Proceedings of Computational Engineering Conference JSCES
Vol. 30, Jun. 2025

Learning to Model the Discrete Particulate Phase with Graph Neural Networks

Jia Guo [1] and Kazunori Fujisawa [2]

[1] Ph.D, Associate Professor (Kitashirakawa Oiwakecho, Sakyo ku, Kyoto, E-mail: guo.jia.8x@kyoto-u.ac.jp)
[2] Ph.D, Professor (Kitashirakawa Oiwakecho, Sakyo ku, Kyoto, E-mail: fujisawa.kazunori.2s@kyoto-u.ac.jp)

In this study, we propose a deep learning framework that models discrete particulate phase directly from particle's position data, eliminating the need for parameter calibration or empirical equations in conventional discrete element method. Our approach leverages a learnable graph neural network (GNN) to represent the dynamic particulate systems, embedding particle state features in graph nodes and interaction features in graph edges. Specifically, the interaction forces are learned during graph training through a message-passing mechanism, where graph nodes progressively refine their representations by exchanging information with neighboring nodes. The feasibility of the proposed method is attribute to the incorporation of relational inductive biases into the deep learning architecture, which naturally align with particle-based problems and thus enhance the learning of pairwise relations.

Key Words : *Deep learning, Data-driven modeling, Discrete particulate phase, Graph Neural Network, Message-passing mechanism*

1. INTRODUCTION

Comprehending the complex behavior of granular materials may commence by modeling the microscopic interactions within the discrete particulate phase that govern macroscopic behavior. The discrete element method (DEM) is a prevalent numerical approach for simulating this process. Advancements in computational power have made the DEM increasingly popular for modeling and understanding complex dynamic systems involving large numbers of small particles. Despite its widespread adoption for addressing engineering challenges like granular flow, DEM might suffer from inaccurate or incomplete representations either due to their high parameter calibration costs or due to the reliance on empirical equations to describe interaction forces.

In this study, we propose a deep learning framework that models discrete particulate phase directly from particle's position data, eliminating the need for parameter calibration or empirical equations in conventional DEM. Our approach leverages a learnable graph neural network (GNN) to represent the dynamic particulate systems, embedding particle state features in graph nodes and interaction features in graph edges. Specifically, the interaction forces are learned during graph training through a message-passing mechanism, where graph nodes progressively refine their representations by exchanging information with neighboring nodes. Moreover, our model is trained using a multi-step loss function based on autoregressive process to enhance long-term prediction. Due to page limitations, this paper presents only the framework of the deep learning method. Detailed numerical and experimental validation results will be provided in a future journal publication.

2. GOVERNING EQUATIONS AND INTERACTION FORCES

According to Newton's second law of motion, the translational and rotational movements of particle i, $(i = 1, \cdots, N)$, are governed by the equations as follows:

$$\begin{cases} m_i \dfrac{\mathrm{d}\mathbf{v}_i}{\mathrm{d}t} = \displaystyle\sum_{j=1}^{n_c} \mathbf{f}_{ij}^{c} + \mathbf{f}_i^{fp} + m_i \mathbf{g} \\ I_i \dfrac{\mathrm{d}\boldsymbol{\omega}_i}{\mathrm{d}t} = \displaystyle\sum_{j=1}^{n_c} \mathbf{T}_{ij}^{c} \end{cases} \tag{1}$$

where m_i, I_i, $\mathbf{v}_i$ and $\boldsymbol{\omega}_i$ are the mass, moment of inertia, linear velocity and angular velocity of the particle, respectively. $\mathbf{f}_{ij}^{c}$ and $\mathbf{T}_{ij}^{c}$ are the particle-particle interaction/contact force and torque acting on particle i by particle j, and n_c is the number of total contacts of particle i. $\mathbf{f}_i^{fp}$ is the fluid-particle interaction force if fluid phase also exists and $\mathbf{g}$ is the gravitational acceleration. In this research, only particle-particle contact force in discrete particulate phase is considered for simplicity.

3. LEARNING STRATEGIES USING GRAPH NEURAL NETWORKS

Our approach is inspired by previous works [1] that represent subsurface as graphs. We formalize the problem of learning particle-particle interaction forces as follows.

(1) Algorithm

Using the governing equations for particles in Eq. (1), let $\hat{\mathbf{y}}^{1:T} = \left\{\hat{\mathbf{y}}^1, \hat{\mathbf{y}}^2, \cdots \hat{\mathbf{y}}^T\right\}$ where $\hat{\mathbf{y}}^t \in \mathbb{R}^{N \times d}$ denotes particles' predicted acceleration at time step t and the hat symbol represents neural network estimated values. d is

the dimension. Given $\mathbf{x}^t \in \mathbb{R}^{N\times 2d}$ as the displacement and velocity states variables at current time step t, we construct a graph network based model GN to approximate the particles' acceleration from the current states and employ an Euler based integration operator Ψ to compute the particles' next states, which is defined as

$$\hat{\mathbf{x}}^{t+1} = f_\theta\left(\mathbf{x}^t, \mathbf{s}, \Delta t\right) := \begin{cases} \hat{\mathbf{y}}^{t+1} = \mathrm{GN}\left(\mathbf{x}^t, \mathbf{s}\right) \\ \hat{\mathbf{x}}^{t+1} = \Psi\left(\hat{\mathbf{y}}^{t+1}, \mathbf{x}^t, \Delta t\right). \end{cases} \tag{2}$$

The static variable $\mathbf{s}$ consists particle related variables, e.g. Young's modulus E, particle's mass m_i, particle shape related parameters d_i, etc. For each particle i, we use semi-implicit Euler integration scheme to update its state given time interval Δt. The operator Ψ is thus written as

$$\begin{cases} \hat{\mathbf{v}}_i^{t+1} = \mathbf{v}_i^t + \Delta t \hat{\mathbf{y}}_i^{t+1} \\ \hat{\mathbf{p}}_i^{t+1} = \mathbf{p}_i^t + \Delta t \hat{\mathbf{v}}_i^{t+1}. \end{cases} \tag{3}$$

To enable accurate long-term prediction, we adopt an autoregressive (AR) process by incorporating an AR sliding window with width K [2]. The AR process utilizes $\mathbf{x}^t$ as input, and implements the following mapping:

$$\begin{cases} \hat{\mathbf{x}}^{t+1} = f_\theta\left(\mathbf{x}^t, \mathbf{s}, \Delta t\right) \\ \hat{\mathbf{x}}^{t+k} = f_\theta\left(\hat{\mathbf{x}}^{t+k-1}, \mathbf{s}, \Delta t\right), \text{where } k = 2, \cdots, K. \end{cases} \tag{4}$$

(2) Graph network based model GN

a) Encoder

The encoder firstly embeds the input $\{\mathbf{x}^t, \mathbf{s}\}$ into a latent graph $G^{(0)} = \left(V^{(0)}, E^{(0)}\right)$ using multilayer perceptrons (MLP), where $v_i^{(0)} \in V^{(0)}$ is the node embedding features and $e_{ij}^{(0)} \in E^{(0)}$ is the edge embedding features. Specifically, in Encoder the features of each node are encoded from $(\mathbf{x}_i^t, \mathbf{s})$ $(i = 1, \cdots, N)$, and the features of its corresponding edge $e_{ij}^{(0)}$ are encoded from $\left(\mathbf{x}_i^t - \mathbf{x}_j^t\right)$ $(j = 1, \cdots N_c)$, where N_c is the number of contacted neighboring nodes from node i.

b) Processor

The processor is a stack of M graph neural network layers, i.e. $\mathbf{G} = \left(G^{(1)}, \cdots, G^{(M)}\right)$, each one performing one round of message passing that finds interaction features between neighboring nodes. Given latent graph $G^{(m)} = \left(V^{(m)}, E^{(m)}\right)$ on the mth layer, where $v_i^{(m)} \in V^{(m)}$ and $e_{ij}^{(m)} \in E^{(m)}$, it first computes the message on the edges based on the target node i as well as its neighboring nodes:

$$e_{ij}^{(m)} = \mathrm{MLP}_e(v_i^{(m)} - v_j^{(m)}),\ j = 1, \cdots, N_c. \tag{5}$$

Given the edge features of the current layer, the target node i will aggregate the messages sent to it by simply edge message summation i.e. $a_i^{(m)} = \sum_j e_{ij}^{(m)}$ and the node features for latent graph $G^{(m+1)}$ are updated as:

$$v_i^{(m+1)} = \mathrm{MLP}_v\left(\left[v_i^{(m)}, a_i^{(m)}\right]\right) \tag{6}$$

where '$[\cdot,\cdot]$' is concatenation on the feature dimension. The number of message-passing steps M required will likely scale with the complexity of the interactions. The message-passing mechanism is illustrated in **Fig.1**.

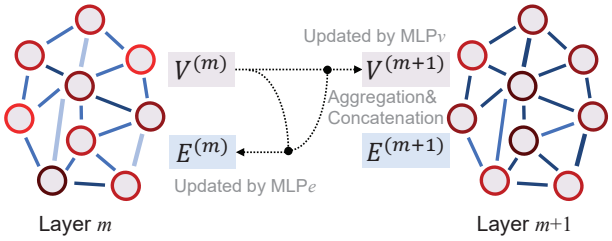

Figure 1 Message passing mechanism in GNN

c) Decoder

We use another MLP to map the out of the processor back to the predicted acceleration $\hat{\mathbf{y}}_i^{t+1}$ at the next time step in Eq. (2) to update the states of each particle.

(3) Loss function and training

To improve long-term prediction, we define the multi-step loss function with the weight parameter λ_k as

$$\mathcal{L}(\theta) = \frac{1}{N\times T}\sum_{t=1}^{T}\left[\sum_{k=1}^{K}\lambda_k\|\hat{\mathbf{y}}^{t+k} - \mathbf{y}^{t+k}\|^2\right], \tag{7}$$

where the weights for the step k greater than one are diminished, as the multi-step loss is significantly worse at the onset of training when the model's accuracy is low. The ground truth $\mathbf{y}^{t+k}$ is obtained from the ground truth position data given time interval Δt. We optimized the model parameters θ over this loss with the Adam optimizer.

4. CONCLUSION

We proposed a deep learning framework using GNN to compute the motion of discrete particles in many-body problems, which allows for probing microscopic interactions within the discrete particulate phase at the level of macro behavior without relying on predefined empirical equations. Future work will extend this framework to encompass more complex numerical and experimental models. The learned models can also be used to infer static properties for the dynamic systems by solving inverse problems with observation data.

References

[1] Wu T, Wang Q, Zhang Y, et al. Learning large-scale subsurface simulations with a hybrid graph network simulator[C]//Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2022: 4184-4194.

[2] Guo J, Enokida R, Li D, Ikago K. Combination of physicsbased and datadriven modeling for nonlinear structural seismic response prediction through deep residual learning. Earthquake Engineering & Structural Dynamics, 2023, 52(8): 2429-2451.