



PAPER • OPEN ACCESS

Extreme time extrapolation capabilities and thermodynamic consistency of physics-inspired neural networks for the 3D microstructure evolution of materials via Cahn–Hilliard flow

To cite this article: Daniele Lanzoni *et al* 2024 *Mach. Learn.: Sci. Technol.* **5** 045017

View the [article online](#) for updates and enhancements.

You may also like

- [Machine-learning-based diabetes classification method using blood flow oscillations and Pearson correlation analysis of feature importance](#)
Hanbeen Jung, Chaebeom Yeo, Eunsil Jang et al.
- [Training machine learning interatomic potentials for accurate phonon properties](#)
Antoine Loew, Hai-Chen Wang, Tiago F T Cerqueira et al.
- [Metric flows with neural networks](#)
James Halverson and Fabian Ruehle



PAPER

OPEN ACCESS






RECEIVED
1 August 2024REVISED
9 September 2024ACCEPTED FOR PUBLICATION
9 October 2024PUBLISHED
18 October 2024

Original Content from
this work may be used
under the terms of the
[Creative Commons
Attribution 4.0 licence](#).

Any further distribution
of this work must
maintain attribution to
the author(s) and the title
of the work, journal
citation and DOI.



Extreme time extrapolation capabilities and thermodynamic consistency of physics-inspired neural networks for the 3D microstructure evolution of materials via Cahn–Hilliard flow

Daniele Lanzoni^{1,*} , Andrea Fantasia¹ , Roberto Bergamaschini¹ , Olivier Pierre-Louis² 
and Francesco Montalenti¹ 

¹ Department of Materials Science, University of Milano-Bicocca, Via R. Cozzi 55, Milano I-20125, Italy

² Institut Lumière Matière, UMR5306 Université Lyon 1–CNRS, 69622 Villeurbanne, France

* Author to whom any correspondence should be addressed.

E-mail: daniele.lanzoni@unimib.it

Keywords: recurrent neural networks, convolutional neural network, Cahn–Hilliard, time evolution of microstructures, temporal extrapolation, physics-inspired neural network

Supplementary material for this article is available [online](#)

Abstract

A Convolutional Recurrent Neural Network (CRNN) is trained to reproduce the evolution of the spinodal decomposition process in three dimensions as described by the Cahn–Hilliard equation. A specialized, physics-inspired architecture is proven to provide close accordance between the predicted evolutions and the ground truth ones obtained via conventional integration schemes. The method can accurately reproduce the evolution of microstructures not represented in the training set at a fraction of the computational costs. Extremely long-time extrapolation capabilities are achieved, up to reaching the theoretically expected equilibrium state of the system, consisting of a layered, phase-separated morphology, despite the training set containing only relatively-short, initial phases of the evolution. Quantitative accordance with the decay rate of the free energy is also demonstrated up to the late coarsening stages, proving that this class of machine learning approaches can become a new and powerful tool for the long timescale and high throughput simulation of materials, while retaining thermodynamic consistency and high-accuracy.

1. Introduction

Recently, there has been a surge in interest in Machine Learning (ML) methods [1, 2] in the Computational Physics and Materials Science community [3–6]. For instance, the new field of ML interatomic potentials [7, 8] promises to push the boundaries of tractable time and spatial scales in molecular dynamics. An initially less explored route, which has however gained traction [9–15], stems in the possibility of leveraging ML approaches, and Neural Networks (NN) in particular, for meso- and macroscopic models. Indeed, while it is easier to reach experimental scales using such tools, computational costs might still be critical for stiff problems and fine discretization requirements. ML may therefore provide an alternative to conventional solvers and allow one to bypass these issues. Indeed, in the last years, efforts have been dedicated to the possibility of approximating Partial Differential Equations (PDEs) with ML models [16, 17], with a particular interest on the morphological and microstructural evolution of materials [13, 14, 18, 19].

In this work, we explore the possibility of using a Convolutional Recurrent Neural Network (CRNN) to reproduce the time-dependent solutions of the Cahn–Hilliard equation, which describes the spinodal decomposition, an important process in binary mixtures leading to a spontaneous separation into two different phases [20–23], focusing on their long-time behavior. As the concentration fields can be conveniently mapped into a single order parameter, φ , the considered model also provides an initial test for the more general class of Phase Field (PF) models [24, 25]. This is advantageous as PF models proved to be effective in several fields of Materials Science [24–28], thanks to their natural ability to tackle complex geometries, eventually involving topological changes, such as domain coalescence and splitting. Additionally,

their formulation is open to the addition of multiple physical contributions, thus making the development of NN surrogates particularly appealing.

A 2D version of the spinodal decomposition has already been addressed via ML methods by the community in previous works, also using CRNNs [12, 13]. However, tackling the full 3D problem is essential to model realistic materials. For instance, the 3D coarsening of the spinodal decomposition may exhibit bi-continuous patterns, under suitable conditions, which cannot be observed in 2D. Additionally, full 3D simulations are computationally more challenging due to the requirement of a higher number of collocation points in discretization procedures, which further calls for acceleration with novel tools. Steps in this direction can be found in a few, very recent publications. In [29] a Convolutional (but not Recurrent) NN is shown to nicely predict relatively few steps of spinodal decomposition evolution, sufficient for the specific application therein. [19], instead, shows that a recurrent, graph network can surrogate grain coarsening dynamics in 2D and 3D. In the same work, an adaptive time-stepping scheme is proposed, which allows one to obtain an even larger speed-up in predictions. Still, only sequences of some hundred states are shown for the 3D case. In the present work, we therefore focus on the possibility of obtaining a model capable of producing stable, extremely long microstructural evolutions that are consistent with the underlying thermodynamic assumptions. This is done by exploiting a physics-inspired layer, which closely mimics the underlying material flow process in the ML model architecture. Importantly, predicted sequences are shown to reach the correct stationary state of the system, despite the CRNN only being presented orders of magnitude shorter, far from equilibrium sequences during training.

The paper is organized as follows. First, we briefly revise the spinodal decomposition model considered and we discuss the dataset creation procedure (section 2.1). Next, in section 2.2 we define the NN method and we inspect training/validation performances. In section 3.1, we analyze the generalization capabilities of the CRNN, both in terms of computational cell size and length of the generated sequence. Lastly, a quantitative comparison between predictions and ground-truth evolution is performed, both in terms of local, voxel-level correspondence and global thermodynamic properties (section 3.3). Statistical significance is also tested.

2. Methods

2.1. PF model and dataset generation

Spinodal decomposition consists in the spontaneous separation of a binary mixture in two different phases below a critical temperature. From a mathematical standpoint, it is convenient to map the concentration fields tracking the local chemical composition to a single scalar field φ . This way, the dynamics of the system can be described in a diffuse interface, PF model [25]. The temporal evolution of the system may be derived from the definition of the free energy functional $F[\varphi]$:

$$F[\varphi] = \int_{\Omega} g(\varphi) + \frac{\varepsilon}{2} |\vec{\nabla}\varphi|^2 dx \quad (1)$$

where Ω represents the physical domain considered, $g(\varphi)$ is the ‘bulk’ contribution to the free energy, and the gradient term $|\vec{\nabla}\varphi|^2$ is related to interface energy and width ε [24, 25]. As we are not interested in modeling a specific material, we choose for the bulk term g the common and numerically convenient double-well potential expression $g(\varphi) = \frac{18}{\varepsilon} \varphi^2(1 - \varphi)^2$ [24].

The equations of motion may be derived from a variational principle [25]. φ flows in the direction of free-energy minimization, resulting in the well-known Cahn–Hilliard equation [30, 31]:

$$\frac{\partial\varphi}{\partial t} = -\vec{\nabla} \cdot \vec{J} = M \nabla^2 (g'(\varphi) - \varepsilon \nabla^2 \varphi) \quad (2)$$

where $g' = dg/d\varphi$, M is a mobility constant and \vec{J} is φ diffusion flux, proportional to the gradient of the generalized chemical potential $\mu = \delta F/\delta\varphi$. Among PF models, the Cahn–Hilliard equation is particularly well suited as a relatively simple testing ground for advanced ML methods, as it can exhibit rich, non-linear behavior during evolution despite the relatively simple mathematical formulation, e.g. resulting in the formation and subsequent coarsening of 3D patterns, which are also interesting in the context of metamaterials design [29, 32].

The above-defined model is herein exploited to create a dataset of sequences of microstructure evolutions. Equation (2) is solved with a finite difference scheme on a cubic $64 \times 64 \times 64$ uniform grid using an explicit forward-Euler method with a constant integration step of $\delta t = 1.25 \times 10^{-3}$. M is set to unity and ε to 3 grid units. These parameters are chosen because they offer a good compromise between the accuracy of the solution of the PDE and the time required to generate the dataset. Periodic boundary

conditions (PBCs) are imposed. As starting configurations for the field, we consider periodic Perlin noise samples [33], obtained by suitable adaptation of the Python project at [34]. Perlin noise is a type of gradient noise that allows one to effectively select the typical correlation length of features in random φ states. This is in contrast with the simpler initialization with uniform, random φ values typically used, but provides coarser, grid-independent initial conditions that are easier for the CRNN to correlate with subsequent states of the evolution. Indeed, pure white noise cannot be recognized by CRNN as a sufficiently featured input, acting as an ill-defined initial condition and leading to arbitrary subsequent morphologies.

The generated training set comprises 1850 sequences with different average φ values and correlation lengths, which are then split into a 70:30 ratio, forming the training and validation sets. The resulting average value of φ , $\bar{\varphi}$, has a bell-shaped distribution with mean 1/2 and a standard deviation of $\approx 1/10$ as a result of the different Perlin noise parameters, ensuring the variability of initial conditions. Each sequence is composed of 50 consecutive configurations separated by a time interval $\tau = 0.5$ in time units (corresponding to 400 δt integration steps). Training sequences start either from the initial Perlin noise or later states in the evolution, allowing the NN to generalize to partially coarsened microstructures as initial conditions.

2.2. NN structure and training

The core architecture of the CRNN used in this work is based on [14] but for three critical modifications. First, since we are considering 3D evolutions in the present work, all convolutional layers are promoted to their three-dimensional counterparts, using the corresponding PyTorch [35] implementation. Circular padding [36] enforces PBCs, but alternative boundary conditions could be straightforwardly translated into other padding modes. Second, a deeper mapping between the input, hidden and output layers in the recurrent modules of the CRNN (see GitHub repository for technical details) is exploited to increase the representation capabilities of the network. Third, a new output layer inspired by the physical formulation of the underlying problem is introduced. In particular, our main concern here is to strongly constrain the CRNN prediction to be consistent with conservative flow dynamics of equation (2), a property that standard convolutional layers do not possess. To this end, we use a simple forward-Euler integration scheme ansatz and, instead of directly predicting the next state of the field $\hat{\varphi}_{t+\tau}$ (hat denotes predicted quantities), we define the NN output as the vector field \vec{u}_t , such that:

$$\hat{\varphi}_{t+\tau} = \hat{\varphi}_t + \vec{\nabla} \cdot \vec{u}_t \quad (3)$$

where \vec{u}_t is a vector field acting as the flux term $-\tau \vec{j}_t$ in the standard forward-Euler integration scheme of equation (2). Here $\vec{\nabla} \cdot \vec{u}_t$ is calculated using a finite difference scheme, efficiently implemented as non-trainable convolutional layers with circular padding and zero bias. Moreover, due to PBCs, \vec{j} should have a vanishing mean, which in the NN architecture is translated as removing from \vec{u}_t its mean value. While resembling a forward-Euler integration scheme, we remark that this approach is strictly more powerful, as the NN can exploit recurrence to access information farther in the evolution past, thus being more similar to a high-order integration scheme. More importantly, this procedure ensures the exact conservation of φ by construction, thus making it suitable for other diffusive processes, such as the surface diffusion case of [14], where φ conservation was enforced only weakly through the loss function. A similar strategy using an intermediate vector field was also exploited at [9] for incompressible flows.

This physics-inspired layer is in line with the well-known and used fact that good inductive biases encoded in the NN architecture lead to an easier training procedure, better generalization capabilities and a lower number of parameters, which in turn reduces overfitting risks [2, 4]. The present choice of fully convolutional layers [37] itself serves as an encoding of the translational equivariance symmetry [38] and of the locality of the evolution, allowing for applications to arbitrary domain sizes. In the supplementary material, it is shown how omitting the divergence layer degrades the predictive capabilities of the CRNN when extrapolating over long-time evolutions.

Based on our tests, the best set of hyperparameters is the following: the CRNN is composed of two stacked convolutional gated recurrent unit (GRU) [39, 40] blocks, each using 10 channels for hidden states and $3 \times 3 \times 3$ convolution kernels. A sketch of the overall CRNN architecture is shown in figure 1(a). The resulting NN model contains $\approx 2.9 \times 10^4$ trainable parameters, almost an order of magnitude less than the 2D prototype presented in [14], despite the more challenging 3D setting here tackled. Such compression is made possible by the modifications previously discussed. The training loss used is the voxel-wise mean squared error between the predicted and ground truth sequence. Training of the NN parameters is performed using the standard implementation of the Adam optimizer [41] with a learning rate of 10^{-5} . Before being passed to the CRNN, training sequences are downsampled by a factor of 1/2 in all spatial dimensions using nearest interpolation to ease GPU memory requirements during training. As it will be shown in the following, the NN prediction accuracy is not significantly impacted by such a downscale

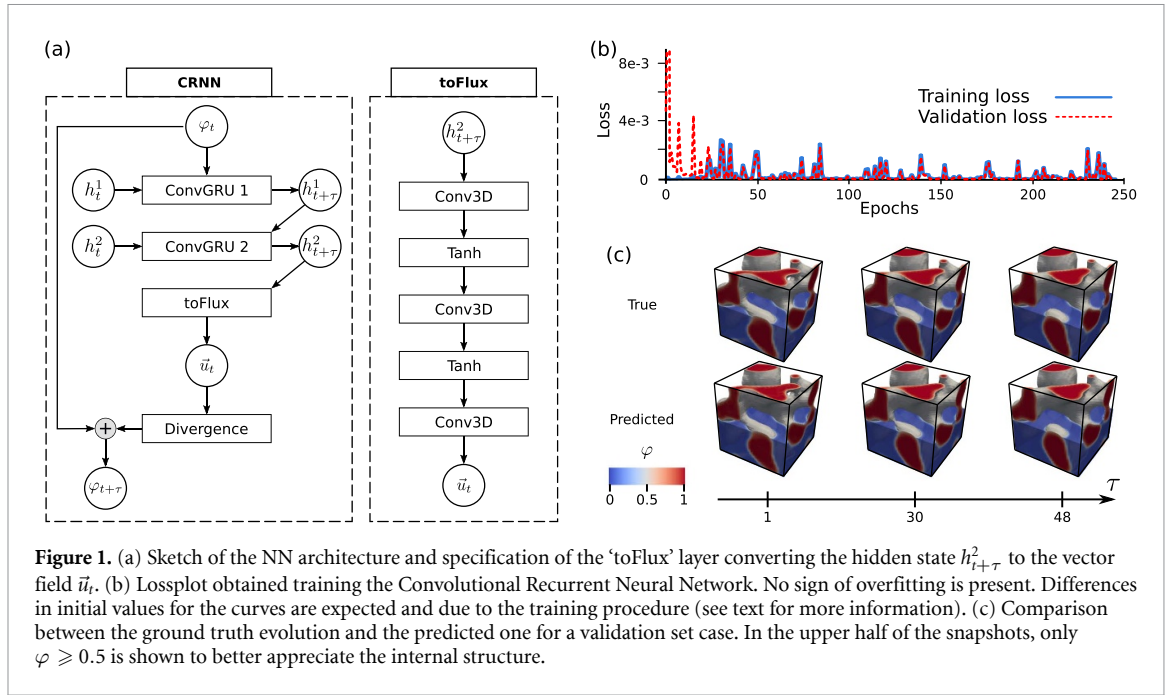


Figure 1. (a) Sketch of the NN architecture and specification of the ‘toFlux’ layer converting the hidden state $h_{t+\tau}^2$ to the vector field \tilde{u}_t . (b) Lossplot obtained training the Convolutional Recurrent Neural Network. No sign of overfitting is present. Differences in initial values for the curves are expected and due to the training procedure (see text for more information). (c) Comparison between the ground truth evolution and the predicted one for a validation set case. In the upper half of the snapshots, only $\varphi \geq 0.5$ is shown to better appreciate the internal structure.

process, which, on the other hand, can be leveraged to tackle larger computational cells at reduced computational costs, as has also been demonstrated in previous works [14, 19]. Data augmentation based on reflections, 90° degree rotations and the \mathbf{Z}_2 symmetry $\varphi \rightarrow (1 - \varphi)$ is also exploited in training. We point again readers interested in implementation details to the GitHub repository, where the full code used to perform training and evolutions here reported is available.

Figure 1(b) reports training and validation loss respectively. To perform the full training, $\approx 50h$ are required using a workstation using a single Nvidia RTX A4000 GPU. Discrepancies between the two values in the initial training stages are due to the use of a curriculum learning schedule [2, 42], which proved to speed up training convergence. Specifically, at the beginning of training the model parameters are optimized using a simplified loss function, i.e. the CRNN is required to generate only the last state provided the previous 49 ones. In the following 25 epochs, the loss function gets gradually more complex as the CRNN is required to generate progressively longer sequences. At the end of this curriculum stage, the ML model has to reproduce the full sequence based only on the initial condition.

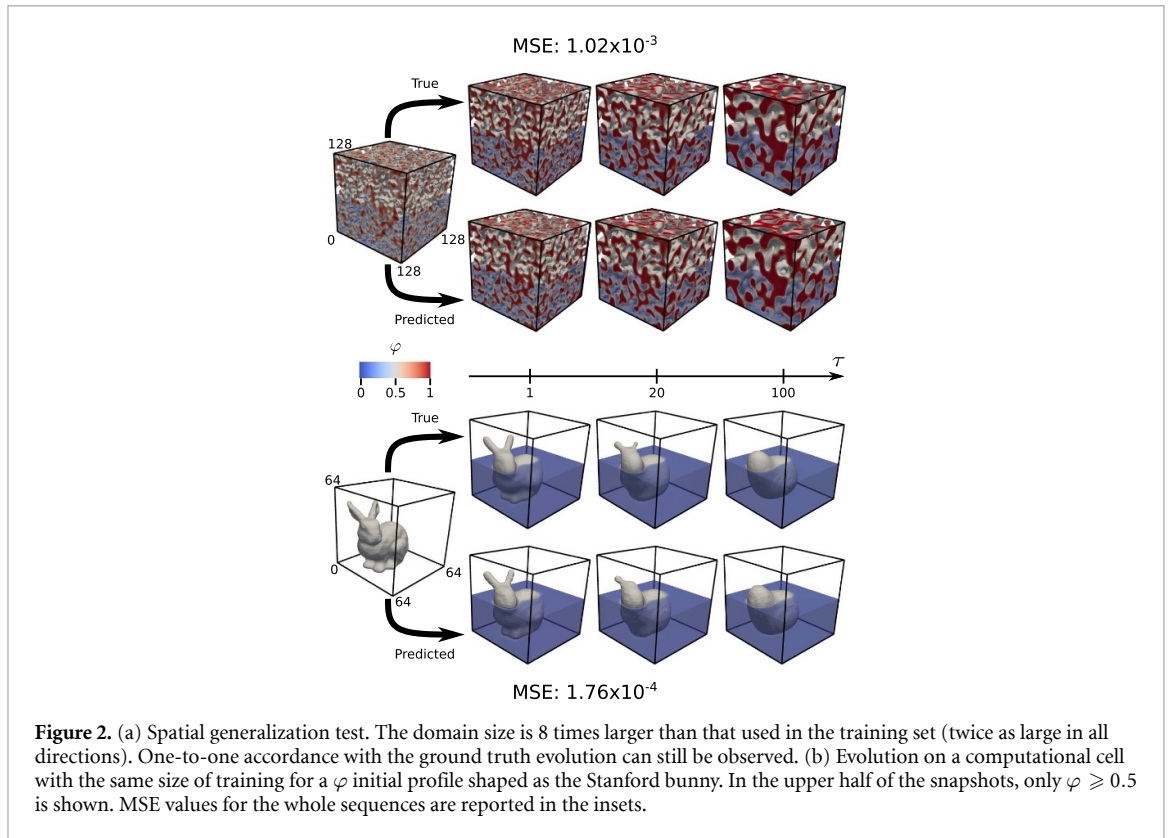
The validation loss does not increase in the late stages of training, indicating that no overfitting is present. As both training and validation losses present spikes, the model with the lowest validation loss in the last 50 epochs has been selected. A comparison between the ground-truth finite difference evolution and the one provided by the NN for a validation case is reported in figure 1(c). One-to-one correspondence in the φ maps can be observed. Notice that the typical variation between subsequent τ intervals in the training evolution is fairly localized, which further justifies the Euler-like ansatz of equation (3).

3. Results

As a good practice, after the training and validation procedure, we check the model predictivity on an independent test set with similar characteristics to the examples used during the optimization. The obtained results present the almost one-to-one correspondence already observed for the validation set in figure 1(c) and are therefore not reported. Instead, we move directly to more challenging cases, showing how the CRNN can generalize to larger computational domains and generate far longer sequences than those observed in training. In the following, we will inspect generated sequences (sections 3.1 and 3.2), discussing the quantitative aspects in the last section (3.3).

3.1. Domain size and evolution time generalization tests

One of the main advantages of a fully convolutional NN is the possibility of applying the trained model to inputs of arbitrary size. This is made possible thanks to the implicit assumption of locality embedded in the Convolutional layer choice. While for non-local PDEs this kind of procedure should be considered with care, the current application to the Cahn–Hilliard equation as defined in equation (2) does not pose any issues. To check the generalization capabilities of our CRNN in this setting, we report in figure 2(a) a computational



domain twice as large in all spatial dimensions and evolved for double the amount of time steps used for training examples. It can be observed that the NN prediction closely reproduces the finite difference evolution at all reported stages. The MSE loss on the whole sequence is also reported, exhibiting a value slightly higher than training and validation.

As an additional proof of the CRNN generalization capabilities on qualitatively different configurations, in figure 2(b) we also report the evolution under the Cahn–Hilliard flow of a domain initialized with the shape of the Stanford bunny [43], which is clearly out of the distribution of typical training set microstructures. The domain size, in this case, is the same as the $64 \times 64 \times 64$ grid used in training, but coarsening stages at 100τ are inspected as in panel (a). Close correspondence with the ground truth evolution is again achieved, despite some local variations (e.g. the discrepancy at 20τ in the left ear), which however are not preventing the close correspondence at later stages. The MSE loss, reported at the bottom of figure 2 is comparable to values in training/validation, confirming the excellent generalization capabilities of the CRNN.

We remark that, similarly to a simple forward-Euler integration scheme, computational costs scale linearly with the number of collocation points, albeit with a longer time interval between subsequent predicted states. We point the technical reader to appendix A where this aspect is quantitatively analyzed.

Another generalization test regards the possibility of generating sequences of arbitrary length by iteratively re-feeding the CRNN's own output as input to generate the following evolution stage. This aspect is particularly critical, as this recurrence may lead to prediction error accumulation and loss of reliability in the generated sequence. In principle, therefore, one could expect the evolution obtained by NN and the one coming from finite difference integration to diverge gradually. Furthermore, as dynamics progress, predicted morphologies arising from domain coarsening might be less represented in the training set. This could lead to particularly severe limitations for ML approaches, as configurations emerging from the considered PF model are subject to several constraints, in particular the conservation and boundedness of φ . Notice that, while local conservation is ensured (up to numerical precision) by the specialized, physics-inspired layer ('toFlux' and divergence operation in figure 1(b)), there is no guarantee, so far, that very long sequences will not produce invalid PF representations, e.g. with diverging φ values. To demonstrate the generalization capabilities of the proposed CRNN, in figure 3(a) we show the comparison between the predicted and ground truth evolution for an initial configuration with fluctuations localized in the central slice of a $128 \times 128 \times 128$ grid. It can be seen that there is a substantial agreement between such 'long-time' predictions from the NN and the true evolutions provided by the explicit integration scheme. Once again, despite local differences, the overall sequence and the final state of the evolution exhibit compelling

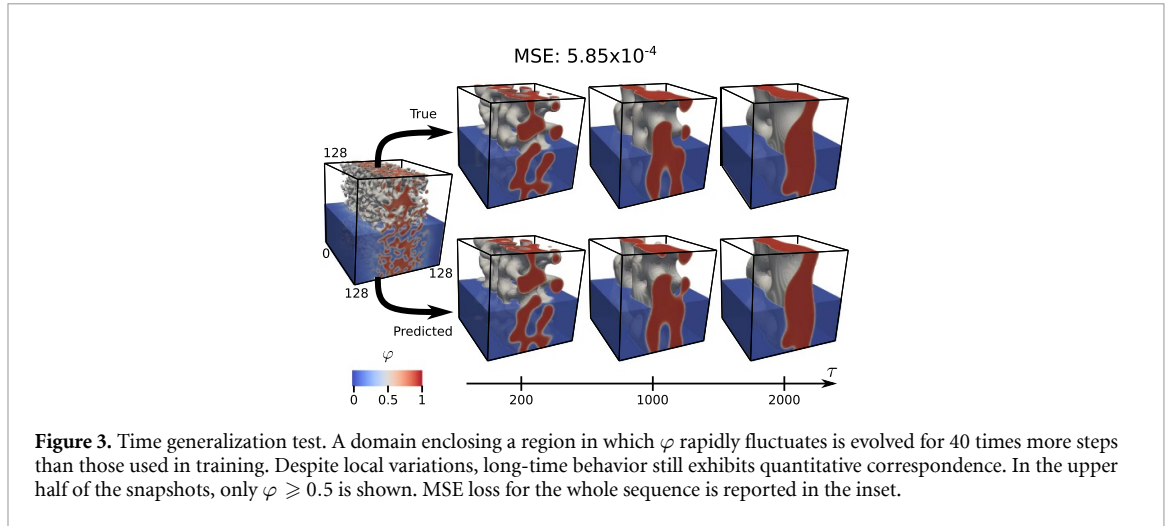


Figure 3. Time generalization test. A domain enclosing a region in which φ rapidly fluctuates is evolved for 40 times more steps than those used in training. Despite local variations, long-time behavior still exhibits quantitative correspondence. In the upper half of the snapshots, only $\varphi \geq 0.5$ is shown. MSE loss for the whole sequence is reported in the inset.

correspondence, with an MSE loss of the same order of magnitude as the one observed in the top evolution of figure 2. Notice that the complete evolution requires 8×10^5 finite difference integration steps, which correspond to a sequence 40 times longer than those contained in the training set.

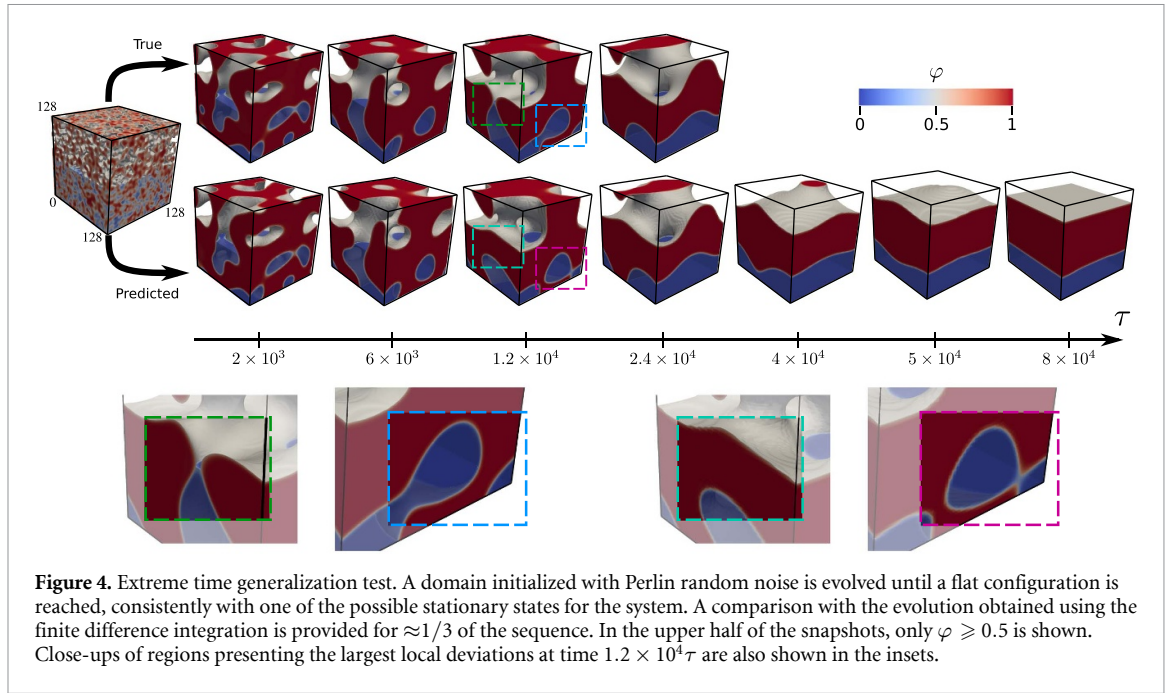
3.2. Stationary state prediction

As stated in previous sections, CRNN approaches are particularly convenient from a computational standpoint, as numerical operations can be easily parallelized on GPUs and multi-threaded systems. This allows for a strong compression of simulation wall times. Regarding examples in previous sections, the speedup obtained on an Nvidia RTX A4000 GPU running on the same machine that performed the (non-optimized) PF simulations was of $\approx 10^3$. This is only an indicative figure of merit, as, for instance, more advanced integration schemes could significantly speed up spinodal decomposition simulations [44]. On the other hand, a similar NN approach could be applied to more complex PDEs, where such advanced schemes are possibly not available or not as advantageous.

We now take profit from this computational speedup to inspect what happens if the NN model is run to generate sequences orders of magnitude longer than those provided in training. In particular, our scope is here to investigate whether the error accumulation eventually leads the NN predictions to un-physical configurations and whether the predicted stationary states (if any) are consistent with those expected from the Cahn–Hilliard model. Figure 4 reports the NN-predicted sequence of microstructure coarsening stages going from a Perlin-noise profile (with similar parameters to those used in training but not present in the dataset) to its stationary state, together with a one-to-one comparison with the corresponding finite difference integration up to $\approx 1/3$ of the evolution. Notably, the generated sequence requires 8×10^4 NN iterations, which is 4 orders of magnitude longer than those provided in training. A full simulation using the explicit integration scheme therefore requires more computational resources than those used for the whole dataset building.

Regarding error accumulation, local discrepancies reduce as dynamics progress, as expected from the Cahn–Hilliard flow minimizing the interface area between the two phases. Indeed, small dimples or corrugations in a lower curvature interface tend to be flattened by the evolution law defined by equation (2), as can be observed in the initial stages of the evolution (see e.g. fine details in the Stanford bunny in figure 2). Inspecting the configurations at time $1.2 \times 10^4 \tau$ of figure 4, it is possible to notice that the connectivity of the red domain shows significant deviations (see also close-ups provided in the insets at the bottom of the figure). At the time $2.4 \times 10^4 \tau$, however, the difference between the two configurations is far less severe, showing that the NN prediction has closed the gap with the ground-truth evolution. This testifies that the CRNN has correctly learned the underlying PF model and that no net error accumulation is pushing the predictions systematically toward un-physical configurations. This can be, at least in part, ascribed to the Recurrent nature of the NN: since the model is trained to maximize the likelihood of a whole sequence, temporary discrepancies are allowed, as long as the overall prediction gets closer again to the ground truth in the long run. A quantification of this effect is provided in section 3.3.

Figure 4 also reports a stationary state presenting a layered microstructure, which is reached at time $\approx 8 \times 10^4 \tau$ and corresponds to a time sequence 1600 times longer than those observed in training. Importantly, such configuration is stable once reached, despite being out of the distribution of the examples in the dataset. This is a striking proof of the generalization capabilities of the CRNN, as this layered



microstructure is one of the expected stationary solutions of the Cahn–Hilliard equation under the PBCs considered here [25].

3.3. Quantitative evaluation of predictive performances

Until this point, the assessment of the NN performances was only qualitative and mainly based on visual inspection. This last section addresses this issue by performing a more quantitative analysis. The long-time behavior of the system is investigated using a spinodal decomposition simulation on a $128 \times 128 \times 128$ grid, starting from a Perlin noise initial condition and $7.2 \times 10^4 \tau$ long, compatible with the one already shown in figure 4. The comparison between representative stages of the two sequences is reported in figure 5(a). To quantify the difference between the two evolutions, we report the root mean squared error (RMSE), here evaluated as (V is the computational domain volume)

$$\text{RMSE}(\varphi, \hat{\varphi}) = \sqrt{1/V \int_{\Omega} (\varphi - \hat{\varphi})^2 dx}.$$

And reported in figure 5(b).

Notice that, despite the initial condition being the same for both the NN and the ground truth evolution, the RMSE starts from a non-vanishing value. The origin of such behavior is due to the artifacts introduced by the downscaling interpolation required by the NN. The initial RMSE of ≈ 0.13 should therefore be considered ‘intrinsic’ to the downscaling procedure, but it does not affect the subsequent evolution. In appendix B, we explicitly verify this by calculating the same quantity after interpolation on a $64 \times 64 \times 64$ grid. In later evolution stages, the RMSE increases and decreases again, e.g. at $\tau \approx 1.8 \times 10^4$ and at the end of the reported evolution, when both the NN prediction and the ground truth evolution reach a layered microstructure. This shows that the accumulation of deviations is progressively recovered, as expected from the Cahn–Hilliard evolution. It is also interesting to note that the time required to reach a two-interface microstructure is consistent between the true and predicted evolution, although the CRNN is faster in smoothing interface undulations.

While the RMSE is closely related to the loss function used in training, it lacks an intuitive meaning of its numerical value. To quantify the prediction error with a variable with more direct physical interpretation, we also report in figure 5 the quantity $\bar{\delta}$, defined as:

$$\bar{\delta} = \sqrt{\frac{\int_{\Omega} (\varphi - \hat{\varphi})^2 dx}{\int_{\Omega} |\nabla \varphi|^2 dx}}. \tag{4}$$

If the ground truth φ and the prediction $\hat{\varphi}$ differ by a small displacement of the position of the interface between the two phases, equation (4) yields the root mean square value of such displacement. Appendix C

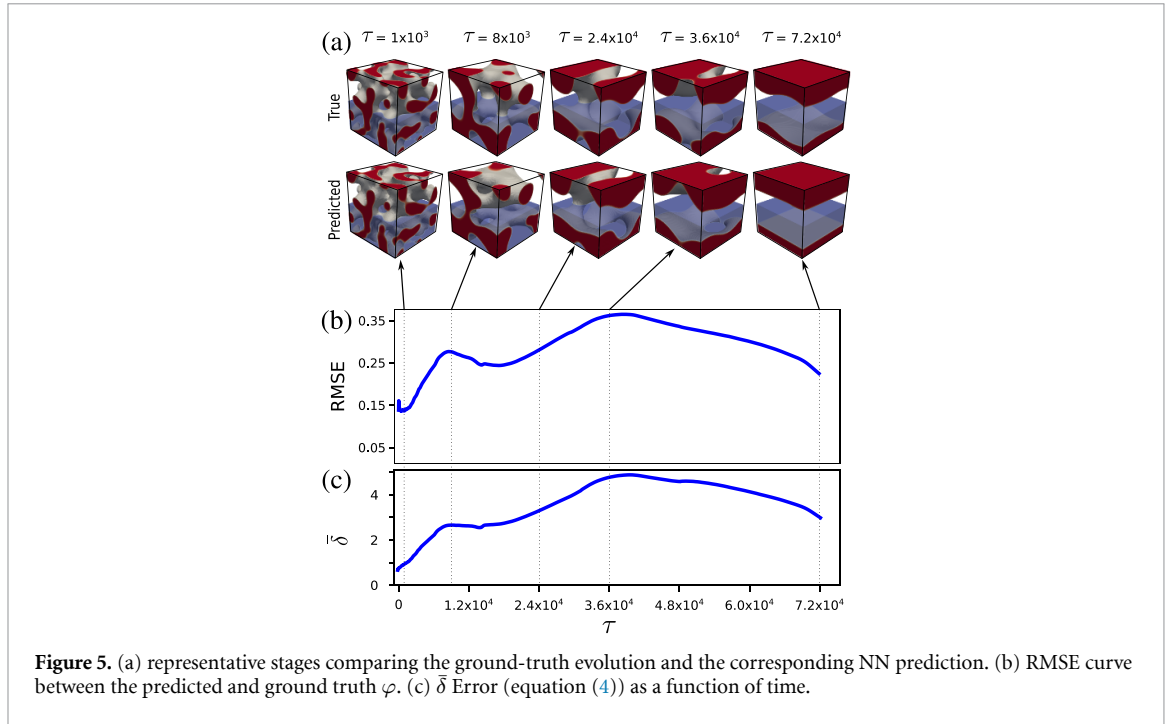
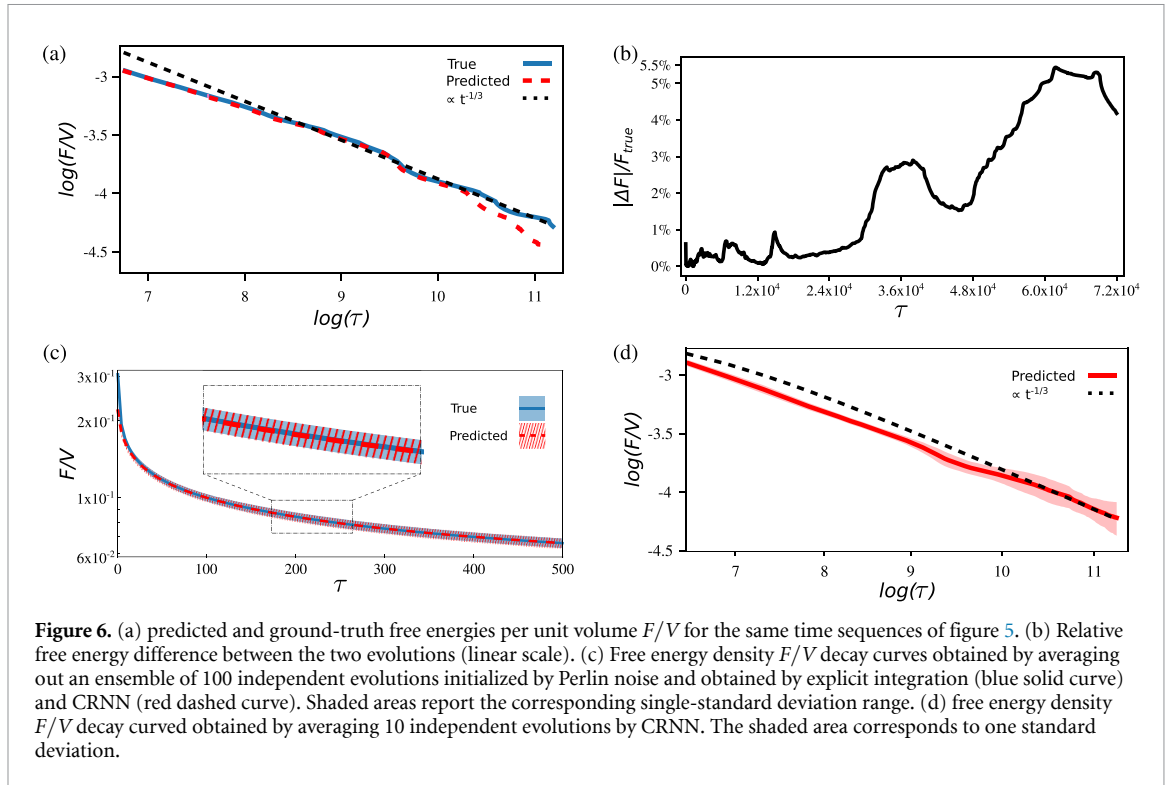


Figure 5. (a) representative stages comparing the ground-truth evolution and the corresponding NN prediction. (b) RMSE curve between the predicted and ground truth φ . (c) $\bar{\delta}$ Error (equation (4)) as a function of time.

reports a more complete discussion and a mathematical derivation. $\bar{\delta}$ can therefore be interpreted as the average, local distance between the predicted and true interface in grid units. Similarly to what was observed for the RMSE, this value increases in the intermediate stages and decreases towards the end of the reported dynamics, as the microstructure coarsens forming interfaces parallel to the computational domain sides. The initial condition high value in the RMSE is far less pronounced in terms of $\bar{\delta}$, but notice that this value refers to a random noise configuration lacking well-defined interfaces. We recall that the interface thickness parameter $\varepsilon = 3$ (see equation (2)), which means that the root mean square distance between the interfaces is comparable for most of the evolution to the width of transition regions between the two phases.

A point-by-point comparison of the evolution yields quantitative information on the NN prediction on a fine level. Indeed, both the RMSE and $\bar{\delta}$ are sensitive to local variations. However, they do not provide information on how much the predicted sequences are globally consistent with the underlying physical mechanisms. In this respect, it is crucial to assess whether the CRNN prediction remains consistent with the Cahn–Hilliard flow even when the RMSE or $\bar{\delta}$ values are at their maxima. Figure 6(a) reports the time dependence of the free energy of the system normalized on the domain volume F/N for both the NN prediction and the corresponding ground-truth evolution. A logarithmic scale is used to appreciate finer details and to analyze the power-law trends of free-energy decay. In panel (b), the relative absolute error is also reported (linear scale is used in this case). It is evident how the predicted evolution of the free energy closely reproduces the ‘true’ curve, even when the error values of figure 5(a) increase, testifying that the CRNN predictions follow the underlying free-energy minimization process. The largest deviations are observed near the late stages of the evolution ($\log(\tau) \geq 10.25$, i.e. $\tau \geq 2.8 \times 10^4$), which exhibit an acceleration in the coarsening rate with respect to the true evolution. Nonetheless, the relative absolute error in the predicted F stays below 5.5% for the whole evolution, as shown in figure 5(b).

Figure 6(a) also allows for a second quantitative analysis, regarding the coarsening rate of the microstructure. In Cahn–Hilliard evolution the typical domain length is expected to grow in time as $t^{1/3}$ and, as a consequence, the interface area per unit volume is expected to decay as $t^{-1/3}$ [45, 46]. From the definition of F in equation (1), in this work, the free energy corresponds to the interface area between the two phases. We therefore expect that $F(t) \propto t^{-1/3}$ in the long-time regime. The dashed black line reported in figure 6(a) corresponds to such behavior. Clearly, the ground truth $F(t)$ line is parallel to the expected one, indicating the same dependence for the free energy decay. Local deviations and fluctuations are related to finite domain size effects and interface splitting/merging processes. The NN prediction also follows the same power-law, up to $\log(\tau) \approx 10.25$ (again, corresponding to $\tau \approx 2.8 \times 10^4$), where the coarsening rate increases with respect to the ground truth and gets closer to a $\propto t^{-0.44}$ dependence (not shown). This behavior is probably due to the lack of long-time configurations in the training set and may be eliminated with active or transfer learning workflows, the implementation of which we leave for future works. Remarkably, while not completely satisfactory, this does not prevent the NN from predicting the correct stationary state.



To confirm the statistical significance of the above discussion, we perform a similar analysis on an ensemble of 100 evolutions, obtained using a $64 \times 64 \times 64$ grid and Perlin-noise independent initial configurations consistent with those in the training set. A total time of $\tau = 500$ is considered, as the computational effort required makes it unpractical to consider longer sequences. Figure 6(c) reports in semi-logarithmic scale the decay of the free energy obtained by averaging across all evolutions, along with its variability range, represented by the standard deviation and shown as the shaded area. A close-up of the two curves is also shown to better appreciate fine details. As it can be seen, predicted and ground-truth quantities are in close agreement, both in terms of means and standard deviations, with the largest difference in the initial stages, where the effects of the random initialization are stronger. These results confirm that the generated microstructures are quantitatively consistent with the underlying Cahn–Hilliard material flow also on an ensemble level.

We close the discussion by addressing the problem of the wrong exponent in the free energy dissipation observed at the end of the evolution in figure 6(a). While performing an extensive CRNN-ground truth comparison is computationally demanding, we can attack the problem indirectly by exploiting the low cost of ML predictions. Figure 6(d) reports in logarithmic scale the average free energy decay curve over 10 independent evolutions 80000τ long on a $128 \times 128 \times 128$ grid, each starting from a different Perlin noise initial condition. One standard deviation interval is also reported with the red-shaded area. As it can be clearly appreciated, the average long-time behavior exhibits the expected $t^{-1/3}$ power law, showing that the acceleration previously reported is not systematic. It may therefore be concluded that the NN predictions do not exhibit a net, strong deviation of the expected behavior of $F(t)$, despite the relatively short sequences used in training.

4. Conclusions

This work shows that Convolutional Recurrent NNs provide an excellent approach to approximate the microstructural evolution of materials undergoing the spinodal decomposition process in three dimensions. The use of fully convolutional and recurrent structures, together with a physics-inspired specialized layer, yields the accurate generation of spatiotemporal sequences at a fraction of the computational costs of the explicit method used to build the training set. Close correspondence for evolutions much longer than those provided in training has been demonstrated. Notably, the CRNN is found to properly predict the stationary states consistent with the learned equation, despite never being observed at train time. A main limitation of the present approach is that the trained model is only applicable to Cahn–Hilliard flow and new training, together with an ad-hoc training set, is required should the evolution law be changed. Additionally, should a

non-conservative phenomenon be considered, e.g. dendritic growth or grain coarsening, a suitable modification of the physics-inspired layer should be implemented. Due to the flexibility of NN approaches and successful applications to other microstructure evolution problems [13, 14, 18, 19], however, we expect that a suitably adapted version of the approach used in this work to be effective. On the other hand, if application to more challenging but related dynamical problems, such as evolution by surface diffusion, are to be tackled, the present model trained on Cahn–Hilliard flow could provide a good starting point for transfer learning procedures.

One of the main drawbacks of ML methods in physics is the possibility of extrapolation errors leading to inaccurate or even physically unrealistic predictions. In this respect, the present approach is shown to deliver close correspondence with the underlying physical driving force on a quantitative level, even when φ values are no longer in one-to-one accordance. These characteristics, together with the long-time stability properties, pave the way to applications to more complex physical models and show the importance of physics-driven design for CRNN architectures.

Data availability statement

The data that support the findings of this study are openly available. The dataset is available upon reasonable request from the authors. ML code implementation is available at the following URL: <https://github.com/dlanzo/CRANE>.

Acknowledgments

R B, A F and F M acknowledge financial support from ICSC—Centro Nazionale di Ricerca in High Performance Computing, Big Data and Quantum Computing, funded by European Union—NextGenerationEU.

D L acknowledges hardware support from Luca Lanzoni.

Conflict of interest

The authors declare no conflicts of interest.

Appendix A. Computational costs assessment for the NN method

Here we report the scaling of the used algorithm with respect to domain size. Figure A1 shows the wall time required to perform 1000 τ steps as a function of the total collocation points N on a workstation using an Nvidia RTX A4000 GPU. Quantities are reported as a ratio with respect to the training set used in the main text. The computational cost is linear in the number of collocation points, as expected from the parallel implementation for the convolutional layers. Notably, this is more efficient than the $O(N \log N)$ scaling of pseudo-spectral schemes.

Appendix B. Mesh rescaling effect on RMSE evaluation

We here report (figure B2) the RMSE plot of figure 5 as evaluated after downscaling the ground truth evolution on the same $64 \times 64 \times 64$ grid of collocation points used by the NN. Notice that the initial phases of the evolution now exhibit a lower RMSE value, due to the compensation of the resolution difference artifacts. In the subsequent phases of the evolution, however, values are only slightly affected, showing that the downscaling procedure is less relevant for coarser morphologies and does not significantly impact the predictive capabilities of the NN model. As the RMSE on the $64 \times 64 \times 64$ grid is smaller than the one evaluated on the $128 \times 128 \times 128$ one, we report only the latter on the main text as it provides a stricter error measure.

Appendix C. $\bar{\delta}$ Error measure

We here discuss the physical meaning of the $\bar{\delta}$ error measure reported in equation (4). First, let us consider a PF representation of a microstructure $\varphi(x)$, which implicitly defines the interface between the two phases through the $\varphi = 0.5$ isoline. Suppose that the interface displaces with respect to its original position by a small distance $\delta(S)$. S represents the intrinsic coordinates (locally) parallel to the interface. To leading order

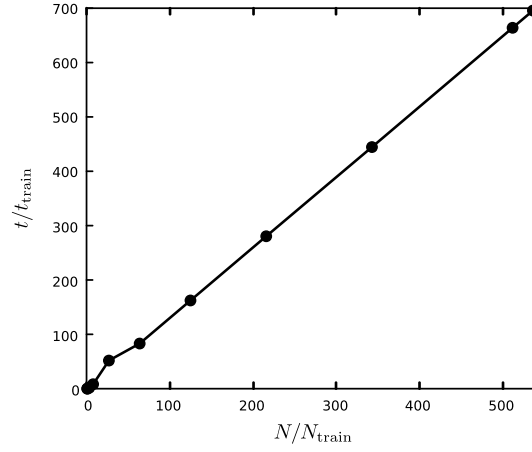


Figure A1. Computational cost for 1000 timestep prediction as a function of the number of collocation points in the domain. The number of collocation points is reported relative to the training set. $O(N)$ computational complexity can be observed. No additional optimization with respect to standard PyTorch implementation has been performed.

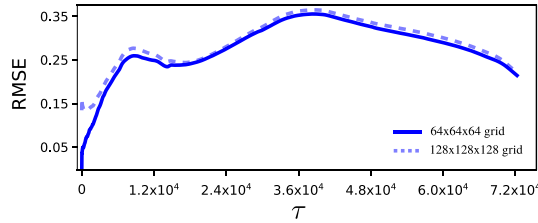


Figure B2. RMSE curve between the predicted and the ground truth φ (rescaled on a $64 \times 64 \times 64$ grid) for the same evolution of figure 5. The original curve on the $128 \times 128 \times 128$ grid is reported for comparison.

in δ , the new PF representation is given by:

$$\hat{\varphi}(x) = \varphi(x) + \hat{n} \cdot \nabla \varphi(x) \delta(S). \quad (\text{C1})$$

With $\hat{n} = \nabla \varphi / |\nabla \varphi|$ the surface normal. Then

$$(\varphi - \hat{\varphi})^2 = \delta^2(S) |\nabla \varphi|^2. \quad (\text{C2})$$

And we define the mean squared interface displacement as

$$\bar{\delta}^2 = \frac{\int_S \delta^2(S) dS}{A} \quad (\text{C3})$$

where A is the interface area and $\int_S \dots dS$ is integration on the interface manifold.

Since both $(\varphi - \hat{\varphi})^2$ and $\nabla \varphi$ are localized near the interface, we can perform integrals on the whole domain $\int_{\Omega} \dots dx$ involving these quantities using local parallel and normal coordinates S and ξ respectively. We then obtain that:

$$\begin{aligned} \frac{\int_{\Omega} (\varphi - \hat{\varphi})^2 dx}{\int_{\Omega} |\nabla \varphi|^2 dx} &= \frac{\int_S \delta^2(S) dS \int_{\hat{n}} (\partial_{\xi} \varphi)^2 d\xi}{\int_S dS \int_{\hat{n}} (\partial_{\xi} \varphi)^2 d\xi} \\ &= \frac{\int_S \delta^2(S) dS}{\int_S dS} = \bar{\delta}^2, \end{aligned} \quad (\text{C4})$$

where it is assumed in the second equality that the integral $\int (\partial_{\xi} \varphi)^2 d\xi$ does not depend on S , which is valid in the curvature \ll interface thickness limit. Equation (4) is recovered by square root.

ORCID iDs

Daniele Lanzoni  <https://orcid.org/0000-0002-1557-6411>

Andrea Fantasia  <https://orcid.org/0009-0009-1169-5083>

Roberto Bergamaschini  <https://orcid.org/0000-0002-3686-2273>

Olivier Pierre-Louis  <https://orcid.org/0000-0003-4855-4822>

Francesco Montalenti  <https://orcid.org/0000-0001-7854-8269>

References

- [1] Bishop C M 2006 *Pattern Recognition and Machine Learning* (Springer)
- [2] Goodfellow I, Bengio Y and Courville A 2016 *Deep Learning* (MIT Press)
- [3] Butler K T, Davies D W, Cartwright H, Isayev O and Walsh A 2018 *Nature* **559** 547
- [4] Mehta P, Bukov M, Wang C H, Day A G, Richardson C, Fisher C K and Schwab D J 2019 *Phys. Rep.* **810** 1
- [5] Bedolla E, Padierna L C and Castaneda-Priego R 2020 *J. Phys.: Condens. Matter.* **33** 053001
- [6] Nguyen B D, Potapenko P, Demirci A, Govind K, Bompas S and Sandfeld S 2024 *Mach. Learn. Appl.* **16** 100544
- [7] Bartók A P, Payne M P, Kondor R and Csányi G 2010 *Phys. Rev. Lett.* **104** 136403
- [8] Kocer E, Ko T W and Behler J 2022 *Annu. Rev. Phys. Chem.* **73** 163
- [9] Kim B, Azevedo V C, Thuerey N, Kim T, Gross M and Solenthaler B 2019 *Comput. Graph. Forum* **38** 59
- [10] Fulton L, Modi V, Duvenaud D, Levin D I and Jacobson A 2019 *Comput. Graph. Forum.* **38** 379
- [11] Zhang X and Garikipati K 2020 *Comput. Methods Appl. Mech. Eng.* **372** 1
- [12] Montes de Oca Zapiain D, Stewart J A and Dingreville R 2021 *npj Comput. Mater.* **7** 1
- [13] Yang K, Cao Y, Zhang Y, Fan S, Tang M, Aberg D, Sadigh B and Zhou F 2021 *Patterns* **2** 100243
- [14] Lanzoni D, Albani M, Bergamaschini R and Montalenti F 2022 *Phys. Rev. Mater.* **6** 103801
- [15] Martín-Encinar L, Lanzoni D, Fantasia A, Rovaris F, Bergamaschini R and Montalenti F 2024 Quantitative analysis of the prediction performance of a convolutional neural network evaluating the surface elastic energy of a strained film (arXiv:2405.03049 [physics.comp-ph])
- [16] Raissi M, Perdikaris P and Karniadakis G 2019 *J. Comput. Phys.* **378** 686
- [17] Bretin E, Denis R, Masnou S and Terii G 2022 *J. Comput. Phys.* **470** 111579
- [18] Peivaste I, Siboni N H, Alahyarizadeh G, Ghaderi R, Svendsen B, Raabe D and Mianroodi J R 2022 *Comput. Mater. Sci.* **214** 111750
- [19] Fan S, Hitt A L, Tang M, Sadigh B and Zhou F 2024 *Mach. Learn.: Sci. Technol.* **5** 025027
- [20] Langer J 1971 *Ann. Phys., NY* **65** 53
- [21] Kwon Y, Thornton K and Voorhees P W 2007 *Phys. Rev. E* **75** 021120
- [22] Andrews W B, Elder K L, Voorhees P W and Thornton K 2020 *Phys. Rev. Mater.* **4** 1
- [23] Jinnai H, Nishikawa Y, Morimoto H, Koga T and Hashimoto T 2000 *Langmuir* **16** 4380
- [24] Li B, Lowengrub J, Rätz A and Voigt A 2009 *Commun. Comput. Phys.* **6** 433 (available at: https://global-sci.org/intro/article_detail/cicp/7688.html)
- [25] Provatas N and Elder K 2011 *Phase-Field Methods in Materials Science and Engineering* (John Wiley Sons)
- [26] Chen L-Q 2002 *Annu. Rev. Mater. Res.* **32** 113
- [27] Boettinger W J, Warren J A, Beckermann C and Karma A 2002 *Annu. Rev. Mater. Res.* **32** 163
- [28] Albani M, Bergamaschini R and Montalenti F 2016 *Phys. Rev. B* **94** 075303
- [29] Wang Z, Dabaja R, Chen L and Banu M 2023 *Sci. Rep.* **13** 5414
- [30] Cahn J W and Hilliard J E 1958 *J. Chem. Phys.* **28** 258
- [31] Cahn J W 1965 *J. Chem. Phys.* **42** 93
- [32] Kumar S, Tan S, Zheng L and Kochmann D M 2020 *npj Comput. Mater.* **6** 73
- [33] Perlin K 1985 *SIGGRAPH Comput. Graph.* **19** 287–96
- [34] 2023 Python implementation for perlin noise (available at: <https://pypi.org/project/perlin-noise/>)
- [35] Paszke A et al 2019 Pytorch: an imperative style, high-performance deep learning library (arXiv:1912.01703 [cs.LG])
- [36] Schubert S, Neubert P, Pöschmann J and Protzel P 2019 *2019 IEEE Intelligent Vehicles Symposium (IV)* (IEEE) pp 653–60
- [37] Long J, Shelhamer E and Darrell T 2015 *Proc. IEEE Conf. on Computer Vision and Pattern Recognition* pp 3431–40
- [38] Cohen T and Welling M 2016 *Proc. 33rd Int. Conf. on Machine Learning (Proceedings of Machine Learning Research)* vol 48, ed M F Balcan and K Q Weinberger (PMLR) pp 2990–9 (available at: <https://proceedings.mlr.press/v48/cohenc16.html>)
- [39] Chung J, Gulcehre C, Cho K and Bengio Y 2014 Empirical evaluation of gated recurrent neural networks on sequence modeling (arXiv:1412.3555)
- [40] Shi X, Chen Z, Wang H, Yeung D Y, Wong W K and Woo W C 2015 *Advances in Neural Information Processing Systems* vol 2015 p 802 (available at: https://papers.nips.cc/paper_files/paper/2015/hash/07563a3fe3bbe7e3ba84431ad9d055af-Abstract.html)
- [41] Kingma D P and Ba J 2017 Adam: A method for stochastic optimization (arXiv:1412.6980 [cs.LG])
- [42] Bengio Y, Louradour J, Collobert R and Weston J 2009 *Proc. 26th Annual Int. Conf. on Machine Learning* pp 41–48
- [43] Turk G and Levoy M 2023 Zipped polygon meshes from range images *SIGGRAPH'94* pp 311–8 (available at: <https://graphics.stanford.edu/data/3Dscanrep/>)
- [44] Chen L and Shen J 1998 *Comput. Phys. Commun.* **108** 147
- [45] Kohn R V and Otto F 2002 *Commun. Math. Phys.* **229** 375
- [46] Kwon Y, Thornton K and Voorhees P 2010 *Phil. Mag.* **90** 317