Multi-Step Time Series Generator for Molecular Dynamics

Katsuhiro Endo, Katsufumi Tomobe, Kenji Yasuoka
Department of Mechanical Engineering, Keio University
3-14-1 Hiyoshi, Kohoku-ku, Yokohama
Kanagawa, Japan 223-8522

Abstract

Molecular dynamics (MD) is a powerful computational method for simulating molecular behavior. Deep neural networks provide a novel method of generating MD data efficiently, but there is no architecture that mitigates the well-known exposure bias accumulated by multi-step generations. In this paper, we propose a multi-step time series generator using a deep neural network based on Wasserstein generative adversarial nets. Instead of sparse real data, our model evolves a latent variable \( \mathbf{z} \) that is densely distributed in a low-dimensional space. This novel framework successfully mitigates the exposure bias. Moreover, our model can evolve part of the system (Feature extraction) with any time step (Step skip), which accelerates the efficient generation of MD data. The applicability of this model is evaluated through three different systems: harmonic oscillator, bulk water, and polyethylene melts. The experimental results demonstrate that our model can predict time series of the MD data with sufficient accuracy to calculate the physical and important dynamical statistics.

1. Introduction

Molecular dynamics (MD) is a standard and powerful tool for investigating molecular behavior, and has been used in various fields, e.g., polymers (Gee, Lacevic, and Fried 2006), proteins (Lindorff-Larsen et al. 2011), methane hydrates (Walsh et al. 2009), nucleation (Matsumoto, Saito, and Ohmine 2002), carbon nanotube (Legoa et al. 2003), and metals (Yamakov et al. 2004). The method solves the Newton’s equation of motion and provides the coordinates, velocities, and forces of all the atoms in the system at each time step. Although the applicability of the MD simulations is well known, the high computational costs of large-scale, long-time MD simulations limit the expansion of the method. If we directly calculate two-body forces between all atom pairs, the computational complexity per step is proportional to the square of the number of atoms, which presents a serious bottleneck when the MD simulations are conducted with large numbers of atoms. Recently, the number of atoms \( N \) and time length of the simulation \( T \) have reached scales of \( 10^9 \) (Shibuta et al. 2017) and \( 10^{10} \) (Needham et al. 2016), respectively; thus, a novel solution to overcome the bottleneck is desired.

Recently, deep neural networks have achieved significant results in many fields. Recurrent neural networks (RNNs) (Elman 1990), long short-term memory (LSTM) (Hochreiter and Schmidhuber 1997), and other derivative models (Mogren 2016; Fabius and van Amersfoort 2014) have led to remarkable advances in both discriminative and generative sequence data tasks. However, when the generative models are used repeatedly, the generations are affected by previous ones, leading to an accumulation of bias, called the exposure bias. In a naive generator, this bias causes the probability of error to grow quadratically with the iteration number (Ross and Bagnell 2010). Various studies have attempted to mitigate the exposure bias. In the natural language processing field, many studies use the beam search to mitigate effects of the exposure bias (Sutskever, Vinyals, and Le 2014). Some studies have mitigated the bias by using the specific loss functions that directly evaluate the correlations with the training sequence (Shen et al. 2015). A recent study by Yu et al. used extended generative adversarial nets (GANs) (Yu et al. 2017). Although the above studies achieved some success, our task of generating the MD data suffers from more serious exposure bias. The computational costs of the MD simulations mean that we would like to obtain long-term dynamical statistics from short-time-length MD data. To fulfill this requirement, we have to generate the short-time-length MD data repeatedly and concatenate them to make a long-time-length MD data; therefore, a novel framework for mitigating the exposure bias is needed.

To develop a new deep neural generative model, we introduce a multi-step time series generator (MD-GAN) that efficiently samples time series of the MD data (see Fig. 1A). Our model has the following advantages:

1. **Mitigation of the exposure bias**: as we consider time series generation not as \( p(X_t|X_{t-1}) \) but as \( p(X_{t-1}|z_{t-1}), p(X_t|z_t), \) and \( p(z_t|z_{t-1}) \), our model successfully mitigates the exposure bias.

2. **Feature extraction (contributing to the efficient sampling)**: our model can simulate the time evolution of a part of the whole system.

3. **Step skip (contributing to the efficient sampling)**: it can operate with any time step; thus, users can set a suitable...
time step for the target dynamics.

4. **Probabilistic time evolution:** thanks to the Wasserstein GANs (Arjovsky, Chintala, and Bottou 2017), our model gives a probabilistic time evolution.

5. **Identification of latent structures:** our model reveals the latent structure of the data through the behavior of the latent variable.

2. **Preliminaries**

Let us first consider a deterministic system and the evolution equation

$$\frac{d}{dt} x(t) = f(x(t)),$$

where $f$ is the function for time evolution determined by scientific considerations and $x(t) \in \mathbb{R}^D$ is a state variable in a $D$-dimensional space at time $t$. From this equation, we obtain the time series from the initial state $x(0)$. In computer simulations, the function $f$ adopts a finite-difference approximation in both the spatial and temporal dimensions to give a discretized equation $f_{\Delta t}$ with a time step $\Delta t$. The time step $\Delta t$ depends on the required precision, e.g., femtosecond timescale is required in all-atom MD simulations. The discretized equation $f_{\Delta t}$ evolves the state of the system as $x_{n+1} = f_{\Delta t}(x_n)$ where $x_n$ is $x(n \Delta t)$.

Consider an extracted feature $y_n \in \mathbb{R}^{D'}$ ($D' \leq D$) which contains rich physical information but exists in a lower-dimensional space (see Fig. 1 Feature extraction). However, unlike $x_n$, deterministic equations for most $y_n$ are unknown. Instead of the deterministic evolution, we consider stochastic evolution represented by a conditional probability of $y_n$, given all the previous states $p(y_n | Y_{n-1:0})$, where $Y_{k:a:s}$ is the time series of features with time step $s$ from time $a$ to time $b$ ($y_a, y_{a-s}, ..., y_b$). By convention, $s$ is omitted when $s = 1$. When calculating dynamical statistics that only depends on the time series of the features $Y$, we calculate the statistics from the time series obtained by the stochastic evolution $p(y_n | Y_{n-1:0})$. Moreover, when the dynamical statistics of interest do not depend on small-step dynamics, we use only the step-skipped stochastic evolution $p(y_n | Y_{k:n-1:0})$ (see Fig. 1 Step skip).

Our goal is to obtain the stochastic evolution $p(y_n | Y_{n-1:0})$. Unfortunately, this probability distribution is usually intractable. Machine learning can be applied to obtain an approximated distribution for the stochastic evolution. In this paper, we assume the following two assumptions:

- **Assumption 1:** the stochastic evolution of $y_i$ does not depend on $y_{i-m} (m > M)$, $p(y_n | Y_{n-1:0}) = p(y_n | Y_{n-1:M})$.

- **Assumption 2:** the distribution of the features $y_n$ is stationary and the stochastic evolution is time invariant. In this model, we focus on the physical “equilibrated systems”, which means that $p(y_n)$ is stationary, and we can use the same distribution for stochastic evolution at any time.

With these assumptions, we construct a novel generative model for time series. Let us consider $p(Y_{(k+1)M-1:M} | Y_{k:M-1:0})$. According to assumption 1, this can be denoted by $p(Y_{(k+1)M-1:kM} | Y_{k:M-1:(k-1)M})$. Using this sequence-to-sequence probability distribution, we form the time series as a generative model; however, when we use the generative model iteratively, we encounter the exposure bias (Ross and Bagnell 2010). In multi-step generations, the generative model uses the previous output as the real input, and this includes some small bias from the incompleteness of the model. Over the multi-step generations, this small bias accumulates and becomes non-negligible.

To clarify the problem, we assume that this bias can be divided into two types: *internal* and *external* biases. In many
th the mitigation of the bias. The distribution of feature $y_i$ is stationary; thus, the internal bias becomes small under the constraint $p(z_k) = p(z_{k-1})$.

3. Proposed architecture

Architecture of our model

In this section, we implement the novel generative model described in Section 2. The Wasserstein GAN (WGAN) (Arjovsky, Chintala, and Bottou 2017) is used to approximate the probability distribution. The WGAN has two networks: a generator $G : z \rightarrow Y$ and a discriminator $D : Y \rightarrow \mathbb{R}$. $G$ generates $Y$ from the probability distribution $p_G$ with a seed $z \sim p(z)$, and $D$ calculates the Wasserstein distance (WD) between the generated distribution $p_G$ and the objective distribution $p_R$ defined as

$$W(p_G,p_R) = \sup_{||g||_L \leq 1} \mathbb{E}_{x \sim p_G}[g(x)] - \mathbb{E}_{x \sim p_R}[g(x)],$$

where the supremum is over all the 1-Lipschitz functions $g$. The network $G$ is trained to minimize this WD, and the generator captures the objective distribution after the training.

Figure 2B shows the architecture of the proposed model. Visual icons show multi-layer affine networks (green pentagon), U-net based networks with $z$ inserted in the bottleneck of the U-net (green bow tie), a random vector (pink circle), the latent vector (blue circle), a discriminator (yellow rectangle), and the generated data (white circle).

Let $z_1 \sim \mathcal{U}(R^W)$ be sampled from $W$ dimensional uniform distribution between 0 and 1. $G_z$ is a generator and generates $z$ using previous $z$ and a random value sampled from $\mathcal{U}(R^W)$. Then, we apply $G_z$ to $z_1$, and it generates $z_2, z_3, \ldots$, respectively. $D_z$ is a discriminator and estimates the WD between $z_k$ and $z_{k+1}$ to make the distribution $p(z)$ stationary. Since we consider that the former $z$ ($z_1, z_2, z_3$, and $z_4$) are not stationary, we only apply the $D_z$ to the later pairs of $z$. After a consecutive pair of $z$ is randomly picked up ($z_5$ and $z_6$, or $z_0$ and $z_7$, or $\cdots$), $G_y$ generates $Y_1$ and $Y_2$ and makes $z_k$ and $z_{k+1}$. Training process of this network is as follows: (1) $G_z$ is trained to minimize the total WD estimated by all $D_z$ and $D_y$. (2) $G_y$ is trained to minimize the WD estimated by $D_y$. (3) Each $D_y$ was trained to estimate the WD between the pair of $z$, (4) $D_z$ is trained to estimate the WD between the generated and real data. At the inference, time series $z$ is generated by $G_z$ from $z_1 \sim \mathcal{U}(R^W)$, the former some steps is excluded, and time series $Y$ is generated from $z_k$ by $G_y$.

Inside structure of the networks

The $G_Y$ consists of the U-net (Ronneberger, Fischer, and Brox 2015) based networks, which transform uncorrelated multi dimensional time length $M$ normal distributions into some distributions. The $D_Y$ is multi-layer convolutional network. In our experiments, the U-net based networks achieved more accurate than the network only having deconvolution. We found that deconvolution in the $G_Y$ achieved most accurate when the stride size was equal to kernel size. 2194
The number of convolution layers is same as that of deconvolution layers as well as the U-net. The $G_z$ and $D_z$ consist of multi-layer affine networks. Training of $G_z$ can be dramatically fast when the Residual Network (ResNet) was used (He et al. 2016). Other parameters of the networks are needed to be tuned for each dataset.

**Advantages of the proposed model for MD**

This model has a number of advantages for the rapid prediction of dynamical statistics. The mitigation of the exposure bias is one of the novel advantages for the MD simulations. Because of the the dense variables $z$ embedded in a low-dimensional space, we can obtain long-term dynamical statistics using multi-step time evolutions. The other advantage is that this model needs only $2M$ steps of real data for the training. In the MD simulations, we have to calculate time series of the state variables $x$ that are much longer than the time length of the dynamics of interest. In many cases, this is the bottleneck in the scaling of MD simulations. Therefore, the short real time series requirement enhances the computational efficiency of the simulations. In addition to the above advantages, as our model uses the latent variable $z$ in a low-dimensional space, it is expected that the structures of the time series features $y_n$ are mapped into $p(z)$. Thus, we are able to extract hidden time-directional structures of the features. Moreover, compared with LSTM, our model is easier to analyze behavior of the latent variable because our model has only one latent variable. This characteristic provides a great benefit for the MD studies through this high interpretability.

**4. Experimental results**

We performed three experiments: harmonic oscillator with noise, water vibrational spectra, and polymer dynamics in melts. Each experiment was performed by following four steps: (1) running short and long MD simulations to get time series of $x$ for learning and comparison, respectively, (2) getting the pair of $Y$ (the consecutive pair of 64 steps of the extracted features) by Feature extraction and Step skip, (3) training the model using $Y$, and (4) comparing the generated and real long data. Table 1 shows dimensions of the parameters ($x$, $Y$, and $z$) in the experiments.

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Harmonic</th>
<th>Water</th>
<th>Polymer</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x$</td>
<td>1</td>
<td>2$x$3$x$330</td>
<td>2$x$3$x$30000</td>
</tr>
<tr>
<td>$Y$</td>
<td>64x1</td>
<td>64x3</td>
<td>64x3</td>
</tr>
<tr>
<td>$z$</td>
<td>16</td>
<td>16</td>
<td>16</td>
</tr>
</tbody>
</table>

**Harmonic oscillator with noise**

We first demonstrate that our model can evolve features while mitigating the exposure bias. Figure 3A shows our results for a harmonic oscillator with noise, where the top and bottom figures represent the reference and generated time series, respectively. The reference time series was generated by the equation

$$0.1 \sin(0.7t) + \eta,$$

where $\eta \sim \mathcal{N}(0, 0.01)$. Our model provides time series containing 64 steps, and each color shows the time series of each generation in Figure 3A (bottom). This result clearly shows that our model can generate time series with the correct amplitude, frequency, and noise. Remarkably, these quantities did not change after multi-step generations, indicating that our model successfully mitigates the exposure bias. Moreover, when the frequency changes due to the bias, the generated time series exhibits self-modification. Figure 3B shows the vibrational spectra obtained by the Fourier transform of the time series. The frequency and intensity of the mode in the generated time series are in good agreement with those in the reference. Thus, the generated time series quantitatively match the reference. To compare with other machine learning models, we calculated mean square error of the normalized spectra with real data. When we extended 128 into 1024 steps, the performance of our model is 79 and 104 times better than that of a naive model of the proposed architecture and C-RNN-GAN (Mogren 2016). The naive model
At and red denote whether the second dimension of the latent and fourth dimension of the latent variable, and the yellow.

Figure 3C shows transitions of the latent variable in the generation of the equilibrium bulk system. (A) Snapshot of the system containing 110 water molecules. (B) Schematic view of vibrational motion of the water molecule. (C) OH relative velocities of the MD (top) and generated data by our model (bottom). Blue, green, and red lines represent x, y, and z components of the velocity vector, respectively. (D) OH vibrational spectra of water molecules using the MD and generated time series. The spectra were obtained by the Fourier transform of the OH relative velocity correlation function (the vibrational density of states) (Tanzi, Ramondo, and Guidoni 2012),

$$P(\omega) = \sum_{k=1}^{N} \int_{-\infty}^{\infty} \langle \dot{r}_k(0) \dot{r}_k(t) \rangle e^{i\omega t} dt,$$

where $$\dot{r}_k(0) \dot{r}_k(t)$$ is the time series velocity auto-correlation function and $$\langle \rangle$$ is the ensemble average. Figure 4D shows OH vibrational spectra from the MD and generated time series of OH velocities. A notable point is that input velocities are not long time enough to calculate the statistic ($$M = 64$$ steps = 61.92 fs). Figure 4C shows the OH velocities of the MD and generated data in the water molecules. As this figures shows, the vibrational motion of the water molecules is complex. Vibrational spectra were obtained by taking the Fourier transform of the OH relative velocity correlation function (the vibrational density of states) (Tanzi, Ramondo, and Guidoni 2012).

Water vibrational spectra

Next, we applied our model to a more complex system: the bulk water. Water molecules are an important target of research in chemical, physical, and biological fields. Because of thermodynamic fluctuations and interactions with other molecules, the OH bonds of water molecules have specific vibrational motions. We performed ab initio MD simulations, which is an MD method on the basis of ab initio energy calculations, for the bulk water (see Fig. 4A). To test the applicability for real time series, we used the OH motions as an input and calculated vibrational spectra using the generated time series. For the ab initio MD simulations, we used the CPMD code (Hutter and Iannuzzi 2005) using the Car-Parrinello method (Car and Parrinello 1985). In the CPMD simulations, the Perdew-Burke-Ernzerhof functional was used to approximate the exchange-correlation terms (Perdew, Burke, and Ernzerhof 1996), and we described the valence-core interaction using the Martins-Troullier pseudo-potentials (Troullier and Martins 1991). The CPMD simulation was conducted under the isothermal and iso-volume conditions (305 K), and the volume of the system was predefined by the classical isothermal and isobaric classical MD simulation using the NAMD 2.9 software (Phillips et al. 2005) with the TIP3P water model (Jorgensen et al. 1983). In this experiment, the OH relative velocities were prepared as input for our model, and our model generated the time series of OH velocities. A notable peak at around 8300 cm$^{-1}$ using the generated time series. Unfortunately, there is one unexpected peak at around 8300 cm$^{-1}$. This peak is unphysical and can be attributed to the bias. Note that larger training data reduce a noise of the libration peak (see Fig. 4D insets).
Polymer dynamics in melts

Polymer materials have unique and complex dynamics in the melts, and the dynamics depends on various molecular and thermodynamic parameters. We performed the MD simulations of polymer polyethylene (PE) melts to simulate the complex dynamics of the polymer. The MD simulations were conducted using the TraPPE-UA force field (Martin and Siepmann 1998) and the GROMACS package (Pronk et al. 2013) under the same conditions and preparations as a previous study (Takahashi et al. 2017). Figure 5A shows a snapshot of the system filled with 300 PE polymer chains. In the MD simulations, the diffusion coefficient can be characterized by the mean square displacement (MSD):

\[ g(t) = \langle (r(t) - r(0))^2 \rangle, \]

where \( r(t) \) is the coordinate vector at time \( t \). It is known that the slope of the MSD (diffusion coefficient) of the chain center anomaly changes with time because of the chain entanglement (Takahashi et al. 2017). Figure 5B shows a schematic view of the chain entanglement and the concept of tube theory (Doi and Edwards 1978). In tube theory, chain entanglement is described as a tube constraint, and the polymer chain can only move along the major axis like a snake (reptation motion). Figure 5C shows the MSDs of the chain center of the polymers using time series generated by the MD simulation and our model. In our model, each color shows each generation which contains 64 steps (= 640 ps). In the MD simulation results, the slopes of the MSDs (i.e., diffusion coefficient) changes from \( t^{-0.5} \) to \( t^1 \); this transition time is called the end-to-end relaxation time \( \tau_R \). This retardation can be described as the tube constraint by the entanglement of surrounding polymers, and \( \tau_R \) corresponds to the chain length of the polymer (see Fig. 5B right). In the result given by our model, the MSD also has a transition time \( \tau_R \). Remarkably, \( \tau_R \) is much longer than the length of the one-step sequence; thus, our model generates sequence data from the precise phase space (a space containing all possible states of the system), and the generated time series enclose the short- and long-term dynamics of the polymers.

Table 2 shows calculation performance of the MD simulation and our model with one process of the Intel Core i7-4930K. Using one of the fastest MD package GROMACS, whole calculations reached 55.2 days when we obtained the averaged MSD of the polymers for 40 ns. In our model, total calculation time was 67.9 hours in this experiment, and most calculation time was the training. As a result, we achieved our goal of obtaining the averaged MSD of chain center 19 faster than the MD simulations. These results show that the time evolutions given by our model are precise enough to simulate the complex dynamics of the polymers; moreover, our model much more efficiently generates time series of the target features than the MD simulations.

4. Conclusions and future work

In this paper, we have presented a multi-step time series generator using the Wasserstein GAN-based deep neural network to efficiently generate time series of MD simulations. Using our novel framework, the model successfully mitigates the exposure bias. MD simulation research will ben-
Table 2: Calculation performance of the MD simulation and our model with one process of the Intel Core i7-4930K.

<table>
<thead>
<tr>
<th></th>
<th>MD (GROMACS)</th>
<th>Our model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed (1.38 day/ns)</td>
<td>67.9 h/model</td>
<td></td>
</tr>
<tr>
<td>Speed (40 ns MSD)</td>
<td>55.2 days</td>
<td>2.83 days</td>
</tr>
</tbody>
</table>

benefit from several advantages (e.g., step skip, feature extraction, and the structures of the latent variable z). Using various systems (harmonic oscillator, bulk water, and polymer melts), we have demonstrated the capability of our model to generate time series that are sufficiently precise to calculate physical and important dynamical statistics. However, it remains difficult to verify that the assumption 1 was satisfied and the trained probability was the same as that of the real data. In future work, we will study five aspects.

(1) The latent variable z can extract the important structures of the system in a low-dimensional space; however, it is still difficult to analyze the characteristics of the structures. Some suitable analysis methods are required. (2) In our model, there are two generators, G_x and G_y; thus, it is difficult to determine when to terminate the learning step, because the convergence properties of the two generators are basically different. Other multi-network coexistence architectures (e.g., GAN) also suffer from this drawback. We will attempt to provide some metric for finding the best point between over- and under-fitting. (3) The model requires sensitive adjustment of hyperparameters, as in other deep neural networks. (4) Our results showed that dynamics properties (e.g., MSD) can be calculated using smaller time length than the time length of the dynamics properties. However, it is unclear what time length is required. (5) Finally, there is no method to verify that the generated data follows the probability distribution of the real data. Future works will also include an improved model that is easier to use.

Acknowledgments

This work is supported in part by Ministry of Education, Culture, Sports, Science and Technology (MEXT) as Research and Development of Next-Generation Filed. K.T. is supported by MEXT Grant-in-Aid for the Program for Leading Graduate Schools.

References


