Long Short-Term Memory in Chemistry Dynamics Simulation

Heng Wu, Shaofei Lu, Colmenares-Diaz Eduardo, Junbin Liang, Jingke She, and Xiaolin Tan

1 Introduction

Classical trajectory chemical dynamics simulations are widely and powerful tools that have been used to study reaction dynamics since the 1960s [1]. In contrast to the variational transition state theory (VTST) and reaction path Hamiltonian methods [2], they provide much greater insight into the dynamics of reactions for the classical equations of motion of the atoms are numerically integrated on a potential energy surface (PES). The traditional approach uses an analytic function that is gotten by fitting ab initio and/or experimental data [3] to construct the surface. Regarding a small number of atoms or a high degree of symmetry [4, 5], it is practical. Researchers recently proposed additional approaches and algorithms for representing PESs. Wang and Karplus firstly demonstrated that the trajectories may

H. Wu
Department of Mathematics and Computer Science, West Virginia State University, Institute, WV, USA
e-mail: heng.wu@wvstateu.edu

S. Lu (✉)
College of Computer Science and Electronic Engineering, Hunan University, Changsha, China
e-mail: sflu@hnu.edu.cn

C.-D. Eduardo
Department of Computer Science, Midwestern State University, Wichita Falls, USA
e-mail: eduardo.colmenares-diaz@msutexas.edu

J. Liang
School of Computer and Electronics Information, Guangxi University, Guilin, China

J. She · X. Tan
College of Computer Science and Electronic Engineering, Hunan University, Changsha, China
e-mail: shejingke@hnu.edu.cn; lxtanglx@hnu.edu.cn

H. R. Arabnia et al. (eds.), *Advances in Artificial Intelligence and Applied Cognitive Computing*, Transactions on Computational Science and Computational Intelligence, https://doi.org/10.1007/978-3-030-70296-0_9

be integrated "on the fly" when the potential energy and gradient are available at each point of the numerical integration according to an electronic structure theory calculation. During the numerical integration, the method directly calculates the local potential and gradient under an electronic structure theory in a "direct dynamics" simulation. However, regarding a high-level electronic structure theory, the computation of direct dynamics simulations become quite expensive. Thus, it is important to use the largest numerical integration step size when maintaining the accuracy of the trajectory. In order to use a larger integration step, Helgaker et al. adopt the second derivative of the potential (Hessian). After the Hessians are gotten directly by an electronic structure theory, using a second-order Taylor expansion, a local approximation PES can be constructed and the trajectories can be approximately calculated. For local quadratic potential is only valid in a small region (named a "trust radius"), the equations of motion are only integrated under the trust radius. The new potential, gradient, and Hessian, calculated again at the end of the trust radius, define a new local quadratic PES where the integration of the equations of motion is successive. Millam et al. used a fifth-order polynomial or a rational function to fit the potential between the potential, gradients, and Hessians at the beginning and end of each integration step. It provides a more accurate trajectory in the trust region and calculates larger integration steps. That involves a predictor step, the integration on the approximate quadratic model potential. The following step, the fitting on the fifth-order PES between the starting point and the end point in the trust radius, is also called the "corrector step." It is named the Hessian-based predictor–corrector integration scheme. Around it, some scholars proposed their own methods. Because of extrapolation, errors in prediction–correction algorithms grow rapidly, usually four predictions are followed by an ab initio calculation. This limits the improvement of computing performance.

The successful application of the prediction of deep learning in computational chemistry greatly expanded its application. Deep learning is a machine learning algorithm, not unlike those already in use in various applications in computational chemistry, from computer-aided drug design to materials property prediction [6]. Deep learning models achieved top positions in the Tox21 toxicity prediction challenge issued by NIH in 2014 [7]. Among some of its more high-profile achievements include the Merck activity prediction challenge in 2012, where a deep neural network not only won the competition and outperformed Merck's internal baseline model, but did so without having a single chemist or biologist in their team. Machine learning (ML) models also can be used to infer quantum mechanical (QM) expectation values of molecules, based on reference calculations across chemical space [8]. Such models can speed up predictions by several orders of magnitude, demonstrated for relevant molecular properties such as polarizabilities, electron correlation, and electronic excitations [9]. LSTM is an artificial recurrent neural network (RNN) architecture used in the field of deep learning. The prediction of LSTM has been widely used in different fields [10, 11].

In this chapter, we explore the idea of integrating LSTM layer with chemistry dynamics simulations to enhance the performance in trust radius. This idea is inspired by the recent development and use of LSTM in material simulations

and scientific software applications [12]. We employ a particular example, H_2O molecular dynamics simulation on NWChem/Venus (cdssim.chem.ttu.edu) package [13] to illustrate this idea. LSTM has been used to predict the energy, location, and Hessian of atoms. The results demonstrate that LSTM-based memory model, trained on data generated via these simulations, successfully learns preidentified key features associated with the energy, location, and Hessian of molecular system. The deep learning approach entirely bypasses simulations and generates predictions that are in excellent agreement with results obtained from explicit chemistry dynamics simulations. The results demonstrate that the performance gains of chemical computing can be enhanced using data-driven approaches such as deep learning which improves the usability of the simulation framework by enabling real-time engagement and anytime access.

This chapter is organized as follows. Section 2 presents the idea that integrate chemistry dynamics simulations with LSTM. Section 3 shows the experiment setting and results on H_2O molecular dynamics simulation, followed by data analysis. Section 3 presents the conclusions and lays out future work.

2 Methodology

2.1 Prediction–Correction Algorithm

In chemistry dynamics simulation, Hessian's calculation consumes most of the CPU time because Hessian is the third derivative of the position. Hessian updating is a technique frequently used to replace electronic structure calculations of the Hessian in optimization and dynamics simulations. Existing generally applicable Hessian-update schemes, for example, the symmetric rank one (SR1) scheme, Powell's symmetrization of Broyden's (PSB) method, the scheme of Bofill, the Broyden–Fletcher–Goldfarb–Shanno (BFGS) scheme, the scheme of Farkas and Schlegel, and other Hessian update schemes, are based on the Eq. (1)

$$H\,(X_{k+1})\;\;(X_{k+1}-X_k) = G\,(X_{k+1}) - G\,(X_k) \tag{1}$$

where $G(X)$ and $H(X)$, respectively, denote the gradient and Hessian of the potential energy at point X. Some researchers employed Hessian update method to build Hessian-based prediction–correction integration method to calculate the trajectory of atom in order to reduce the calculation time of Hessian and ab initio.

As illustrated in Fig. 1, in each time step of the integration method, the prediction is used to identify the direction the trajectory, ab initio potential energy, ab initio gradient, and ab initio or Hessian are computed at the end point $X_{i,p}$ of predicted trajectory. The potential information calculated at the end of predicted trajectory is used with the potential energy information at point $X_{i-1,p}$ near the trajectory starting point X_{i-1} of this time step, which is the end point of corrected trajectory of the

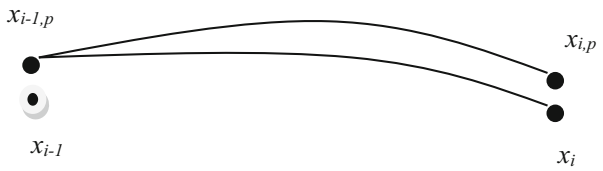


Fig. 1 During the i[th] step, the algorithm first predicts the trajectory from X_{i-1} to $X_{i,p}$ using potential approximated by the quadratic Taylor expansion about $X_{i-1,p}$. Then performs electronic structure calculation of the potential energy information at $X_{i,p}$ and reintegrate the trajectory from X_{i-1} to X_i using potential interpolated from ab initio potential information at $X_{i-1,p}$ and $X_{i,p}$

previous time step, to interpolate a highly accurate local PES. This highly accurate PES is used in the correction phase of the time step to recompute a more accurate trajectory.

In each time step, to obtain an accurate predicted trajectory, the prediction utilizes the Hessian in addition to the potential energy and its gradient. Assuming the current time step is the i*th* time step, the potential energy information needed during the prediction to integrate the trajectory is obtained by the quadratic expansion.

$$E(X) = E\left(X_{i-1,p}\right) + G\left(X_{i-1,p}\right)\left(X - X_{i-1,p}\right)$$
$$+1/2\left(X - X_{i-1,p}\right)^T H\left(X_{i-1,p}\right)\left(X - X_{i-1,p}\right) i > 2 \tag{2}$$

P is an integer. About the point $X_{i-1,p}$, the end point of the predicted trajectory of the (i–1)th time step at which ab initio potential energy $E(X_{i-1,p})$, ab initio gradient $G(X_{i-1,p})$, and ab initio or updated Hessian $H(X_{i-1,p})$ have been computed on a region within a trust radius from $X_{i-1,p}$.

If we use $X_{i-1,p}$ as the current location, the next part will show how to calculate the potential energy for the next X location. We can calculate the Potential Energy (P) and Gradient (G) at the $X_{i-1,p}$ from known position. For example, there are eight atoms in $F^- + CH_3OOH$. There are $3 \times N$ dimensions in the gradient and location vectors and N^2 dimensions in the Hessian matrix of the reaction system. Therefore, most of calculation of Eq. (2) is to compute $H(X_{i-1,p})$. The biggest challenge is to choice different approaches to fast the calculation of $H(X_{i-1,p})$ with the position and others of the current location, at the same time, cannot enlarge the system error.

2.2 Long Short-Term Memory

As shown in Fig. 2, a neural network is the connection of many single neurons, an output of a neuron can be an input of another neuron. Each single neuron has an activation function. The left layer of the neural network is called the input layer, it

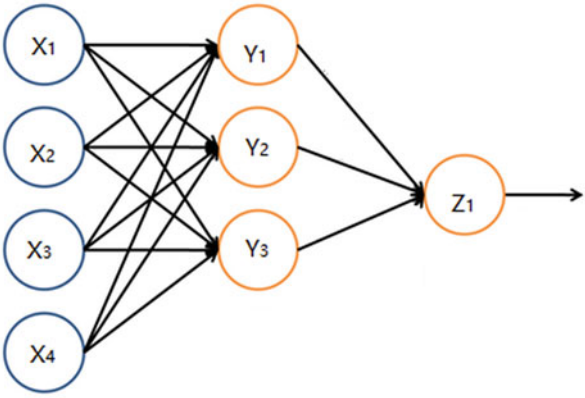


Fig. 2 The structure of a neural network

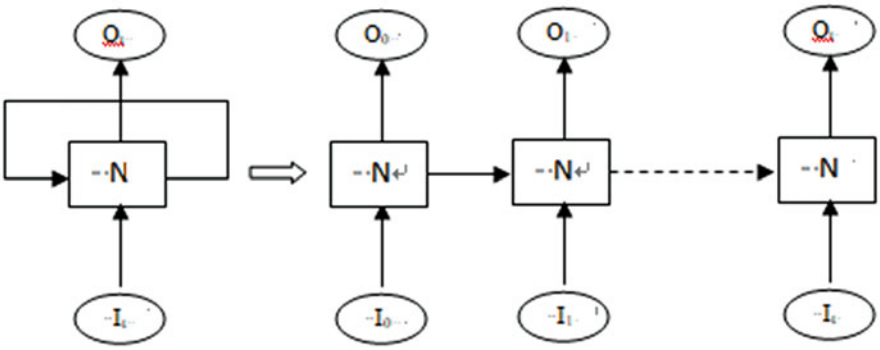


Fig. 3 The structure of a recurrent neural network and its unfolding

includes X1,X2,X3,X4, the right layer of it is output layer, involve Z1. The other layer is hidden layer, it covers Y1,Y2,Y3.

Recurrent neural network (RNN) is a typical kind of neural network. As shown in the leftmost part of Fig. 3.

Like the leftmost of Fig. 3, RNN is a neutral network containing loops. N is a node of neural network. I stands for input and O for output. Loops allow information to be transmitted from the current step to the next step. RNN can be regarded as a multiple assignment of the same neural network, and each neural network module transmits the message to the next one. The right side of Fig. 3 corresponds to the unfolding of the left side. The chain feature of RNN reveals that RNN is essentially related to sequences and lists. RNN applications have been successful in speech recognition, language modeling, translation, picture description, and this list is still growing. One of the key features of RNN is that they can be used to transmit the previous information to the current task. But the distance from previous step to related step is not too long.

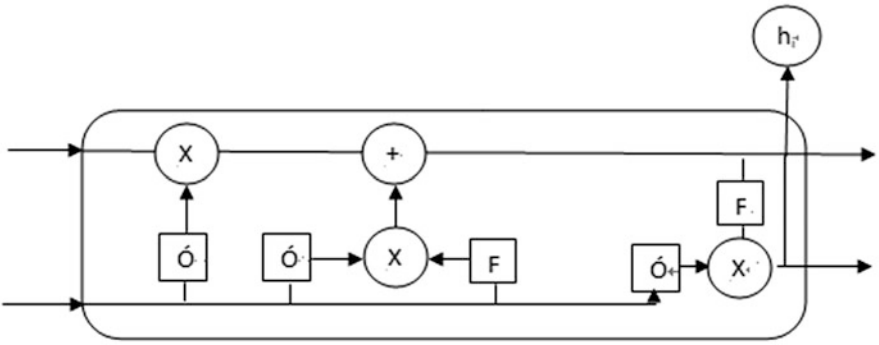


Fig. 4 The structure of a LSTM node

Long short-term memory (LSTM) overcomes this shortcoming. LSTM is a special type of RNN. LSTM solves the problem of long-term dependence of information. LSTM avoids long-term dependencies through deliberate design. Figure 4 shows the structure of a node in LSTM, where a forget gate can be observed. The output of the forget gate is "1" or "0." "1" means full reserve, "0" is abandon completely. The forget gate determines which information will be retained and what will be discarded. The upper horizontal line allows the input information cross neutral node without changing in Fig. 4. There are two types of gates in a LSTM node (input and output gates). The middle gate is an input gate, which determines the information to be saved in the natural node. F means function modular and create a new candidate value vector. The right gate is the output gate. The F module closed the output gate determines which information of the natural node will be transmit to the output gate.

A node has three gates and a cell unit as shown in Fig. 4. The gates use sigmoid as activation function, the tan*h* function is used to transfer from input to cell states. The following are to definite a node. For the gates, the function are

$$i_t = g\left(w_{xi}x_t + w_{hi}h_{t-1} + b_i\right) \tag{3}$$

$$f_t = g\left(w_{xf}x_t + w_{hf}h_{t-1} + b_f\right) \tag{4}$$

$$f_o = g\left(w_{xo}x_t + w_{of}h_{t-1} + b_o\right) \tag{5}$$

The transfer for input status is

$$c\_in_t = \tan h\left(w_{xc}x_t + w_{hc}h_{t-1} + b_{o\_in}\right) \tag{6}$$

The status is updated by

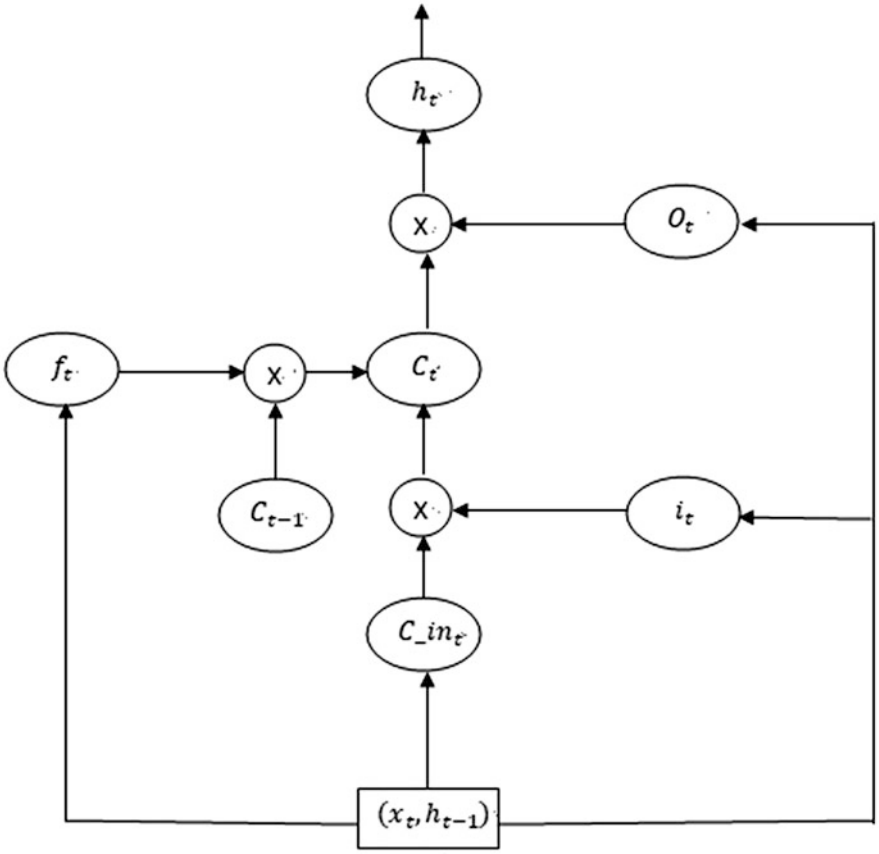


Fig. 5 The workflow of status changing of neutral node

$$c_t = f_t^* c_{t-1} + i_x^* c\_in_t \tag{7}$$

$$h_t = o_t{}^* \tan h\,(c_t) \tag{8}$$

The workflow of a node is shown in Figs. 5 and 6 shows the flowchart for LSTM.

2.3 Model

The calculation of position of the atom, the energy of the system, and Hessian occupy almost all the CPU time in chemistry dynamics simulations. Figure 7 illustrates the Hessian-based predictor–corrector algorithm in chemistry dynamics

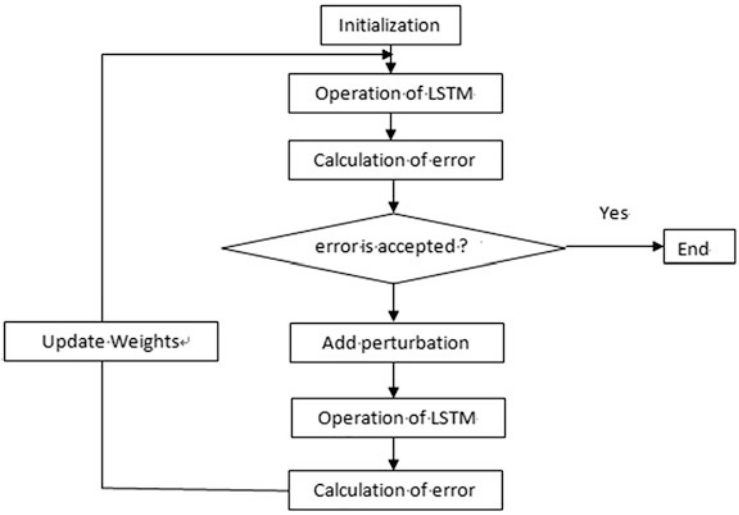


Fig. 6 The flowchart of LSTM

simulations. At each time step, the potential energy, kinetic energy, velocity, Hessian, and other parameters are calculated from the position of the atom. In Fig. 1, assuming $X_{i-1,p}$ is the current point, the calculation potential energy of next point X is as follows. The gradient and potential energy of the current point can be calculated from the known location of the point. Assuming eight atoms, the dimension of gradient and location will be $3 \times$ N, which of H will be N2. Hence, the largest calculation of Eq. (2) will be to calculate $H(X_{i-1,P})$. It is the focus of study of various algorithms to quickly and accurately calculate.

Algorithm 1 Algorithm

Input: Atomic initial parameters
Parameter: location of atoms, initial energy info
Output: the trajectory of atoms.
1. *ab initio* computing
2: **while** less than steps **do**
3: **while** less then training step **do**
4: exec Predictor-Corrector
5: train deep learning model
6: **end while**
7: predict the location,energy and Hessian
8: output trajectory
9: **end while**
10: **return**

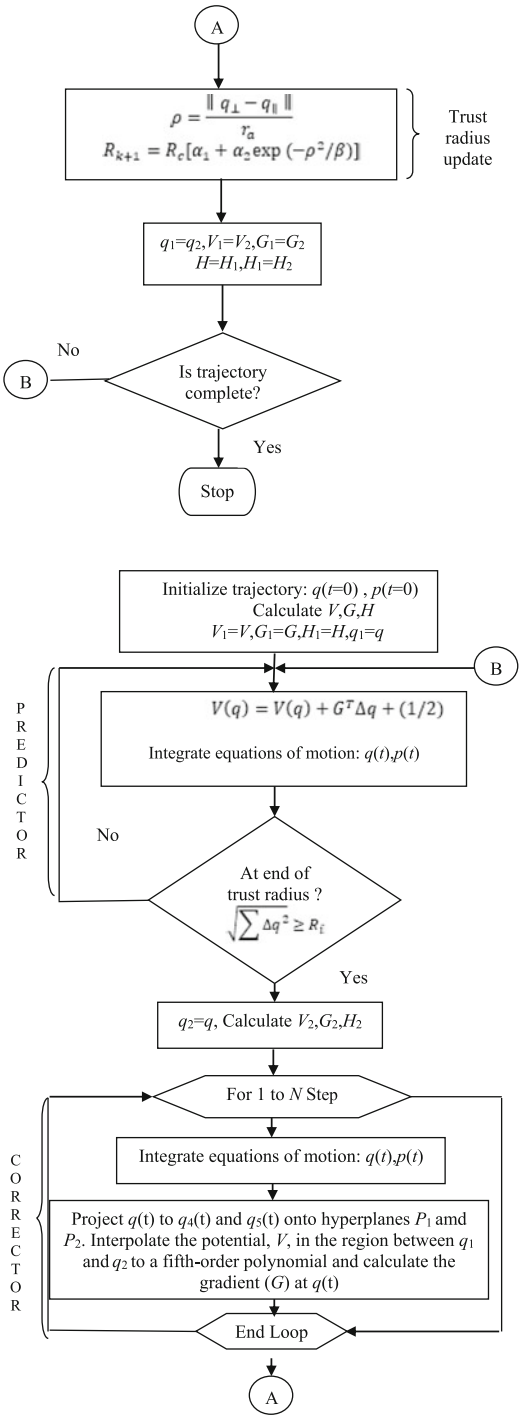


Fig. 7 Flowchart representation of the complete Hessian-based predictor–corrector integrator

H(Xi–1,p) according to the location and time information of the current point, simultaneously systematic error is required least. Researchers proposed some Hessian update methods to saving computing time [14]. Deep learning will be used to predict the location, energy, and Hessian of atoms. Therefore, deep leaning will be used three times to instead of predictor–corrector. It is important to understand that our deep learning model needs to be trained and initialized before predicting. The result of this approach is a novel predictor–corrector algorithm with deep learning.

3 Experimental Results

To test the algorithm with deep learning, we implemented the integration algorithm in the VENUS (cdssim.chem.ttu.edu) dynamics simulation package interfaced with the electronic structure calculation NWChem [13]. We chose the reaction system H_2O as our testing problem. In the tests, ab initio potential energy, gradient, and Hessian were calculated using the density function theory 6–311 + G**, and ab initio Hessian is calculated once in every five steps during training. In the remaining nine steps, the new update scheme is used. We calculated a trajectory for the chemical reaction system with 5000*0.67 integration steps, where each step has a fixed size of 0.02418884 fs (100 a.u.; 1 a.u. = 2.418884e-17 s). The remaining step 5000*0.33 steps were predicted by the proposed deep learning algorithm. There are three prediction parameters in our test. They are atomic position, energy, and Hessian, respectively.

Figure 8 illustrates the computational energy and its predicted values. The above is the H_2O system computational energy chart. The horizontal coordinate is the time step and the vertical one is the energy value. The yellow region represents training data and green section predicted values. After more than 3000 training steps, the predicted value is almost the same as the calculated values. Table 3 lists some relative errors. We find the relative error to be less than 0.1%.

Figure 9 shows a hydrogen atomic location chat. The above is the computational values and the following is training and predicted values. The horizontal coordinate is the time step and the vertical one is the atomic location. Table 1 has some relative error between predicted and computational values. We find the relative error less than 0.7% and some even less than 0.01%. Figure 10 is one of Hessian chat. The

Table 1 Relative error between atomic position prediction and computational value

Computational data (D1)	Predicted data D2	|(D1–D2)|/|D1| (%)
−0.65278023	−0.6570433	0.65%
−0.65459454	−0.6573254	0.4%
−0.6577446	−0.6590448	0.2%
−0.662028	−0.66203827	<0.01%
−0.6672311	−0.6661249	0.16%

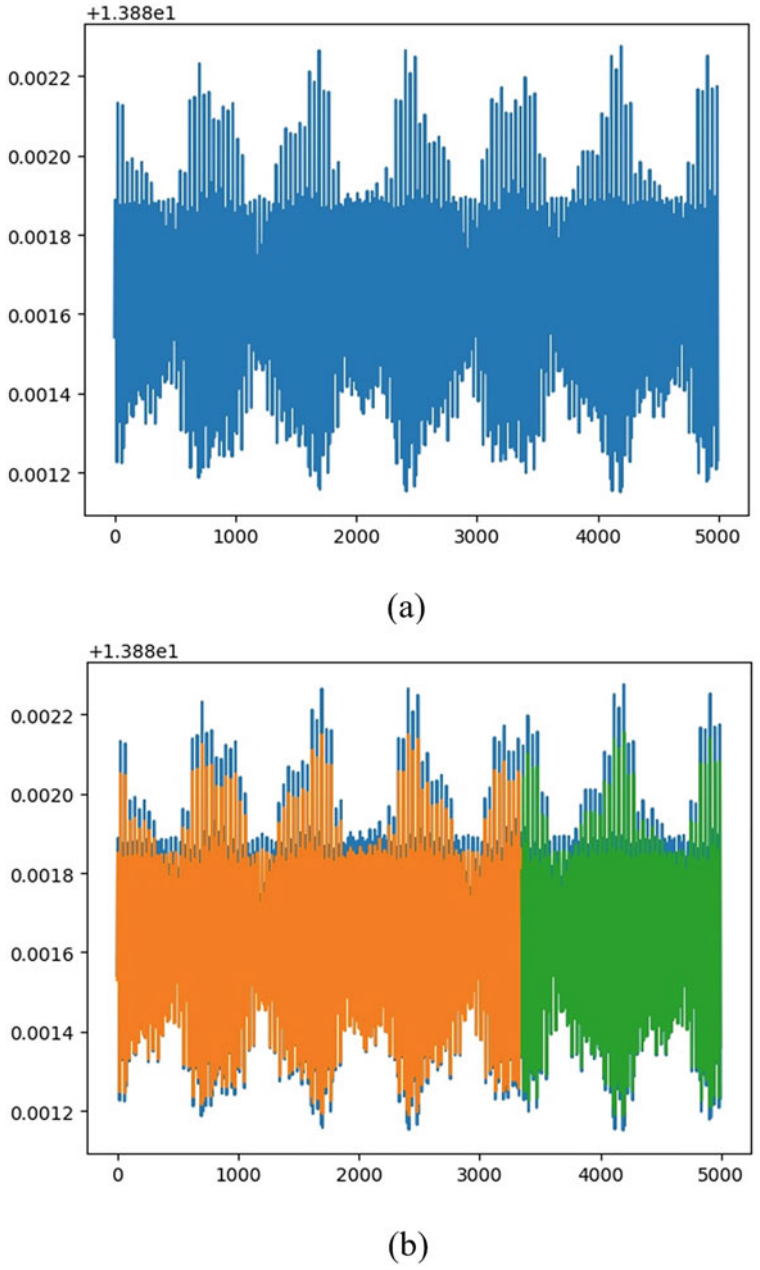


Fig. 8 The energy of H_2O system: (**a**) Output of the prediction–correction algorithm. (**b**) The yellow region corresponds to training data and green is prediction data

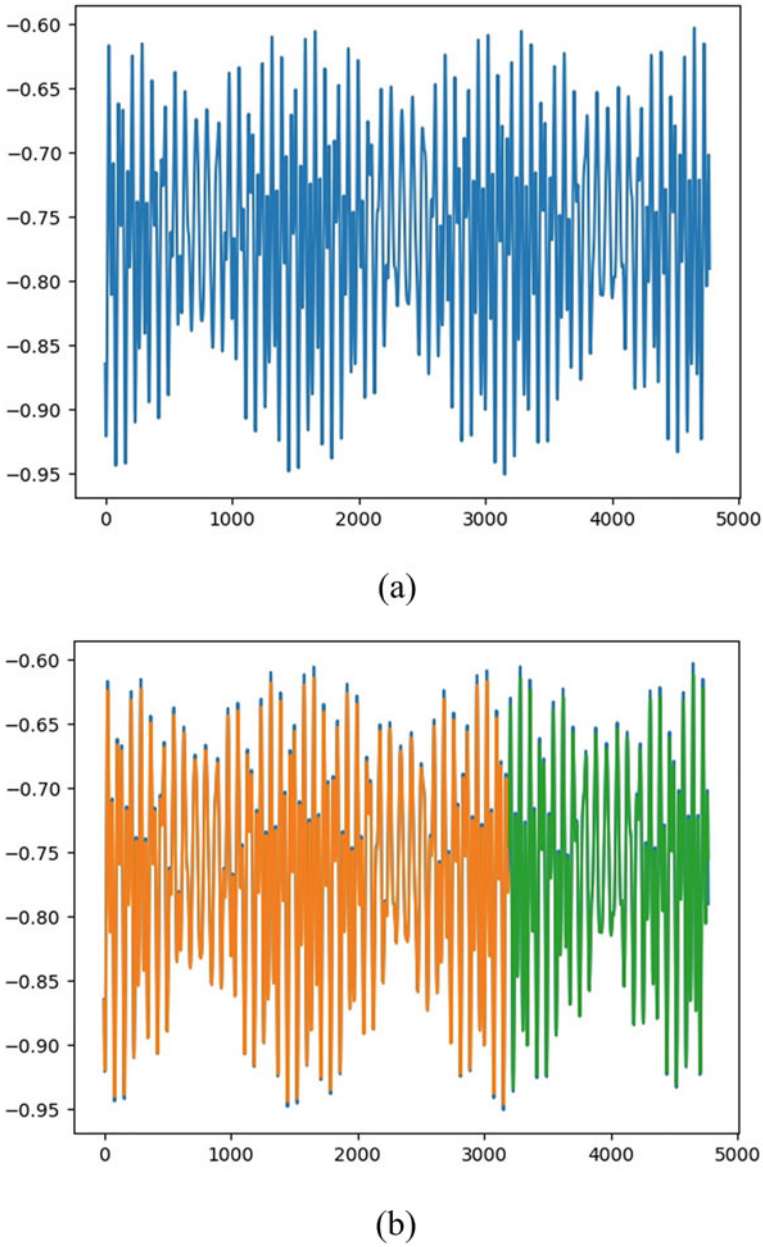
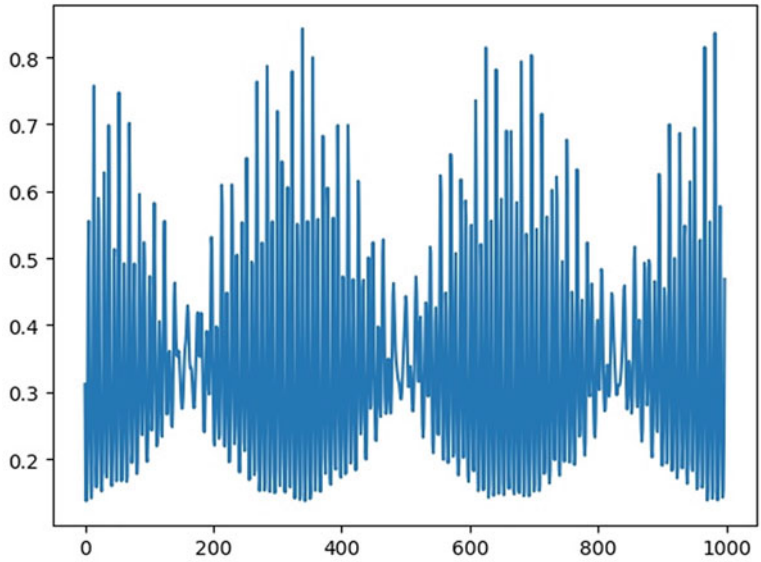
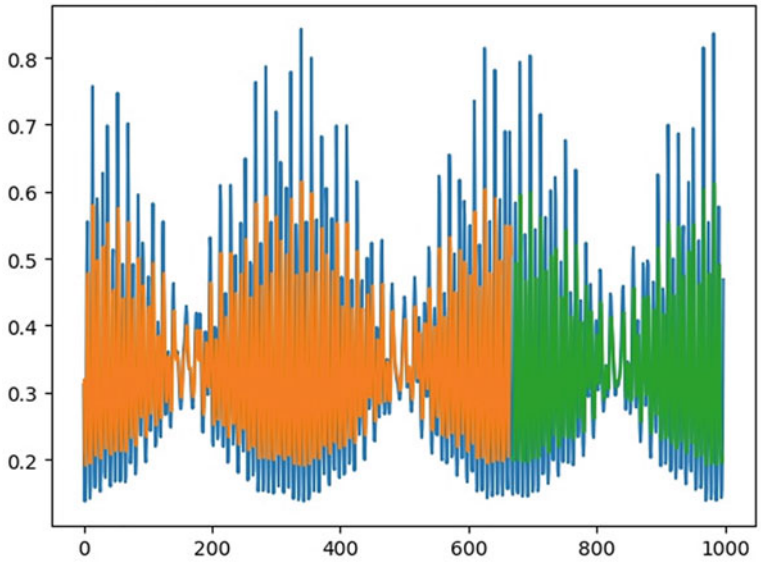


Fig. 9 The location of atoms in H_2O system: (**a**) Output for the prediction–correction algorithm. (**b**) The yellow region corresponds to is training data and green represents prediction data



(a)



(b)

Fig. 10 The Hessian of H_2O system: (**a**) Output of the prediction–correction algorithm; the yellow part is training data and great is prediction data in (**b**)

Table 2 Relative error between Hessian prediction and computational value

Computational data (D1)	Predicted data D2	|(D1–D2)|/|D1| (%)
0.16278428	0.20113048	23%
0.24653251	0.2093605	15%
0.43518633	0.47104657	8%
0.15830526	0.20171289	27%
0.23584451	0.20607977	12.6%

Table 3 Relative error between system energy prediction and computational value

Computational data (D1)	Predicted data D2	|(D1–D2)|/|D1| (%)
13.881894	13.881851	<0.01%
13.881888	13.881856	<0.01%
13.881871	13.881851	<0.01%
13.881843	13.881836	<0.01%
13.881803	13.88181	<0.01%

above are the computational values and the following are training and predicted values. The horizontal coordinate is the time step and the vertical one is Hessian value. Table 2 has some relative error between predicted and computational values. We find the minimum relative error is 8% and some even over 25%. Although it is 5000 steps, Hessian calculated only 1000 steps because of the predictive–correction algorithm. Therefore, the size of the training set is less than 670 and the relative error is relatively large (Table 3).

The prediction–correction algorithm can reduce H_2O reaction system dynamics simulation time from months to days. The stability of the prediction–correction algorithm becomes very weak as simulation goes on. In addition, there must be an ab initio calculation every few steps in the prediction–correction algorithm. As the prediction step increases, the stability becomes weaker. Deep learning can reduce the simulation time of the reaction system by one third. The prediction step can reach over 1200 steps without affecting the system error after enough training. If some reinforcement learning and other methods are used, the calculation time will be further reduced and the prediction steps will be more.

4 Conclusion and Future WORK

In this chapter, a new molecular dynamics simulation algorithm is proposed by combining deep learning and predictive–correction algorithms. The new algorithm can reduce the calculation time of the system by one-third without increasing the error. In the future, the enhanced learning and parameter migration will be used to further reduce the calculation time. Then monodromy matrix [15–17] will be used to monitor the change of the calculation error.

Acknowledgments This work was supported by Dr. Hase research group and Chemdynm cluster at Texas Tech University, as well as the Industrial Internet Innovation and Development Project of China: Digital twin system for automobile welding and casting production lines and its application demonstration (TC9084DY).

References

1. D.L. Bunker, Classical trajectory methods. Comput. Phys., 10, 287–324 (1971, 1971)
2. J.M. Millam, V. Bakken, W. Chen, W.L. Hase, Ab initio classical trajectories on the Born–Oppenheimer surface: Hessian-based integrators using fifth-order polynomial and rational function fits. J. Chem. Phys. **111**, 3800–3805 (1999)
3. N. Sathyamurthy, Computational fitting of AB initio potential energy surfaces. Comput. Phys. Rep. **3**, 1–69 (1985)
4. H.-M. Keller, H. Floethmann, A.J. Dobbyn, R. Schinke, H.-J. Werner, C. Bauer, P. Rosmus, The unimolecular dissociation of HCO. II. Comparison of calculated resonance energies and widths with high-resolution spectroscopic data. J. Chem. Phys. **105**, 4983–5004 (1996)
5. X. Zhang, S. Zou, L.B. Harding, J.M. Bowman, A global ab initio potential energy surface for formaldehyde. J. Phys. Chem. **108**, 8980–8986 (2004)
6. A.P. Bartók, M.J. Gillan, F.R. Manby, G. Csányi, Machine-learning approach for one- and two-body corrections to density functional theory: applications to molecular and condensed water. Phys. Rev. B **88**, 054104 (August 2013)
7. NIH., https://ncats.nih.gov/news/releases/2015/tox21-challenge-2014-winners (2014)
8. M. Rupp, A. Tkatchenko, K.-R. Müller, O.A. von Lilienfeld, Fast and accurate modeling of molecular atomization energies with machine learning. Phys. Rev. Lett. **108**, 058301 (2012)
9. R. Ramakrishnan, P.O. Dral, M. Rupp, O.A. von Lilienfeld, J. Chem. Theor. Comput. **11**, 2087 (2015)
10. S. Lu, Q. Zeng, H. Wu, A New Power Load Forecasting Model (SIndRNN): independently recurrent neural network based on softmax kernel function. IEEE 21st International Conference on High Performance Computing and Communications, https://doi.org/10.1109/HPCC/SmartCity/DSS.2019.00320, 2019
11. H. Wu, S. Lu, A. Lopez-Aeamburo, J. She, Temperature Prediction Based on Long Short-Term Memory Networks, CSCI'19, 2019
12. V. Botu, R. Ramprasad, Adaptive machine learning framework to accelerate ab initio molecular dynamics. Int. J. Quantum Chem. **115**(16), 1074–1083 (2015)
13. E. Apra, T.L. Windus, T.P. Straatsma, et al., NWChem, A computational chemistry package for parallel computers, version 5.0, Pacific Northwest National Laboratory, Richland, Washington, 2007
14. H. Wu et al., Higher-accuracy schemes for approximating the Hessian from electronic structure calculations in chemical dynamics simulations. J. Chem. Phys. **133**, 074101, 2010
15. H. Wu, et al., A High Accuracy Computing Reduction Algorithm Based on Data Reuse for Direct Dynamics Simulations, CSCI 2016
16. H. Wu and S. Lu, Evaluating the accuracy of a third order hessian-based predictor-corrector integrator, *Europe Simulation Conference*, 2016
17. H. Wu, S. Lu, et al., Evaluating the accuracy of Hessian-based predictor-corrector integrators. J. Cent. South Univ. **24**(7), 1696–1702 (2017)