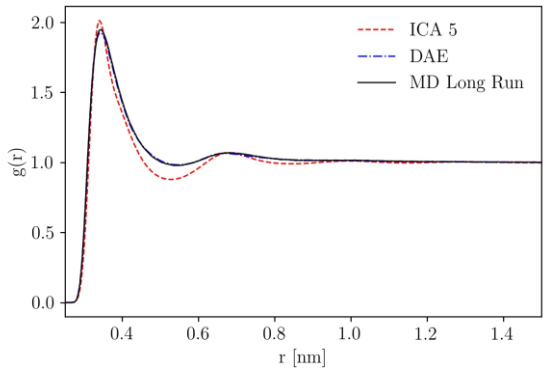
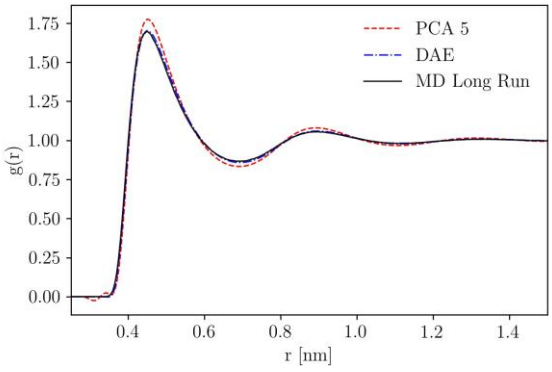
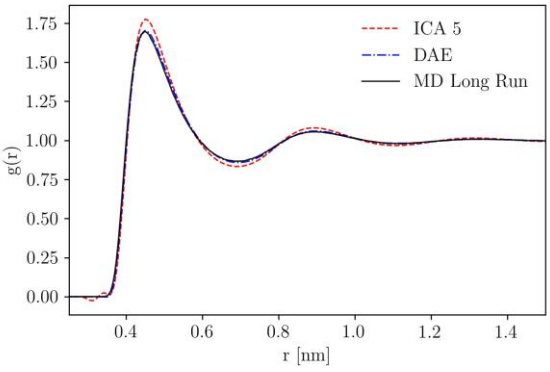
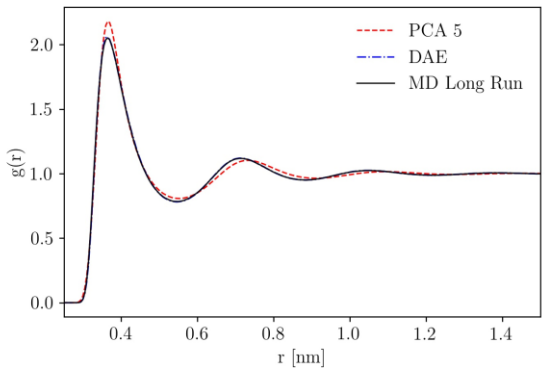
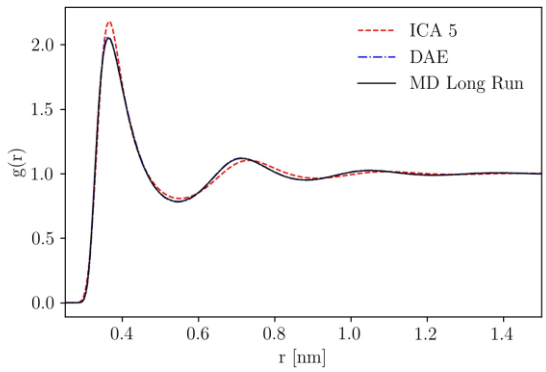
PCA with 5 components	ICA with 5 components
	
	
	

Figure S8. Comparison between DAE, PCA, and ICA (ICA and PCA with 5 components).

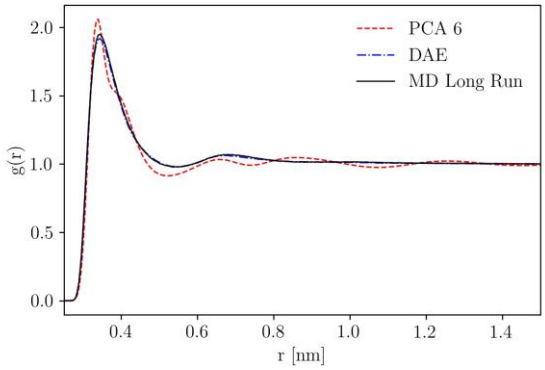
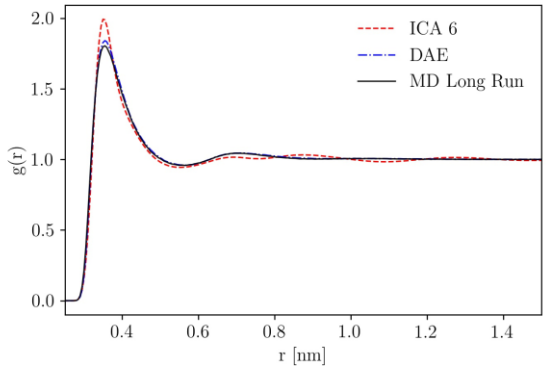
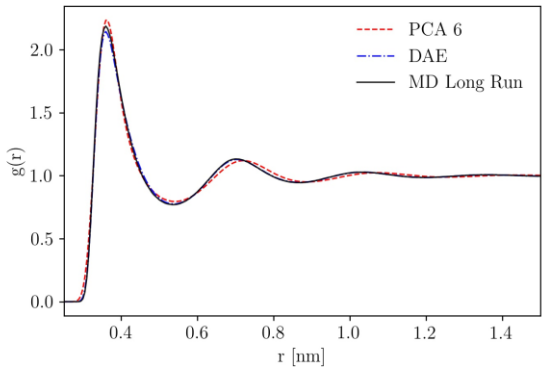
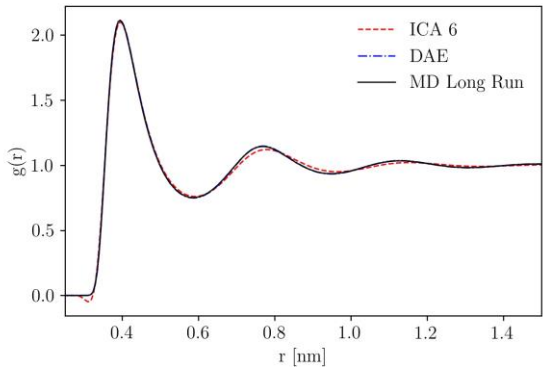
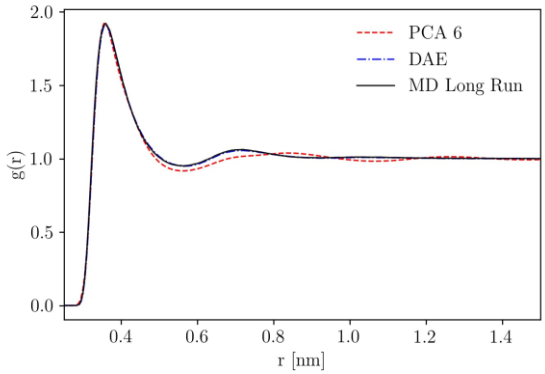
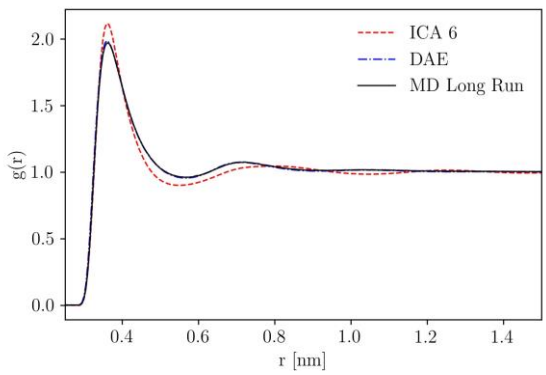
PCA with 6 components	ICA with 6 components
	
	
	

Figure S9. Comparison between DAE, PCA, and ICA (ICA and PCA with 6 components).

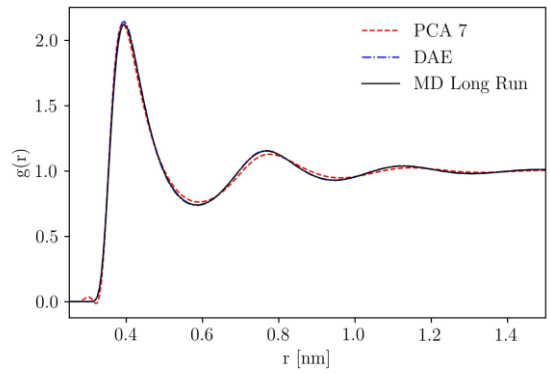
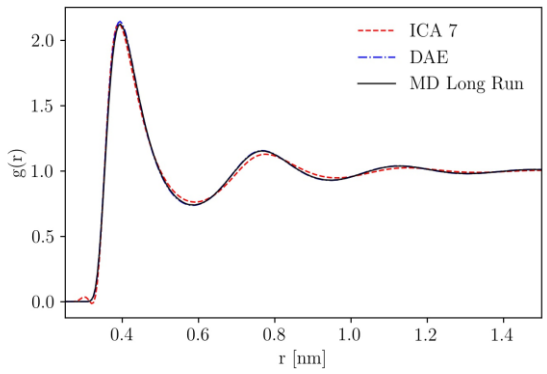
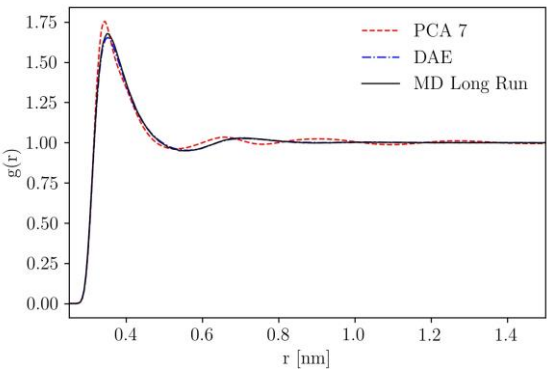
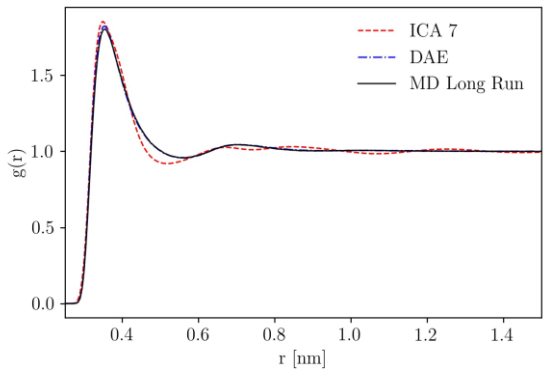
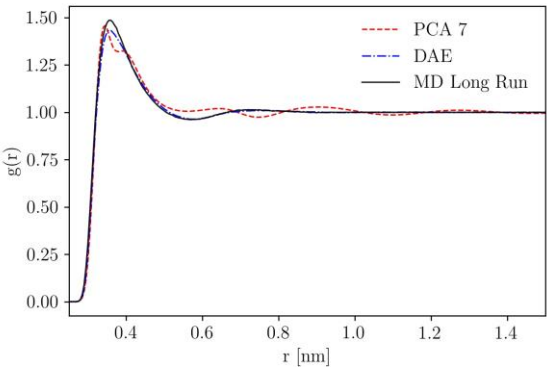
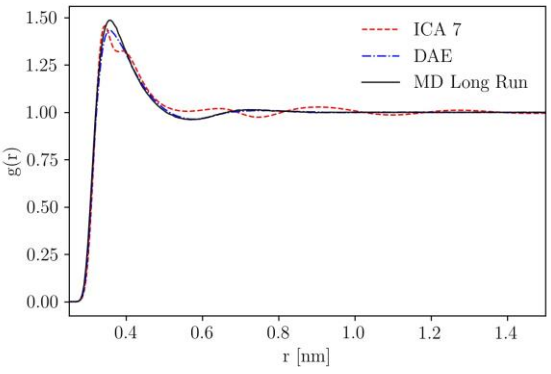
PCA with 7 components	ICA with 7 components
	
	
	

Figure S10. Comparison between DAE, PCA, and ICA (ICA and PCA with 7 components).

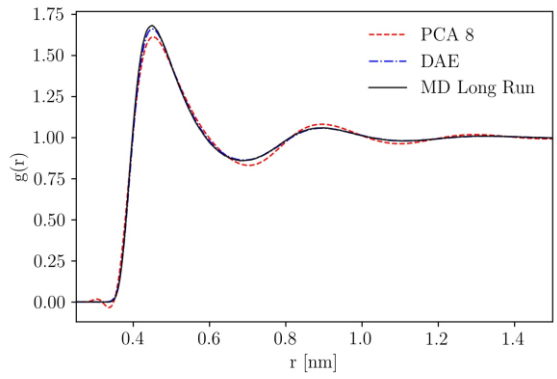
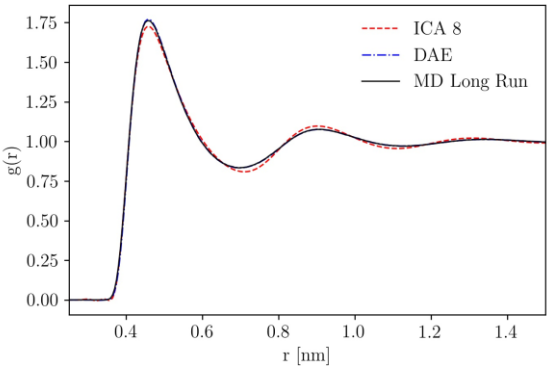
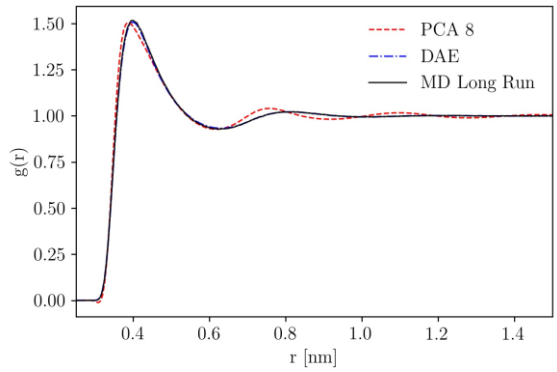
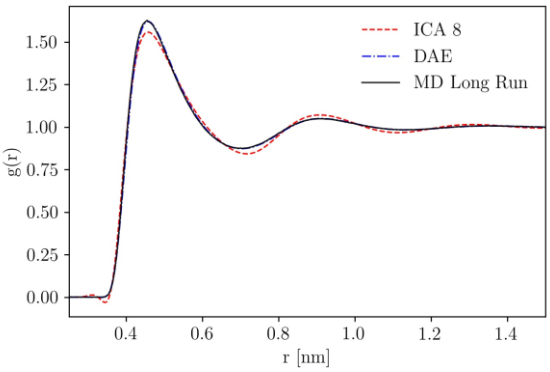
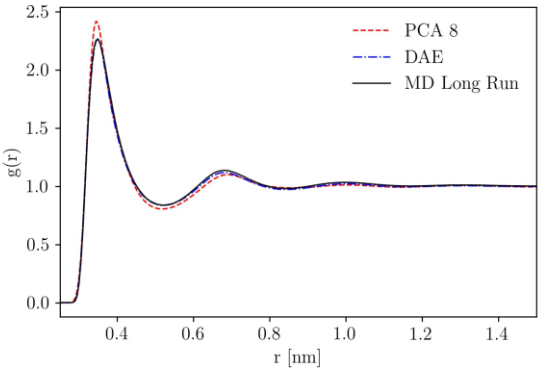
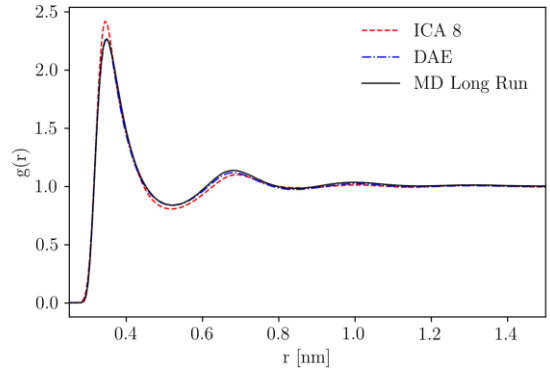
PCA with 8 components	ICA with 8 components
	
	
	

Figure S11. Comparison between DAE, PCA, and ICA (ICA and PCA with 8 components).

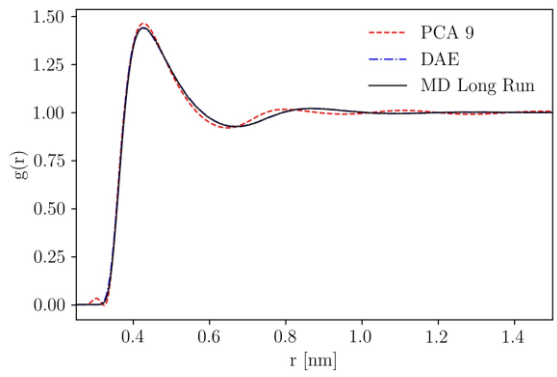
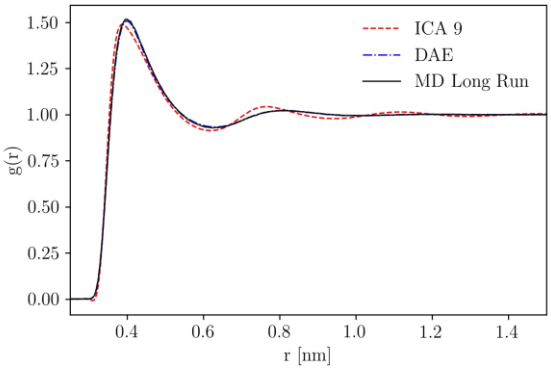
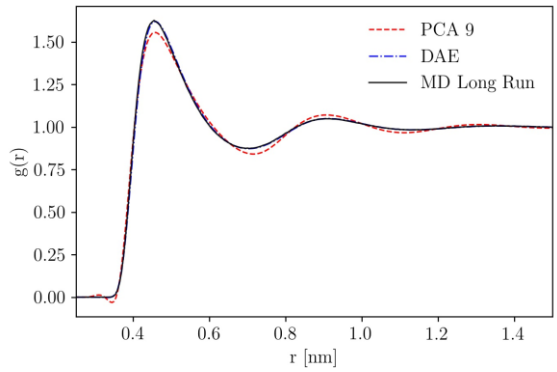
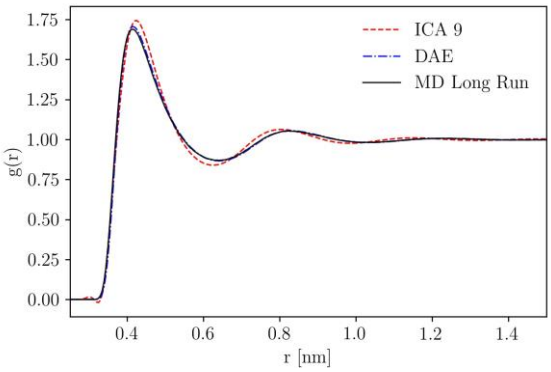
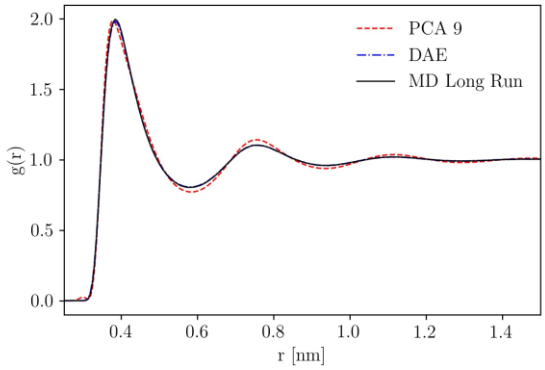
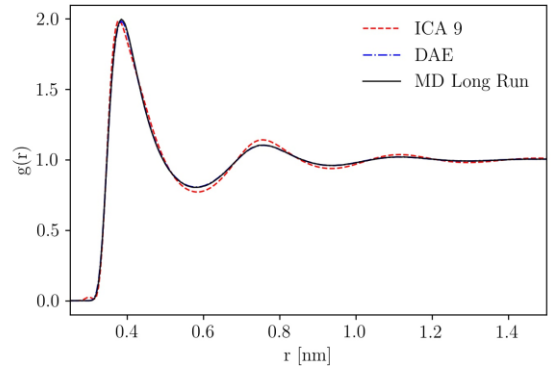
PCA with 9 components	ICA with 9 components
	
	
	

Figure S12. Comparison between DAE, PCA, and ICA (ICA and PCA with 9 components).

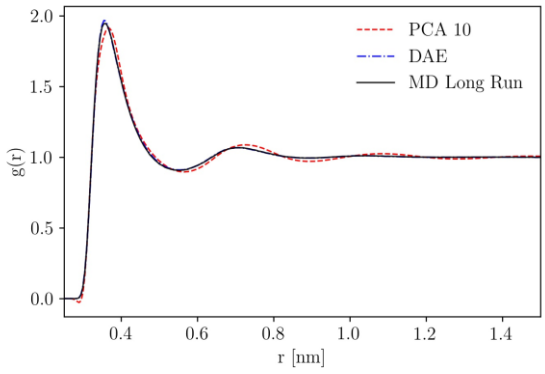
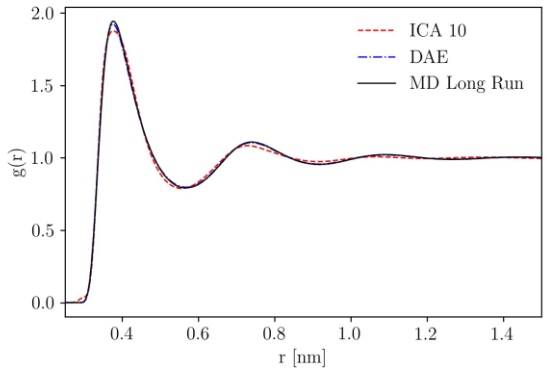
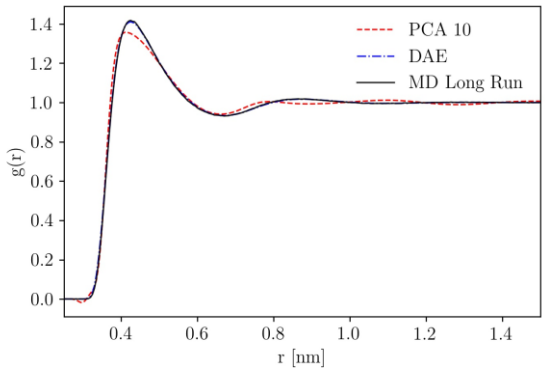
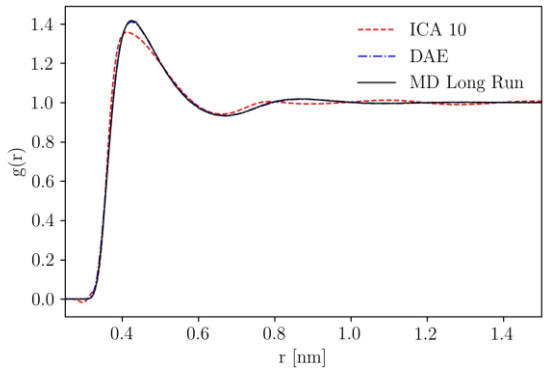
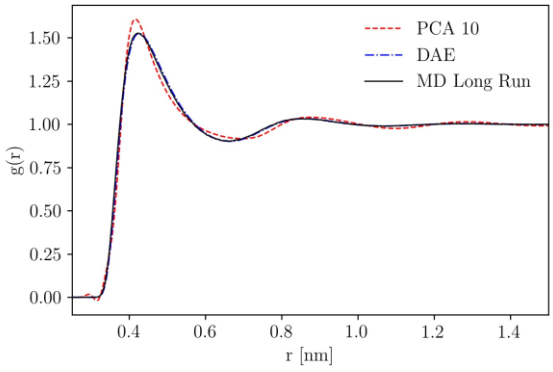
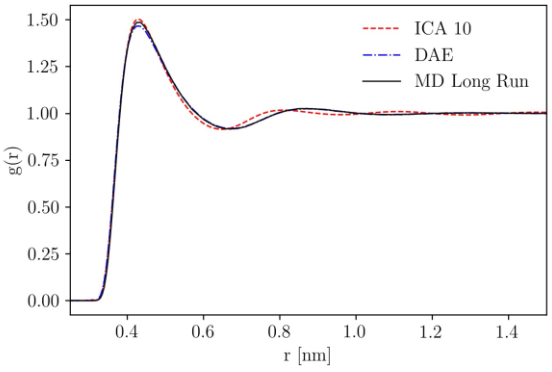
PCA with 10 components	ICA with 10 components
	
	
	

Figure S13. Comparison between DAE, PCA, and ICA (ICA and PCA with 10 components).

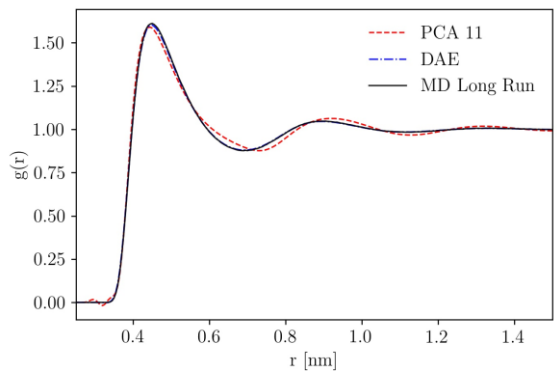
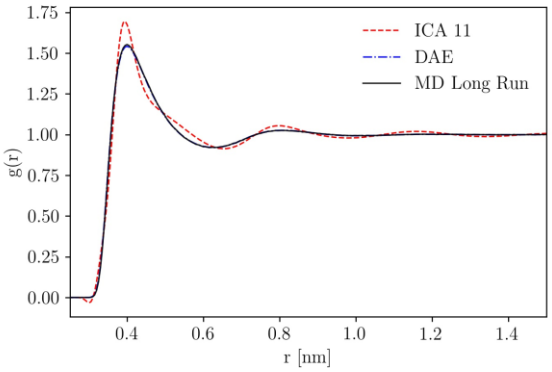
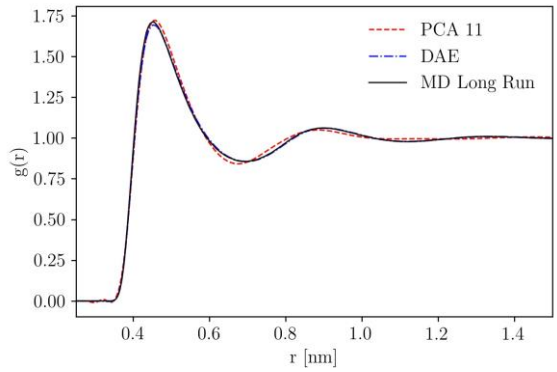
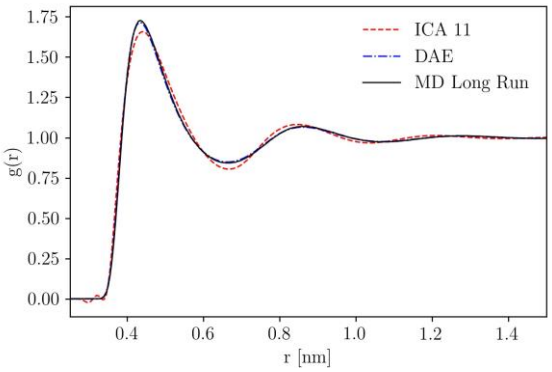
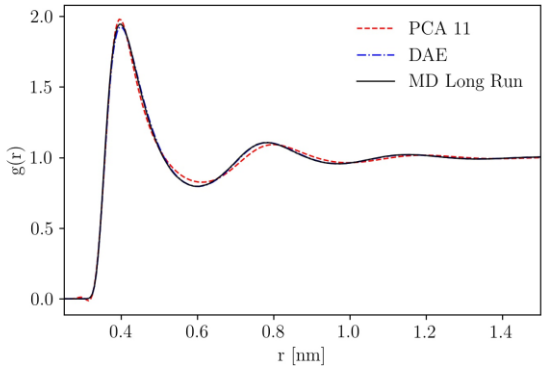
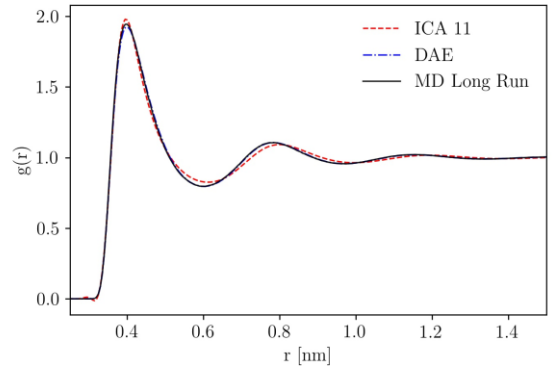
PCA with 11 components	ICA with 11 components
	
	
	

Figure S14. Comparison between DAE, PCA, and ICA (ICA and PCA with 11 components).

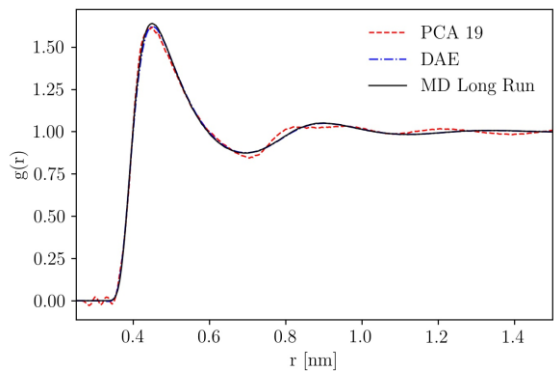
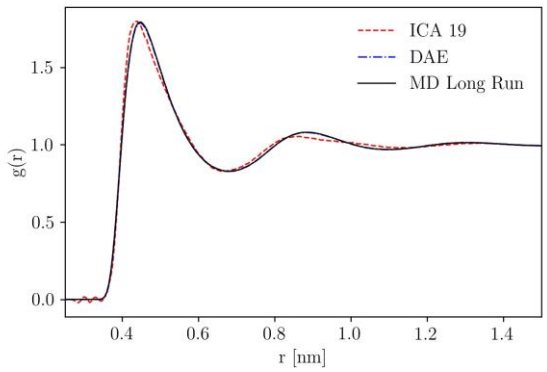
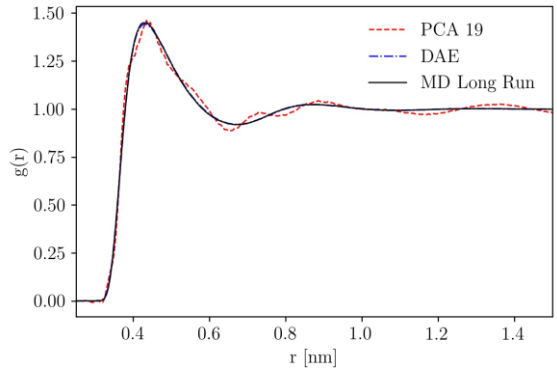
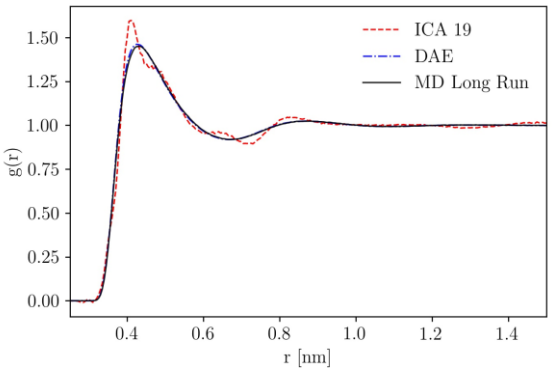
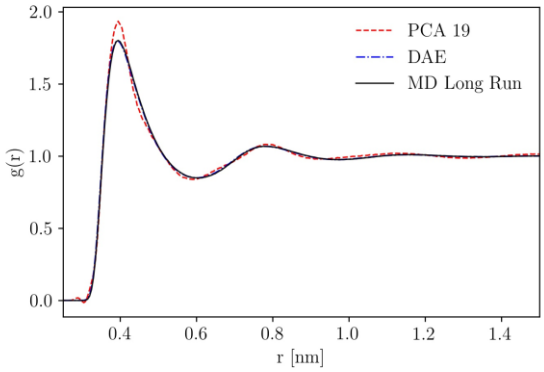
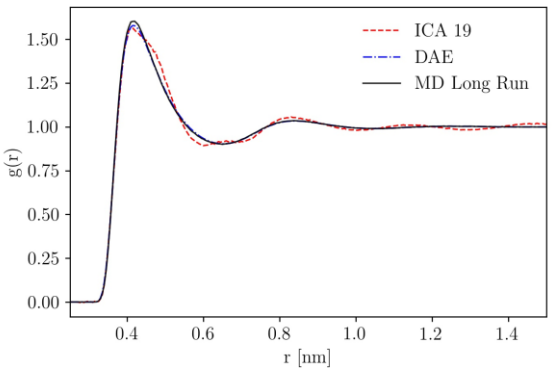
PCA with 19 components	ICA with 19 components
	
	
	

Figure S15. Comparison between DAE, PCA, and ICA (ICA and PCA with 19 components).

S.4. DAE with PCA Input-Output

We augment the denoising deficiency of PCA by using it as input and output of the DAE network. This methodology, in turn, reduces the size of the DAE network. To do so, we reduce dimensions of the feature vector (obtained thorough concatenation of temporally averaged RDF, temperature, and density – denoted as x) from d dimension into n components using the PCA,

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{W}_n \qquad (S.2)$$

where $\boldsymbol{X}$ is the input matrix with dimension of $|D| \times d$, where $|D|$ is the number of systems (*i.e.*,$\boldsymbol{X} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,d} \\ \vdots & \cdots & \vdots \\ x_{|D|,1} & \cdots & x_{|D|,d} \end{bmatrix}$), and $\boldsymbol{W}_n$ is the matrix of size $d \times n$. The matrix $\boldsymbol{W}_n$ is weights whose columns are eigenvectors of $\boldsymbol{X}^T\boldsymbol{X}$ with n largest eigenvectors. Having matrix $\boldsymbol{W}_n$, we transform input vector of all temporally averaged and single snapshot RDFs into reduced dimension. Once the data are transformed using PCA, we train DAE network to map PCA transformation of single snapshot RDFs into corresponding PCA transformation of temporally averaged RDF, analogous to training of DAE in the main manuscript for the RDFs. Once the DAE is trained, we apply inverse-transformation to increase dimension of DAE predicted PCA component into dimension of original data, *i.e.,* $\boldsymbol{X} = \boldsymbol{Y}\boldsymbol{W}_n^{\boldsymbol{T}}$ (see Figure S.16 for the workflow).

We employ PCA algorithm with 20 components, which show good ability in the dimensionality reduction of temporally averaged RDF, however, it comes at the cost of losing denoising functionality for a single snapshot RDF. After both the single snapshot and temporally averaged RDFs are dimensionally reduced with PCA, we train a small DAE network with two 2 hidden layers (encoder and decoder networks have two hidden layers with 20x15x10x15x20 architecture for DAE; we don't go through rigorous optimization of network architecture, as the main objective is to show how DAE network can be combined with PCA). Then, the DAE learns to map noisy PCA component of single snapshot RDF back to PCA components of temporally averaged RDF. The results shown in Figure S17 clearly indicate that DAE combined with PCA

outperform PCA denoising ability. The loss function of the training of DAE network with PCA input-output is also shown in Figure S18.

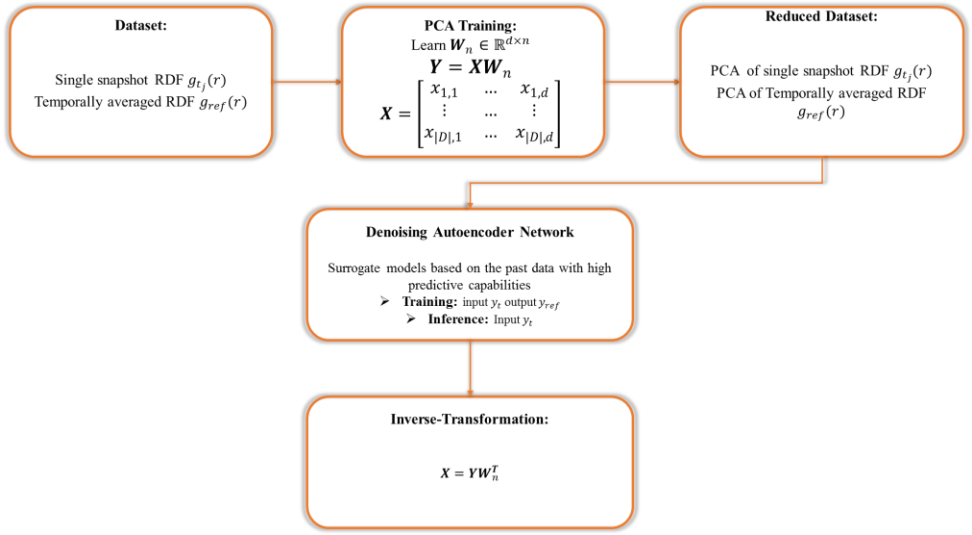


Figure S16. Workflow for using PCA as input and output of the DAE network.

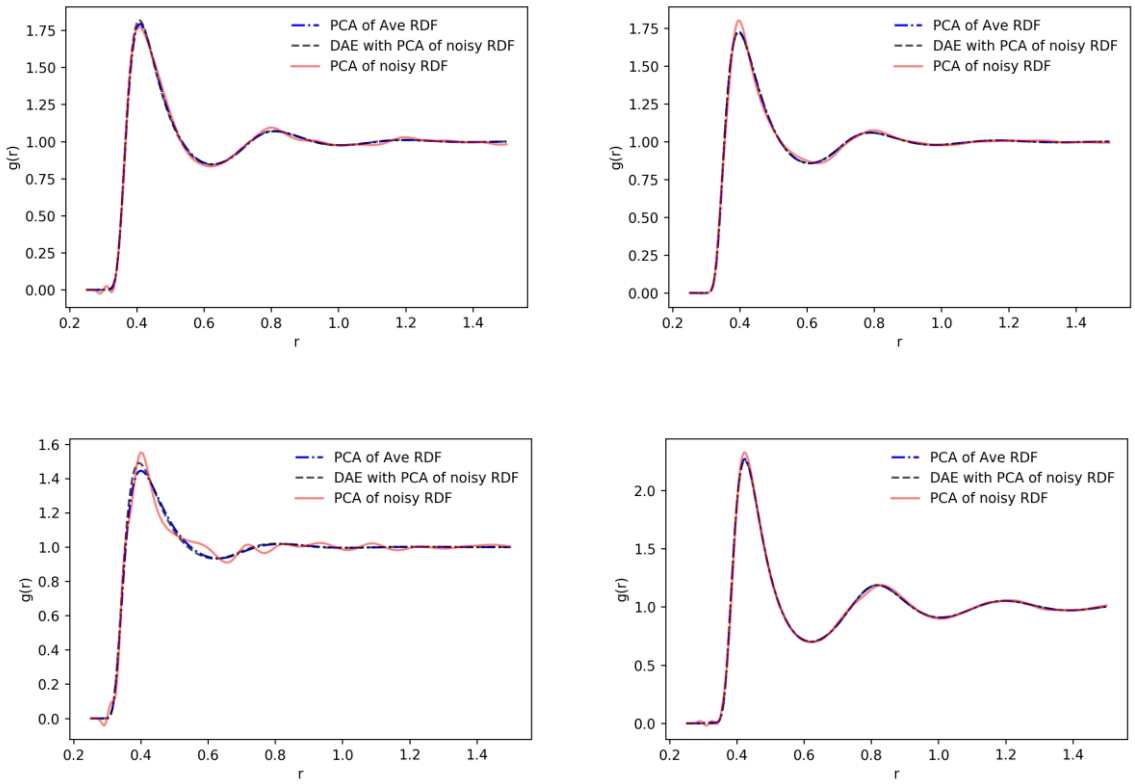


Figure S17. DAE with PCA input/output RDF prediction for four randomly selected systems. PCA with 20 components. The dash-dotted blue lines show PCA output for temporally averaged RDF. Solid red line shows the PCA of a single snapshot RDF, which is fed into DAE network to reproduce DAE predicted temporally averaged RDF shown with dash-dot blue lines.

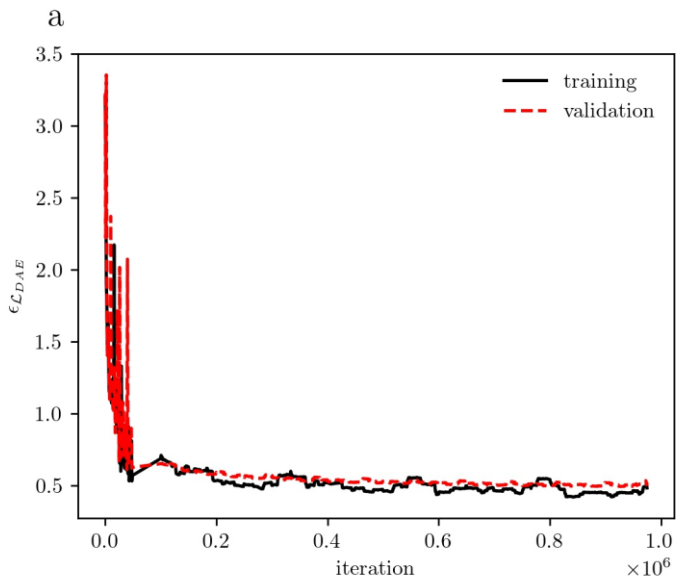


Figure S18. DAE with PCA input/output loss function during training steps.

References

(1) Moradzadeh, A.; Aluru, N. R. Transfer-Learning-Based Coarse-Graining Method for Simple Fluids: Toward Deep Inverse Liquid-State Theory. **2019**, *10* (6), 1242–1250. https://doi.org/10.1021/acs.jpclett.8b03872.

(2) Ingebrigtsen, T. S.; Schrøder, T. B.; Dyre, J. C. What Is a Simple Liquid? *Phys. Rev. X* **2012**, *2* (1), 011011. https://doi.org/10.1103/PhysRevX.2.011011.

(3) Jolliffe, I. *Principal Component Analysis*; John Wiley & Sons, Ltd: Chichester, UK, 2005. https://doi.org/10.1002/0470013192.bsa501.

(4) Anderson, T. W. (Theodore W. *An Introduction to Multivariate Statistical Analysis*; Wiley-Interscience, 2003.

(5) Geiger, B. C.; Kubin, G. Signal Enhancement as Minimization of Relevant Information Loss. In *Proceedings of 2013 9th International ITG Conference on Systems, Communication and Coding, SCC 2013*; 2013.

(6) PEARSON, E. S.; D'''AGOSTINO, R. B.; BOWMAN, K. O. Tests for Departure from Normality: Comparison of Powers. *Biometrika* **1977**, *64* (2), 231–246. https://doi.org/10.1093/biomet/64.2.231.

(7) Sanghi, T.; Aluru, N. R. Thermal Noise in Confined Fluids. *J. Chem. Phys.* **2014**. https://doi.org/10.1063/1.4900501.

(8) Ross, B. C. Mutual Information between Discrete and Continuous Data Sets. *PLoS One* **2014**, *9* (2), e87357. https://doi.org/10.1371/journal.pone.0087357.

(9) Kraskov, A.; Stögbauer, H.; Grassberger, P. Estimating Mutual Information. *Phys. Rev. E* **2004**, *69* (6), 066138. https://doi.org/10.1103/PhysRevE.69.066138.

(10) Shen, H.; George, D.; Huerta, E. A.; Zhao, Z. Denoising Gravitational Waves Using Deep Learning with Recurrent Denoising Autoencoders. **2017**.

(11) Hernández, C. X.; Wayment-Steele, H. K.; Sultan, M. M.; Husic, B. E.; Pande, V. S. Variational Encoding of Complex Dynamics. *Phys. Rev. E* **2018**, *97* (6), 062412. https://doi.org/10.1103/PhysRevE.97.062412.