A The Derivations of NC (0) for Other EGNN Models

In this section, we first illustrate how the other two baseline models (i.e. RF [17] and GMN [19]) can be derived to NC (0), and then discuss multi-layer EGNN.

A.1 RF

The overall equivariant convolution process of RF is similar to that of EGNN. The main difference is that RF does not use the node feature. Instead, it leverages the L2 norm of the velocity as an additional feature to update the predicted velocity:

$$\hat{\mathbf{v}}_i^0 = \text{Norm}(\mathbf{v}_i^0)\big(\phi_v(\mathbf{h}_i)\mathbf{v}_i^0 + \frac{1}{N-1}\sum_{j\neq i}(\mathbf{x}_i^0 - \mathbf{x}_j^0)\mathbf{m}_{ij}^0\big), \tag{19}$$

$$\hat{\mathbf{x}}_i^T = \mathbf{x}_i^0 + \hat{\mathbf{v}}_i^0 T, \tag{20}$$

where $\text{Norm}(\mathbf{v}_i^0)$ denotes the L2 norm of the input velocity $\mathbf{v}_i^0$. Therefore, RF can be also viewed as a NC (0) model.

A.2 GMN

The main difference between GMN and EGNN is that GMN detects the sub-structures in the system and process the particles in each special sub-structure independently. Specifically, GMN re-formulates the original EGNN (i.e., Equations (2-4)) as follows:

$$\mathbf{m}_{ij}^0 = \phi_e(\mathbf{h}_i, \mathbf{h}_j, ||\mathbf{x}_i^0 - \mathbf{x}_j^0||^2, e_{ij}), \tag{21}$$

$$\hat{\mathbf{v}}_k^0 = \phi_v(\sum_{i\in\mathcal{S}_k}\mathbf{h}_i)\mathbf{v}_k^0 + \sum_{i\in\mathcal{S}_k}\phi_k(\mathbf{x}_i^0, \mathbf{m}_{ij}^0), \tag{22}$$

$$\hat{\mathbf{v}}_i^0 = \text{FK}(\hat{\mathbf{v}}_k^0), \tag{23}$$

$$\hat{\mathbf{x}}_i^T = \mathbf{x}_i^0 + \hat{\mathbf{v}}_i^0 T, \tag{24}$$

where $\hat{\mathbf{v}}_k^0$ is the predicted velocity of the sub-structure. GMN then uses it to calculate the velocity of each particle by a function FK which can be either learnable or based on the angles and relative positions in the sub-structure [19].

A.3 Multi-layer EGNN

In current EGNN methods, the input coordinate $\mathbf{x}_i^0$ and velocity $\mathbf{v}_i^0$ are regarded as constant feature and used in different layers. For example, in the multi-layer version EGNN, $\mathbf{m}_{ij}^0$ will be formulated as follows:

$$\mathbf{m}_{ij}^{0,l} = \phi_e(\mathbf{h}_i^{l-1}, \mathbf{h}_j^{l-1}, ||\mathbf{x}_i^0 - \mathbf{x}_j^0||^2, e_{ij}), \tag{25}$$

where $\mathbf{m}_{ij}^{0,l}$ is $\mathbf{m}_{ij}^0$ in layer l and $\mathbf{h}_i^{l-1}$ denotes the hidden feature in layer $l-1$. Evidently, only the hidden features are updated within different layers. The overall formulation of multi-layer layer is akin to the single-layer EGNN.

B Proofs of Things

B.1 Proof of Proposition 3.1

Proof. The objective of the existing methods for a single system can be defined as:

$$\begin{aligned}
&\underset{\hat{\mathbf{v}}^0}{\operatorname{argmin}} \sum_{p_i} (\mathbf{x}_i^T - \hat{\mathbf{x}}_i^T) && (26)\\
&= \sum_{p_i} (\mathbf{x}_i^T - \mathbf{x}_i^0 - \hat{\mathbf{v}}_i^0 T) && (27)\\
&= \sum_{p_i} T(\frac{\mathbf{x}_i^T - \mathbf{x}_i^0}{T} - \hat{\mathbf{v}}_i^0) && (28)\\
&= T \sum_{p_i} \big(\mathbf{v}_i^{t^*} - (\phi_v(\mathbf{h}_i)\mathbf{v}_i^0 + \frac{\sum_{j\neq i} (\mathbf{x}_i^0 - \mathbf{x}_j^0)\mathbf{m}_{ij}^0}{N-1})\big) && (29)\\
&= T \sum_{p_i} \big(\mathbf{v}_i^{t^*} - (w^0 \mathbf{v}_i^0 + \mathbf{b}^0)\big), && (30)
\end{aligned}$$

where $w^0 \in \mathbb{R}^1$ and $\mathbf{b}^0 \in \mathbb{R}^3$ denote the learnable variables irrelevant to $\mathbf{v}_i^0$ and t, concluding the proof. □

B.2 Proof of Proposition 3.3

Proof. As the higher order cases ($k \geq 1$) have already been proved, we only need to show that $\epsilon_{\text{NC}(0)} \geq \epsilon_{\text{NC}(1)}$. The first order of Newton-Cotes formula NC (1) is also known as Trapezoidal rule, i.e.:

$$\int_0^T \mathbf{v}(t)dt \approx \frac{T}{2}(\mathbf{v}^0 + \mathbf{v}^T). \tag{31}$$

As aforementioned, the actual integration $\mathbf{x}^T - \mathbf{x}^0$ for different training examples is different, and we assume that it fluctuates around the base estimation $\frac{T}{2}(\mathbf{v}^0 + \mathbf{v}^T)$ and follows a normal distribution $\mathcal{N}_{NC(1)}$, where the variance $\sigma^2_{NC(1)}$ is positively correlated with the difficulty of optimizing the overall objective. The variance of $\mathcal{N}_{NC(1)}$ is:

$$\sigma^2_{NC(1)} = \frac{\sum_p (\int_0^T (\mathbf{v}(t) - \sum_{k=0}^1 C^k t^{(k)})dt)^2}{NT^2}, \tag{32}$$

where the integration term is a general form of polynomial interpolation error. According to Equation 14, it can be derived to:

$$\int_0^T (\frac{(t-t^0)(t-t^1)\mathbf{v}^{''}(\xi)}{2!})dt, \tag{33}$$

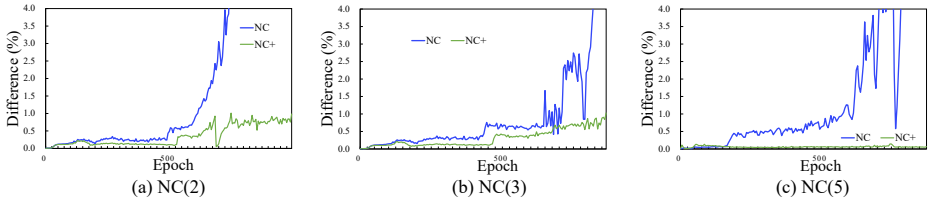
where $\mathbf{v}^{''}$ denote the second derivative of $\mathbf{v}$. Let $s = \frac{t-t^0}{T}$, then $t = t^0 + sh$ and $dt = d(\mathbf{v}^0 + sh) = Tds$. the above equation can be re-written as:

$$T \int_0^1 \frac{s(s-1)T^2\mathbf{v}^{''}(\xi)}{2} ds = -\frac{1}{12} T^3 \mathbf{v}^{''}(\xi) = \mathcal{O}(T^3). \tag{34}$$

Therefore, the final $\sigma^2_{NC(1)}$ is:

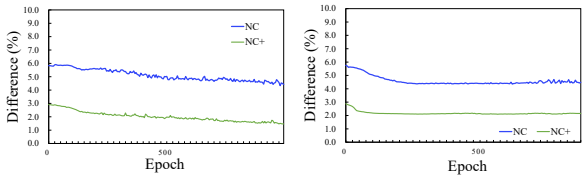
$$\begin{aligned}
\sigma^2_{NC(1)} &= \frac{\sum_p (-\frac{1}{12} T^3 \mathbf{v}^{''}(\xi))^2}{NT^2} && (35)\\
&= \mathcal{O}(T^4) \leq \mathcal{O}(T^2) = \sigma^2_{NC(0)}, && (36)
\end{aligned}$$

concluding the proof. □



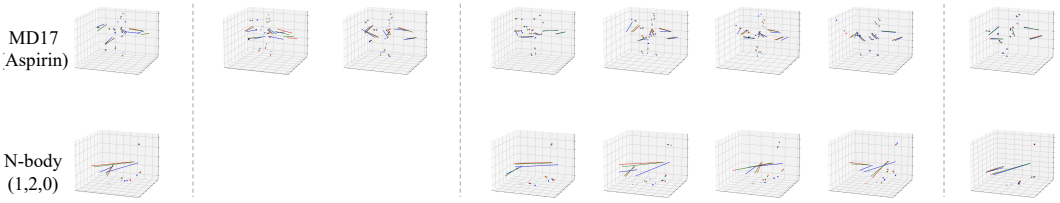
(a) NC(2) (b) NC(3) (c) NC(5)

Figure 6: The prediction errors of intermediate velocities on MD17 dataset, w.r.t. training epoch. The blue and green lines denote the prediction errors of NC and NC$^+$, respectively.



(a) NC(2) (c) NC(5)

Figure 7: The prediction errors of intermediate velocities on N-body dataset, w.r.t. training epoch. The blue and green lines denote the prediction errors of NC and NC$^+$, respectively.



(a) NC(2) intermediate (b) NC(3) intermediate (c) NC(5) intermediate (d) NC(2) final

Figure 8: Visualization of the intermediate velocities w.r.t. k. The red, blue, and green lines denote the target, prediction of NC, and prediction of NC$^+$, respectively.

B.3 Proof of Proposition 3.4

Proof. The GNN models possessing equivariance property are equivariant to the translation, rotation, and permutation of input. NC directly feeds the input into these backbone models and naturally possesses this property.

Formally, let $T_g : \mathbf{x} \to \mathbf{x}$ be a group of transformation operations. If the backbone model $\mathcal{M}$ is equivariant then we will have:

$$\mathcal{M}(T_g(\mathbf{x})) = S_g(\mathcal{M}(\mathbf{x})), \tag{37}$$

where S_g is an equivalent transformation to T_g on the output space. NC can be regarded as a weighted combination of the outputs of $\mathcal{M}$:

$$\sum_i w_i \mathcal{M}(T_g(\mathbf{x}_i))) = \sum_i w_i S_g(\mathcal{M}(\mathbf{x}_i)), \tag{38}$$

where the Newton-Cotes weights w_i is constant and irrelevant to the input, the output, and the model $\mathcal{M}$ itself. Therefore, the above equation will always holds. □

C Additional Experimental Results

The average intermediate velocity prediction errors on MD17 and Motion datasets are shown in Figure 6 and Figure 7, respectively. NC still learned the intermediate velocities we did not feed the intermediate into it. Particularly, the error on N-body dataset was small and stable, which may demonstrate the effectiveness of estimating intermediate velocities even without supervised data.

Table 4: Hyper-parameter settings in the main experiments.

Datasets	# steps	velocity regularization	velocity regularization decay	parameter regularization	parameter regularization decay	loss criterion	input feature normalization	intermediate velocity normalization
N-body	2	0.001	0.999	1.0	0.99	MSE	False	True
MD17	2	0.1	0.999	1.0	0.95	MSE	True	True
Motion	2	0.01	0.999	1.0	0.95	MSE	True	True

	# epoch	batch-size	# training examples	activation	# layers	learning rate	optimizer	clip gradient norm
N-body	1,500	200	500	ReLU	4	0.0005	Adam	1.0
MD17	1,000	100	500	ReLU	4	0.001	Adam	0.1
Motion	1,500	100	200	ReLU	4	0.0005	Adam	1.0

We also provide some visualized examples on these two datasets in Figure 8, from which we can observe the similar results compared with Figure 4 in the main paper.

D Hyper-parameter Setting

We list the main hyper-parameter setting of NC with different EGNN models on different datasets in Table 4. We used the Adam optimizer [47] and adopted layer normalization [48] and ReLU activation [49]) for all settings. For a fair comparison, the parameter settings for the backbone EGNN models were identical to these in the existing implementation [19].