A Deep Learning Approach to Nonconvex Energy Minimization for Martensitic Phase Transitions

Xiaoli Chen[a], Phoebus Rosakis[b,c,*], Zhizhang Wu[d], Zhiwen Zhang[d,*]

[a]*Department of Mathematics, National University of Singapore, 119077, Singapore*
[b]*Department of Mathematics and Applied Mathematics, University of Crete, Greece*
[c]*Institute of Applied and Computational Mathematics, Foundation for Research and Technology-Hellas, Heraklion, Greece*
[d]*Department of Mathematics, The University of Hong Kong, Pokfulam Road, Hong Kong SAR, China.*

Abstract

We propose a mesh-free method to solve nonconvex energy minimization problems for martensitic phase transitions and twinning in crystals, using the deep learning approach. These problems pose multiple challenges to both analysis and computation, as they involve multiwell gradient energies with large numbers of local minima, each involving a topologically complex microstructure of free boundaries with gradient jumps. We use the Deep Ritz method, whereby candidates for minimizers are represented by parameter-dependent deep neural networks, and the energy is minimized with respect to network parameters. The new essential ingredient is a novel activation function proposed here, which is a smoothened rectified linear unit we call SmReLU; this captures the structure of minimizers where usual activation functions fail. The method is mesh-free and thus can approximate free boundaries essential to this problem without any special treatment, and is extremely simple to implement. We show the results of many numerical computations demonstrating the success of our method.

AMS subject classification: 35R05, 65N30, 68T99, 74A50, 74G65.

Keywords: Deep learning method; variational problems; mesh-free method; phase transition problems; nonconvex energy minimization;

1. Introduction

Physics-informed deep learning methods [19, 33, 13, 14, 22, 36, 37] have recently achieved great success in solving nonlinear PDE problems that arise in different fields of STEM. This entails using deep neural networks to approximate the solution of PDEs, often in many dimensions, leading to a mesh-free numerical method in place of more established schemes, such as finite elements or finite differences.

*Corresponding author

Email addresses: xlchen@nus.edu.sg (Xiaoli Chen), rosakis@uoc.gr (Phoebus Rosakis), wuzz@hku.hk (Zhizhang Wu), zhangzw@hku.hk (Zhiwen Zhang)

In this approach [25, 36], the loss function to be minimized is the mean square of the PDE residual. This method is rigorously known to converge for linear elliptic and parabolic problems [39], and is successful with a number of nonlinear partial differential equations [7, 31]. It is not clear how it would perform in situations where there are weak solutions of the PDE that suffer from reduced smoothness. Examples are hyperbolic conservation laws, as in nonlinear elastodynamics, and solid-solid phase transitions described by nonconvex strain energy functions in nonlinear elastostatics [1, 3]. In the latter case, the associated Euler-Lagrange equations change type and lose ellipticity at some ranges of the gradient of the unknown function [1]. In these situations, there are weak solutions that are not continuously differentiable, and the solution gradient may suffer jump discontinuities across interfaces that are not known a priori but are part of the solution; they are known as phase boundaries or twin boundaries [15, 1, 3].

Another type of interface problem involves inhomogeneous media whose properties are a priori described as multiscale functions with high-contrast or discontinuous features, as in heterogeneous porous media. Sometimes the model involves a PDE with discontinuous coefficients jumping across a fixed interface. Solutions suffer the loss of smoothness across this interface.

A number of sophisticated finite element and finite difference methods for interface problems have been developed [2, 29, 17, 8, 9, 35, 27]. These methods are quite accurate, but they often require special treatment of the interface and jump conditions across it in creating the finite element mesh and/or associated basis functions for elements that intersect it.

Recently [41] the deep learning approach was used to solve interface problems involving linear elliptic PDE systems with discontinuous or high-contrast coefficients and/or discontinuous forcing. Some of these problems are motivated by phase-transition models from (piecewise) linear elastostatics applied to biomechanics [42]. The problems studied in [41] can be solved by energy minimization. The energy functional is strictly convex, and weak solutions of the PDE (Euler-Lagrange equation) are its minimizers. The Deep Ritz method [14] used in [41] approximates solutions by a deep neural network and minimizes the energy expressed as a function of the neural network parameters, known as weights and biases. The deep neural network used to approximate the unknown function does not require any special structure to deal with the interface inside the domain, and is able to capture its location spontaneously and accurately, moreover, the method is mesh-free.

In the problems considered in [41], the interface is fixed, for example as the location of the jump in the coefficients of the PDE. Also, the original energy is strictly convex and has a unique minimizer (although the energy as a function of the network parameters is typically nonconvex.)

In many phase transition problems, there is an interface separating parts of the body in different phases (solid and liquid, or austenite and martensite) whose spatial position or shape is not known a priori; it is essentially a free boundary. The shape and position of such an interface are part of the solution and can evolve in time-dependent problems. Phase boundaries (domain walls, twin boundaries, etc. in different physical settings) are

characterized by a jump discontinuity or narrow transition layer of the unknown scalar or vector field (e.g., order parameter, temperature, deformation, polarization) or some of its spatial derivatives. In order to identify the unknown phase boundary, special augmented models have been developed, notably the phase field and level set methods. These introduce additional variables to describe interface evolution. The prototypical phase field models are described by the Allen-Cahn and Cahn-Hilliard equations. A ubiquitous ingredient of most continuum models of phase transitions is a nonconvex, often multiwell energy density function. In the nonlinear elastic model of martensitic transformations and twinning, this energy is a multi-well function of the displacement gradient (or strain), or of the order parameter in phase-field models.

The deep learning approach has been used to solve free boundary problems of the Stefan type [40] and to describe phase boundaries that occur near equilibria of the Allen-Cahn and Cahn-Hilliard equations [32, 36]. The method uses standard neural networks without any modification to identify the phase boundaries that arise spontaneously for large times. These interfaces are rapid transition layers of the solution from one minimum to another of the double-well potential in the free energy density typical of these models.

Thermoelastic phase transitions [15, 3, 1] are often modeled using a free energy density that is a nonconvex, multi-well potential depending on the gradient of the unknown deformation field. In fact, the martensitic phase itself involves two or more symmetry-related variants, known as twins, that correspond to different energy wells. The fact that the energy

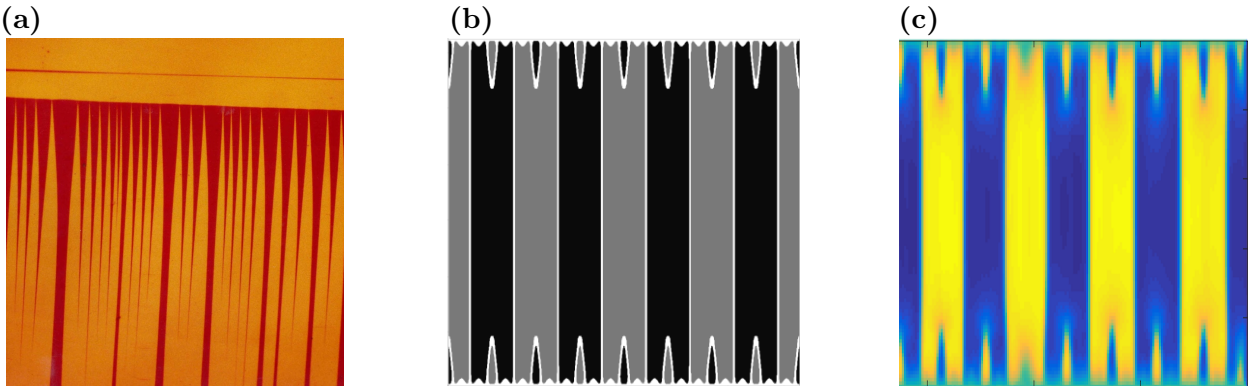

Figure 1: **Twinned Microstructures: (a)** Experimental in CuAlNi single crystal [10](reproduced here with permission of the authors). **(b), (c):** Comparison of results of different numerical methods for a simplified 2D model: (b) Finite Difference solution [20] (reproduced here with permission of the authors) and (c) our DNN solution of the 2D regularized problem (10).

depends on the gradient of the unknown function (deformation) has crucial consequences. Typical weak solutions of the Euler-Lagrange equations, or local minimizers of the energy, involve phase boundaries or twin boundaries, separating parts of the body in different phases. Across these interfaces, the deformation field remains continuous, but its gradient jumps. Continuity of the deformation field poses restrictions on the jump of its gradient in two or more directions, known as the Hadamard compatibility conditions [1]. In many crys-

talline alloys, the austenite and martensite energy wells violate these conditions, which are nonetheless satisfied between the martensitic twin variants. As a result, in some problems, there is no absolute energy minimum achievable [3]. Instead, the energy can be effectively minimized to the limit, by a sequence of deformations that involve a laminated microstructure, consisting of parallel twin interfaces between domains with gradients in alternating martensitic wells. Such a minimizing sequence involves a larger and larger number of twin boundaries, in effect converging to a mixture of phases, which in the limit becomes compatible with austenite. This explains observed interfaces, with austenite on one side, and a finely twinned, nearly periodic laminate with multiple (martensitic) twins on the other side, Fig. 1(a). This microstructure is called finely twinned martensite, and it can also occur because of incompatibility between twins of different orientations and/or rigid (null Dirichlet) boundary conditions, Fig. 1.

Microstructures observed in experiments exhibit limited fineness, with a small but nonzero distance between twin interfaces in the bulk. One of the reasons for this is that interfaces carry interfacial energy proportional to their length, which would increase with increasing fineness. So the finite spatial oscillation frequency is governed by a competition of elastic and interfacial energy. Near an incompatible boundary, the twin layers typically taper into needles, and often an individual layer is observed splitting into two or more needles in experiments, Fig. 1(a) [10, 5]. This is known as twin branching and has been studied analytically [24, 11, 20, 38] and numerically [28, 21, 20, 12] as a mechanism that allows attainment of an energy minimum in the presence of surface energy and geometric incompatibility at an interface or boundary.

Finely twinned microstructures attain considerable complexity, which poses a challenge to finite element computation [28] and analysis [24, 11, 38]. In order to study their behaviour qualitatively, it is possible to consider some lower-dimensional problems, of various degrees of sophistication, whose solutions retain increasing features of the structure of the original. Many of these problems have an energy of the form

$$\int_\Omega W(\nabla u(\boldsymbol{x}))d\boldsymbol{x}, \tag{1}$$

where $\Omega \subset \mathbb{R}^n$, $u : \Omega \to \mathbb{R}^m$ and $n = 2$ or 3, while $m = n$ or $m = 1$ for the vector and scalar case, respectively. In all cases, $W : \mathbb{R}^{m\times n} \to \mathbb{R}$ is a nonconvex function with multiple wells. There are two approaches: one entails solving the equilibrium equations, which are the Euler-Lagrange equations corresponding to (1). The other approach is to find the minimizer of (1).

Letting the derivative of W be $S : \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}$, the Euler-Lagrange (equilibrium) equation of (1) are

$$\operatorname{div} S(\nabla u) = 0 \quad \text{in } \Omega. \tag{2}$$

Because of nonconvexity, more specifically, loss of rank one convexity [3, 1], for values of ∇u between wells, the PDE system loses ellipticity for solutions that involve such values. Also because of ellipticity loss, (2) has weak solutions with phase boundaries, across which

∇u jumps discontinuously from phase to phase, and second derivatives fail to exist. This prohibits the use of the physics-informed neural-network method (PINN), which requires the strong form of (2) to be valid [36]. There does not seem to be a "weak physics-informed" neural network approach that we are aware of. An added problem is the extensive loss of uniqueness of weak solutions [15], many of which are energetically unstable (not even local energy minima). Instead, it is possible to employ minimization of the energy (1) [3, 24, 12, 38, 11]. The reincarnation of energy minimization in terms of deep neural networks is the Deep Ritz method [13], which we utilize here. An additional advantage is that piecewise smooth local minimizers of (1) are also weak solutions of (2) that are at least metastable, thus physically more acceptable than the possibly unstable weak solutions of (2).

Here we probe the ability of the Deep Ritz method to spontaneously capture complex microstructure geometries, such as the finely twinned one, with its associated multitude of gradient discontinuities across interfaces and splitting/tapering topological transitions near boundaries [20, 24, 11, 12, 38, 21].

The plan of the paper is as follows: In Section 2 we outline the Deep Ritz method for energy minimization in the context of nonlinear elasticity, and describe how to apply it to a number of problems of increasing complexity in one and two dimensions. All of these problems involve nonconvex potentials of the deformation gradient. A crucial role is played by the activation function. Our choice is guided by the fact that in the simplest one-dimensional scalar problem, an exact solution is provided by the ReLU activation function, which is piecewise linear. We then consider regularization of the problem, by including the second derivative of the deformation in the energy. Minimizers are smooth [6], with transition layers replacing derivative jumps. This motivates the introduction of a smoothened version of the ReLU activation function, which we call SmReLU, whose usefulness we test extensively in two-dimensional problems as well. In the latter, the goal is to capture spatially complex microstructures with fine twinning and multiple interfaces caused by geometric incompatibility between low energy gradients and the boundary conditions. The boundary conditions are chosen to mimic the presence of austenite. Section 3 reports on extensive numerical computations. A discussion of our results, their significance, and comparison to previous work is in Section 4.

2. Methods

2.1. The Minimization Problem

The problem we study in this paper concerns the attempt to minimize a nonconvex functional that is a low-dimensional model for an austenite-martensite phase transition. The problem involves finding

$$\min E\{u\}, \quad E\{u\} = \int_\Omega W(\nabla u(\boldsymbol{x}))d\boldsymbol{x} \tag{3}$$

over suitable functions $u : \Omega \to \mathbb{R}$, where the function $W : \mathbb{R}^2 \to \mathbb{R}$ is a two-well potential. Here we choose [24, 11]

$$W(\nabla u) = \frac{1}{2}[u_x^2(u_x - 1)^2 + u_y^2]$$

with $\nabla u = (u_x, u_y)$. The wells (minima) of W are at $(0, 0)$ and $(1, 0)$. The domain $\Omega = [0, L] \times [0, 1] \subset \mathbb{R}^2$. Here u is subject to the Dirichlet boundary conditions

$$u(x, y) = \gamma x \quad \forall\, (x, y) \in \partial\Omega. \tag{4}$$

Of particular interest is the case $\gamma = 1/2$ as the linear function $u(x, y) = x/2$ for all $(x, y) \in \Omega$ satisfying the boundary conditions has $\nabla u = (1/2, 0)$ which is a saddle point of W. The zero-energy minima $u \equiv 0$ and $u = x$ on Ω are incompatible with the boundary conditions, whereas there are sequences of functions approaching zero energy, namely problem (3) does not have a minimizer but only minimizing sequences, namely, one can construct sequences of functions u_n with $E\{u_n\} \to 0$ as $n \to \infty$. This occurs due to geometric incompatibility of the energy-minimal states $u = \text{const}$ and $u = x + \text{const}$ on Ω with the boundary conditions (4).

On the other hand the mixed boundary conditions

$$u(0, y) = 0, \quad u(1, y) = \gamma, \quad 0 < y < 1 \tag{5}$$

at the vertical sides $x = 0$, L with the horizontal sides $y = 0$, 1 free, are compatible with the ansatz

$$u(x, y) = u(x), \quad 0 < y < 1$$

(using u for both functions by notation abuse) which reduces (3) to the following one-dimensional problem. Let $\Omega = [0, 1]$, define:

$$W(z) = z^2(1 - z)^2, \quad z \in \mathbb{R}$$

so that W is a double well potential with minima at 0,1. Suppose $u : [0, 1] \to \mathbb{R}$ satisfies $u(0) = 0$ and $u(1) = \gamma$ for a given constant $\gamma \in \mathbb{R}$. Then minimize

$$\min \int_0^1 W(u'(x))dx, \tag{6}$$

with u' the derivative of u. Letting

$$f(x) = \frac{x + |x|}{2}, \quad x \in \mathbb{R} \tag{7}$$

a solution of the one-dimensional problem (6) (with zero energy) is

$$u(x) = f\,(x - 1 + \gamma)\,. \tag{8}$$

This is piecewise smooth with a derivative discontinuity at $x = 1 - \gamma$.

Remark 2.1. Any partition of $[0,1]$ into intervals where the slope of u alternates between 0 and 1 (while u remains continuous) is a minimizer (with zero energy). This shows the massive loss of uniqueness of minimizers of the 1D problem (6), but also the 2D problem with boundary conditions (5).

Remark 2.2. In order to make problem (6) well posed, it is common [6] to add a higher gradient regularization, also known as capillarity, to the energy which becomes

$$\min E_\varepsilon\{u\}, \quad E_\varepsilon\{u\} = \int_0^1 \left[W(u'(x))dx + \frac{\varepsilon^2}{2}[u''(x)]^2 \right] dx \tag{9}$$

where $\varepsilon > 0$ is the higher-gradient coefficient, a small parameter. It is known [6] that (9) has an essentially unique solution, where the gradient discontinuity of (8) is replaced by a single, smooth transition layer between values 0 and 1 of the derivative u'. For small ε the thickness of this layer is of order ε and the gradient term contributes an interfacial or surface energy to the interface, also of order ε.

The analogous 2D regularized energy is

$$E_\varepsilon\{u\} = \int_\Omega \left[W(\nabla u(x,y)) + \frac{\varepsilon^2}{2}\Big[u_{xx}(x,y)\Big]^2 \right] dxdy \tag{10}$$

2.2. Definition of deep neural networks

We briefly recall the definition and properties of a deep neural network (DNN). There are two ingredients in defining a DNN. The first one is a (vector) linear function of the form $T : \mathbb{R}^n \to \mathbb{R}^m$, defined as $T(x) = Ax+b$, where $A = (a_{ij}) \in \mathbb{R}^{m\times n}$, $x \in \mathbb{R}^n$ and $b \in \mathbb{R}^m$. The second one is a nonlinear activation function $\sigma : \mathbb{R} \to \mathbb{R}$. A frequently used activation function in the artificial neural network literature, the sigmoid function, is defined as $\sigma(x) = (1 + e^{-x})^{-1}$. Here, for reasons that will become clear below, we will use another activation function, known as the rectified linear unit (ReLU), defined by $\sigma(x) = \max(0, x)$ [26]. Observe that $\sigma(x) = f(x)$ defined in (7), which provides a solution (8) of the one-dimensional minimization problem (6). By applying the activation function to each component x_j $(j = 1, \dots n)$ of the vector $x \in \mathbb{R}^n$, one can define a vector activation function $\sigma : \mathbb{R}^n \to \mathbb{R}^n$ with components $\sigma(x_j)$, $j = 1, \dots n$.

Equipped with those definitions, we are able to construct a continuous function $F(x)$ as an alternating composition of linear transformations and (vector) activation functions as follows:

$$F(x) = T^k \circ \sigma \circ T^{k-1} \circ \sigma \cdots \circ T^1 \circ \sigma \circ T^0(x). \tag{11}$$

Here $T^i(x) = W_i x + b_i$ is an affine transformation, with the weights W_i as of yet unspecified matrices and the biases b_i unspecified vectors. Also $\sigma(\cdot)$ is the component-wise defined activation function. Dimensions of W_i and b_i are chosen to make (11) meaningful. The parametrization (11) is called a $(k + 1)$-layer deep neural network (DNN), which has k hidden layers. Denoting all the undetermined coefficients (namely W_i and b_i) in (11) as

$\theta \in \Theta$, where θ is a high-dimensional vector and Θ is the space of θ. The DNN representation of a continuous function can be viewed as

$$F = F(x; \theta). \tag{12}$$

Let $\mathbb{F} = \{F(\cdot, \theta) | \theta \in \Theta\}$ denote the set of all functions expressible by DNNs parametrized by $\theta \in \Theta$ as in (11). Then $\mathbb{F}$ provides an efficient way to represent unknown continuous functions [19], compared to a linear solution space used in classic numerical methods, e.g., a trial space spaced by linear nodal basis functions in the FEM.

2.3. The DNN representation of the nonconvex energy minimization

Given the energy functional $E\{u\}$ from (3), we can derive a DNN-based numberical method to solve the minimization problem in order to compute the solution u, in the spirit of [13]. Here we seek the solution of the energy minimization problem

$$u = \arg\min_{v \in \mathbb{H}^1(\Omega)} E\{v\}, \tag{13}$$

subject to the specified boundary condition $u(x) = g(x)$ on $\partial\Omega$.

From the perspective of scientific computing, the energy minimization problem (13) can be solved using numerical methods, such as FDMs or FEMs. From the perspective of machine learning however, the numerical solution of $u(x)$ is interpreted as a function with $x \in \mathbb{R}^2$ as its input and $\mathbb{R}^1$ as its output, that can can be approximated by a DNN $F(x)$ defined in (11). As a result, problem (13) is replaced by the problem

$$\tilde{u} = \arg\min_{F \in \mathbb{F}_b} E\{F\}, \tag{14}$$

Here $\tilde{u}$ denotes the DNN solution of the problem of minimizing the energy over the DNNs (11) in $\mathbb{F}$. Let

$$J(\theta) = E\{F(\cdot, \theta)\} = \int_\Omega W(\nabla F(x, \theta)) dx, \quad \theta \in \Theta, \tag{15}$$

where $F \in \mathbb{F}$ is as in (11), (12). Then the minimization problem (14) reduces to the (finite-dimensional) optimization problem

$$\tilde{\theta} = \arg\min_{\theta \in \Theta} J(\theta), \quad \tilde{u}(x) = F(x, \tilde{\theta}). \tag{16}$$

this yields $\tilde{u}$, the DNN representation of the solution of (14).

Notice that the original energy minimization problem (13) is non-convex. The variational problem (16) is non-convex as well. Clearly, the issue of local minima and saddle points is nontrivial, which brings essential challenges to many existing optimization methods. Since the parameter space Θ is typically very large, one often uses the Stochastic Gradient Descent (SGD) method [4] to solve (16). There are plenty of optimization methods to search among the large parameter space. Here we use the Adam optimizer, a version SGD method [23]. Our

results indicate that the SGD method is very effective in solving this non-convex optimization problem.

How to impose boundary conditions in the DNN method is an important issue [25]. For the Dirichlet boundary condition, we use the equivalent of the Nitsche method in FEM for DNN, known as the Deep Nitsche Method [30]. We add a boundary penalty term in the energy function to address this issue. Specifically, one adds a soft constraint (a boundary integral term) to the energy $J(\cdot)$ defined in (15) and obtains

$$\tilde{u}_\tau = \arg\min_{\theta\in\Theta} \Big[J(\theta) + \tau \int_{\partial\Omega} \big(F(x,\theta) - g(x)\big)^2 dx \Big], \tag{17}$$

where $\tau > 0$ is the penalty parameter to scale and balance the losses from the energy part and the boundary part. The idea is that the term $\int_{\partial\Omega} \big(F(x,\theta) - g(x)\big)^2 dx$ will approach zero when we increase the parameter τ in the calculation, whereas the Dirichlet boundary condition $u = g$ on $\partial\Omega$ will be satisfied in a weak L^2 sense, and one hopes that $\tilde{u}_\tau \to \tilde{u}$ in $\mathbb{F}$ as $\tau \to \infty$.

It has been recently pointed out that the penalty parameter τ should not be chosen arbitrarily but adaptively in a rational way stemming from the specific problem [16]. Here we also point out that the τ can be given physical meaning as the stiffness of the austenite phase that exists beyond the boundary $y = 1$, whereas the martensite phase occupies Ω. The exact satisfaction of boundary conditions such as (4) would correspond to infinitely rigid austenite.

2.4. Implementation of the Deep Ritz DNN method

Since the number $\dim\Theta$ of degrees of freedom in the optimization problem (16) is quite large, we apply the SGD method on the parameter space Θ to solve it. Notice that θ is a high-dimensional vector and let θ_k be any component of θ. We approximate the corresponding component $\partial J(\theta)/\partial\theta_k$ of the gradient of $J(\theta)$ as follows:

$$\begin{aligned} \frac{\partial J(\theta)}{\partial\theta_k} &= \int_\Omega \frac{\partial\big(W(\nabla F(x,\theta))\big)}{\partial\theta_k} dx + \int_{\partial\Omega} \frac{\partial\big(F(x,\theta) - g(x)\big)^2}{\partial\theta_k} ds \\ &\approx \frac{vol(\Omega)}{N} \sum_{i=i}^{N} \frac{\partial\big(W(\nabla F(x_i,\theta))\big)}{\partial\theta_k} + \frac{S(\partial\Omega)}{N_b} \frac{\partial\big((F(y_j,\theta) - g(y_j))^2\big)}{\partial\theta_k}, \end{aligned} \tag{18}$$

where the collocation points $x_i \overset{i.i.d.}{\sim} Unif(\Omega)$ are uniform random points that are sampled from the physical domain Ω, $y_j \overset{i.i.d.}{\sim} Unif(\partial\Omega)$ are uniform random points that are sampled from the boundary $\partial\Omega$, $vol(\Omega)$ is the volume of the domain, and $S(\partial\Omega)$ is the perimeter of the domain. In the context of the deep learning method, N and N_b are called batch numbers, which means the number of training samples utilized in one iteration.

After we compute the approximation of the gradient with respect to θ_k, we can update each component of θ as

$$\theta_k^{n+1} = \theta_k^n - \eta \frac{\partial J(F(\cdot,\theta))}{\partial\theta_k}\Big|_{\theta_k=\theta_k^n}, \tag{19}$$

where η is the learning rate. Here we use the Adam optimizer [23], a highly successful version of the SGD method.

Remark 2.3. From the derivation of the DNN formulation, one can see that the proposed method automatically deals with the phase boundary (or discontinuity in the gradient of the solution) without knowing the locations of the discontinuity *a priori*.

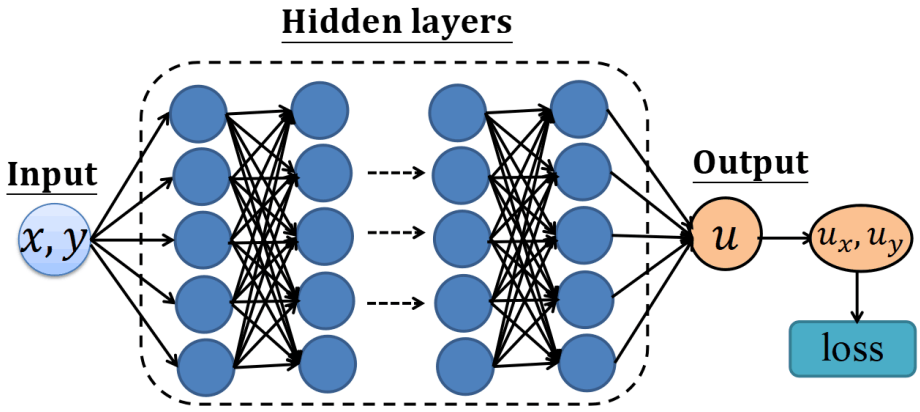

Figure 2: **Schematic of a Deep Neural Network (DNN).**

Fig. 2 shows a possible network layout for approximating u with input (x, y). For the output u, we can use automatic differentiation techniques to compute derivatives of u that appear in the energy in the minimization problem.

3. Numerical Results

We use a fully connected neural network (Fig.2) to perform our computations. The weights are initialized with truncated normal distributions with variances set as two over the sum of input and output unit numbers. The biases are initialized as zero. We use the Adam optimizer with a learning rate of 10^{-3}. In the main text, the training takes $100,000$ iterations for one-dimensional problems, while $300,000$ iterations for two-dimensional problems.

3.1. One-dimensional Case

We demonstrate how to use neural networks to minimize the energy in a one-dimensional problem. We consider the following energy minimization problem:

$$\min \int_0^1 W(u'(x))dx, \tag{20}$$

where

$$W(z) = z^2(1-z)^2 \ \ \forall z \in R, \quad u(0) = 0, \quad u(1) = \gamma. \tag{21}$$

The integrand W has two wells at 0 and 1 and we choose typically choose $\gamma = 1/2$. The functions u are approximated by deep neural networks (see Methods)

$$u_{NN}(x) = F(x; \theta),$$

where x is the input of the neural network and θ is the vector of weights w_i and biases b_i as in (11), which are the trainable parameters in the neural network.

On the one hand, the parameters θ of the neural network $u_{NN}(x)$ should minimize the energy in (20) (Deep Ritz Method [13]). We construct the corresponding loss function as follows:

$$loss_e(\theta) = \frac{1}{N}\sum_{i=1}^{N} W\left(\frac{\partial F}{\partial x}(x_i, \theta)\right), \tag{22}$$

where $\{x_i\}_{i=1}^N$ are the collocation points in Ω. The sun here approximates the integral in (20). The derivative $u'_{NN} = \partial F/\partial x$ is computed with automatic differentiation.

On the other hand, $u_{NN}(x)$ should satisfy the boundary conditions, which corresponds to minimizing the loss function

$$loss_b(\theta) = (F(0,\theta) - 0)^2 + (F(1,\theta)) - \gamma)^2. \tag{23}$$

This corresponds to the Deep Nitsche method [30] Combining the two loss functions (22) and (23), the total loss function to be minimized is

$$loss(\theta) = loss_e(\theta) + \tau loss_b(\theta), \tag{24}$$

where τ is the penalty parameter to scale and balance the losses from the energy part and the boundary part. During training, the shared parameters (θ) are adjusted by back-propagating the error obtained by minimizing a loss function (24) that is the weighted sum of the above two constraints.

3.1.1. Activation function

In this problem, an important role is played by the activation function. As observed in Methods, Section 2.1, an exact solution of (20) is provided by (8). We observe that in (8), the function $f(x)$ defined by (7) is identical with the well-known Rectified Linear Unit activation function

$$\text{ReLU:}\ \ \sigma(x) = \max\{0, x\} = \frac{x + |x|}{2}, \quad x \in \mathbb{R} \tag{25}$$

Thus by choosing one weight equal to 1 and one bias equal to $\gamma - 1$, the neural network can provide the exact solution (8) to this simple one-dimensional problem. This strongly suggests that we choose ReLU as the activation function until further notice. To test the suitability of our choice, we explore how the activation function affects the learning results. The results are shown in Fig. 3. It is clear that only ReLU captures an exact minimizer of the problem (any piecewise smooth function with slope 0 or 1 almost everywhere that satisfies the boundary conditions), whereas other choices including Tanh, Sigmoid, and even Leaky ReLU perform poorly. It is remarkable here that the Deep Ritz method with ReLU captures exact global minimizers in this simple problem, due to the piecewise smooth character of this activation function. The other activation functions struggle and are indeed unable to capture these weak solutions with derivative jumps. Later on we will propose a modification of ReLU to treat more complex problems.

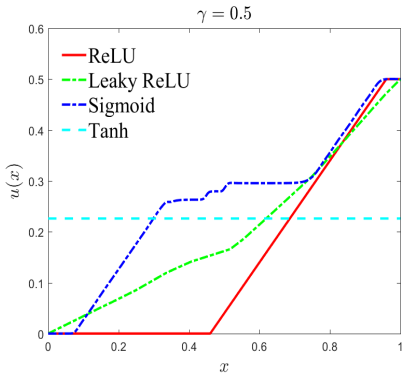

Figure 3: **Results of solving problem** (20) **with different activation function.** Shown is the plot of $u(x)$.

3.1.2. Effect of learning rate for 1D problem

We investigate how the learning rate affects the results for 3 values of the parameter $\gamma = 0.25, 0.5, 0.75$. We use the ReLU activation function. The results are shown in Fig. 4. We can see if we use very large learning rate 10^{-1} to train the loss function, the solution will go to a constant $u(x) = \frac{\gamma}{2}$, which is a local minimum of the elastic energy, but does not satisfy the boundary conditions. If we use very small learning rate, the solution will get stuck in another local minimum $u(x) = \gamma x$ for the case $\gamma = 0.25$, which however is not a global minimum, although it satisfies the boundary conditions. For intermediate values of the learning rate, a global minimum is achieved. We therefore should exercise some care in choosing the learning rate carefully.

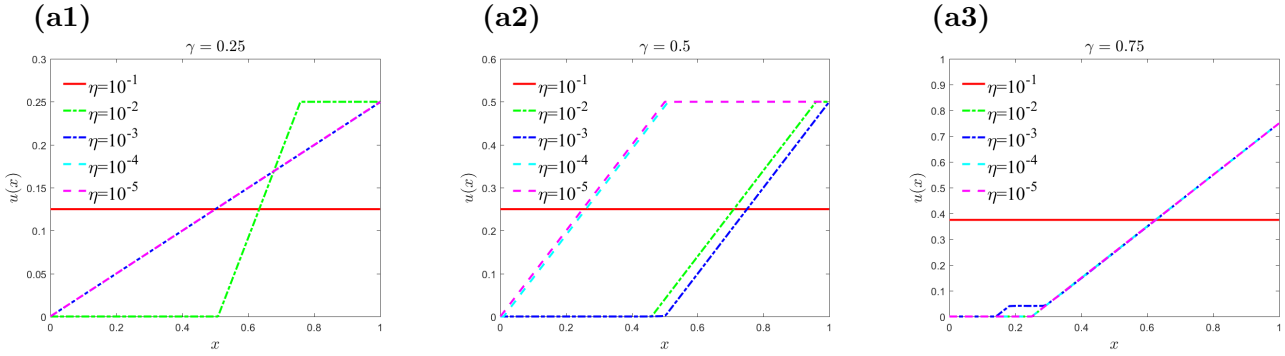

Figure 4: **1d results with ReLU activation function and different learning rates** with *iteration* = 100,000, NN: 3×128. (a) $\gamma = 0.25$; (b) $\gamma = 0.5$; (c) $\gamma = 0.75$.

3.1.3. Effect of neural network size for 1D problem

Then we explore how the depth and width affect the results when $\gamma = 0.25, 0.5, 0.75$. We use the ReLU activation function. The results are shown in Fig. 5. For small γ, we need to use a larger neural network to approximate the solution u.

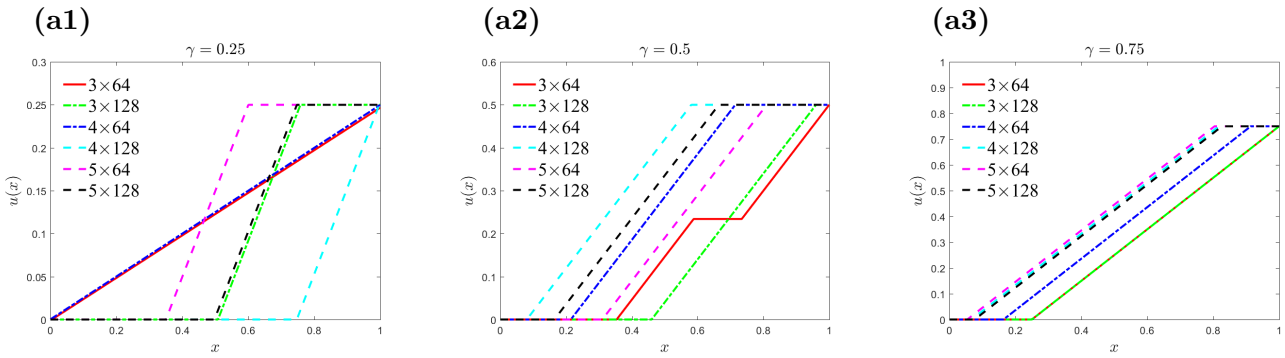

Figure 5: **1d results with different size DNN.** Here *iteration* $= 100,000$ and learning rate $\eta = 10^{-2}$. (a) $\gamma = 0.25$; (b) $\gamma = 0.5$; (c) $\gamma = 0.75$.

3.1.4. Regularized 1D problem

In order to cure the massive loss of uniqueness of minimizers for problem (20) it is common to add higher gradients to the energy; see Remarks 2.1 and 2.2, also [6] Thus we consider the minimization problem

$$\min E_\varepsilon\{u\}, \quad E_\varepsilon\{u\} = \int_0^1 \left[W(u'(x))dx + \frac{\varepsilon^2}{2}[u''(x)]^2 \right] dx \tag{26}$$

where $\varepsilon > 0$ is a small parameter, with boundary conditions $u(0) = 0$ and $u(1) = \gamma$.

It is well known that the higher gradients in the energy (26) smoothen derivative discontinuities, and replace them with a smooth transition whose width is of order ε, e.g. [6]. Thus we expect that the DNN with a ReLU activation function will not be able to capture this accurately due to its lack of smoothness. This leads us to construct a new activation function by smoothening ReLU. In (25), replace the absolute value $|x| = \sqrt{x^2}$ by $\sqrt{x^2 + \rho^2}$, where ρ is a small parameter. We call the result a Smoothened Rectified Linear Unit or

$$\text{SmReLU: } \sigma(x) = \frac{x + \sqrt{x^2 + \rho^2}}{2}, \quad x \in \mathbb{R}. \tag{27}$$

SmReLU is a smooth function for $\rho > 0$; it reduces to ReLU for $\rho = 0$. We typically use $\rho = 0.1$ in computations.

Next, we consider how the initial condition, the value of the higher gradient coefficient ε, and the activation function (ReLU vs SmReLU) affect our computations.

The results are shown in Fig. 6. We use a 3×128 neural network with ReLU and SmReLU activation function to approximate the solution u. The learning rate is $\eta = 10^{-2}$ and $\tau = 500$.

If we use the ReLU activation function, the optimal u is not differentiable at some points, moreover, more than one derivative jumps (kinks) occur when the initial condition has oscillations; see Fig. 6. On the other hand, problem (26) has a unique smooth minimizer (modulo reflection about $x = 1/2$) [6] with a single smoothened kink (transition layer between slopes 0 and 1). This is well captured by the SmReLU DNN, which gives solutions with one

smooth kink. The only exception which has a local minimum with two kinks occurs in Fig. 6 (a4). For larger ε, the DNN learns a more smooth solution with a wider transition layer, in agreement with theory [6]. If ε is larger enough, the regularization term dominates the loss function. So to minimize the second derivative energy term, u is very close to a linear function for this case. We conclude that the SmReLU activation function (27) we proposed works well for this problem and captures the behavior expected from theory.

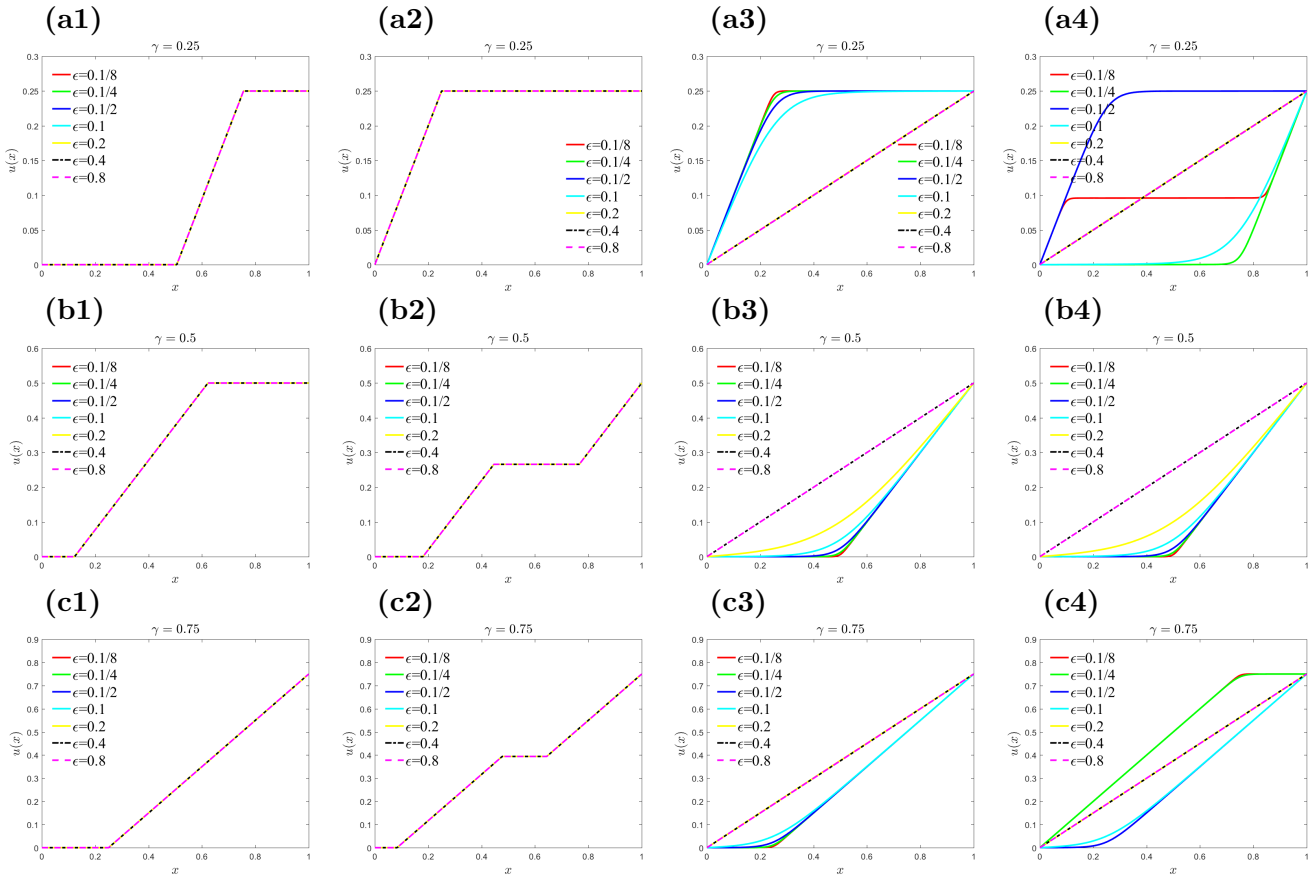

Figure 6: **1d regularized problem results.** Here *iteration* $= 100,000$, $\eta = 10^{-2}$ and NN: 3×128. (a) $\gamma = 0.25$; (b) $\gamma = 0.5$; (c) $\gamma = 0.75$. First column: Random initial condition and ReLU activation function; Second column: $\gamma x + 0.1\sin(4x)$ initial condition and ReLU activation function; Third column: Random initial condition and SmReLU activation function; Last column: $\gamma x + 0.1\sin(4x)$ initial condition and SmReLU activation function.

3.2. Two-dimensional problem

Our main goal is to consider the following two dimensional nonconvex problem:

$$\min \int_{\Omega} W(\nabla u(x,y))dxdy, \tag{28}$$

and its regularized counterpart (33). Here

$$W(\nabla u) = \frac{1}{2}[u_x^2(1-u_x)^2 + u_y^2] \tag{29}$$

and $\Omega = [0, L] \times [0, 1]$ with $L = 1$ or 2. We construct a fully connected neural network for u with input (x, y). The loss function is defined as follows:

$$loss = loss_e + \tau loss_b, \tag{30}$$

where

$$loss_e = \frac{1}{N}\sum_{i=1}^{N} W(\nabla u_{NN}(x_i, y_i)),$$
$$loss_b = \sum_{j=1}^{N_b} \frac{1}{N_b}(u_{NN}(x_j^b, y_j^b) - u_{true}(x_j^b, y_j^b))^2,$$

$\{(x_i, y_i)\}_{i=1}^N$ are the collocation points in the domain Ω, $\{(x_j^b, y_j^b)\}_{i=1}^{N_b}$ are the collocation points at the parts of the boundary where Dirichlet boundary conditions are specified, while τ is the weight to scale and balance the losses from the energy and the boundary conditions.

3.2.1. Mixed Boundary Conditions

Consider the mixed boundary conditions:

$$u(0, y) = 0, u(1, y) = \gamma, \quad 0 < \gamma < 1, \tag{31}$$

The other two sides are free. Functions u that are independent of y minimize the energy provided they are solutions of the one-dimensional problem (20). We use both ReLU and SmReLU activation function, and 3×128 neural networks to approximate u when $\gamma = 0.25,\ 0.5,\ 0.75$. The learned solutions are shown in Fig. 7. In the top row of Fig. 7, we show results from the ReLU DNN. The bottom row of Fig. 7, shows the SmReLU DNN minimizers. In the latter case solutions only have one gradient jump, whereas ReLU solutions typically have two.

Our numerical results match the exact y-independent solutions of the one-dimensional problem.

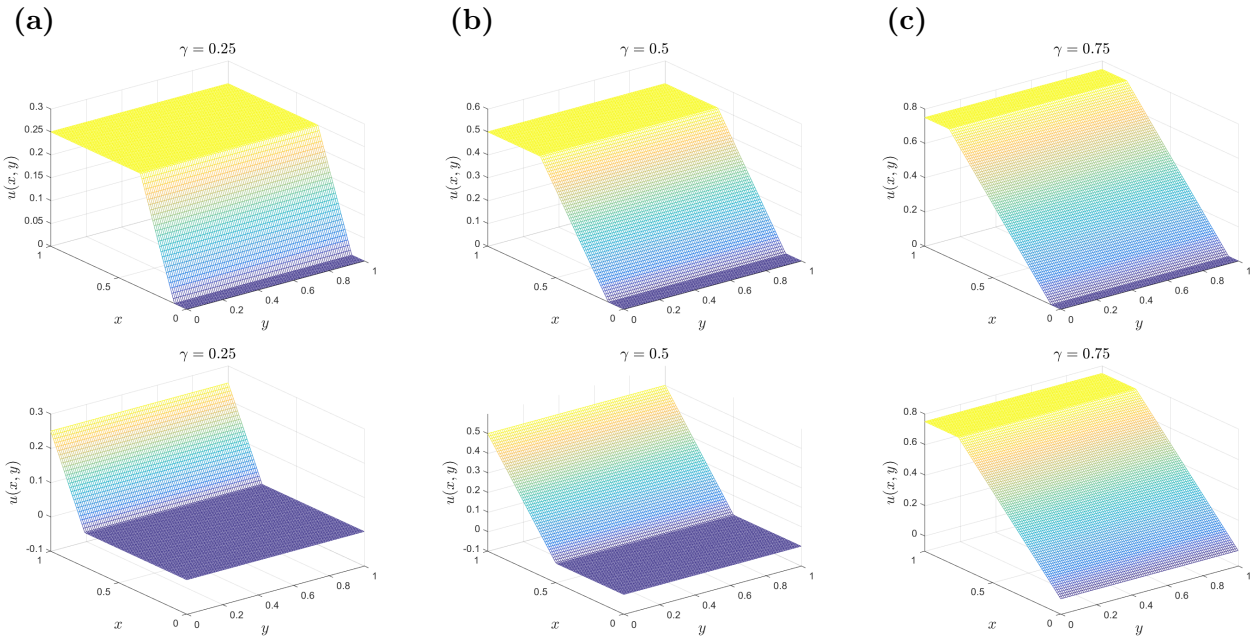

Figure 7: **2d mixed boundary conditions.** (a) $\gamma = 0.25$; (b) $\gamma = 0.5$; (c) $\gamma = 0.75$. Top: ReLU; down: SmReLU activation function.

3.2.2. Dirichlet Boundary Conditions

In the following, we will explore Dirichlet boundary condition

$$u(x, y) = \gamma x \quad \forall (x, y) \in \partial\Omega. \tag{32}$$

A typical value of interest is $\gamma = 1/2$. In order to get a zero energy deformation u_x should take values 0 or 1 with $u_y = 0$, which is what happens in Fig. 7, the mixed boundary condition case considered in subsection 3.2.1. On the other hand, here the boundary conditions (32) dictate that u_x be close to $1/2$ near the boundary so there is an incompatibility between the energy wells and the boundary conditions. In fact it is possible to show that problem (28) subject to (32) does not have a minimizer [3, 24]. Instead, the energy can be effectively minimized in the limit by a sequence of deformations that involve a laminated microstructure, consisting of vertical parallel interfaces between domains with gradients in alternating wells $(u_x, u_y) = (0, 0)$ and $(1, 0)$. Such a minimizing sequence involves a larger and larger number of twin boundaries (jumps in ∇u along lines $x =$const.), in effect converging to a mixture of phases, where in the limit u_x converges weakly to the average value $1/2$, so that it becomes compatible with the boundary conditions $u = x/2$ on $\partial\Omega$.

We show results using the SmReLU activation function and $\gamma = 0.5$ in Fig. 8 and 9. Laminated microstructures prevail with alternating bands having u_x jumping between values near 0 (blue) and 1 (yellow) to minimize the energy density W, except near horizontal boundaries where these values are incompatible with boundary conditions (32) that require $u_x = 1/2$. This causes each band to split into two or more near the boundary as also observed in experiments Fig. 1(a) [10]. In Fig. 8, we study the effect of neural network

depth (number of layers) in capturing local minima with more and more bands. With one layer only, the solution is stuck at an unstable state (saddle point of W) with $u_x \approx 0.5$, Fig. 8(a). The number of bands increases as we increase the number of layers (depth) of the DNN, albeit with some oscillations. The energy also decreases and then appears to tend to a constant. In Fig. 9, we fix the depth as 5 and increase the width of the neural networks. The energy will decrease for wider neural networks. If we use larger neural networks (deeper or wider), we can get more refined microstructures in our results, but we recall that there is no limit to the number and fineness of bands in this problem all the way to infinity [3, 24], as there is no global minimum, but apparently a sequence of local minima with an increasing number of bands, like the ones observed here, that approach the infimum of energy. In a wider network, there will be more parameters (weights and biases) to be trained, rendering optimization costlier.

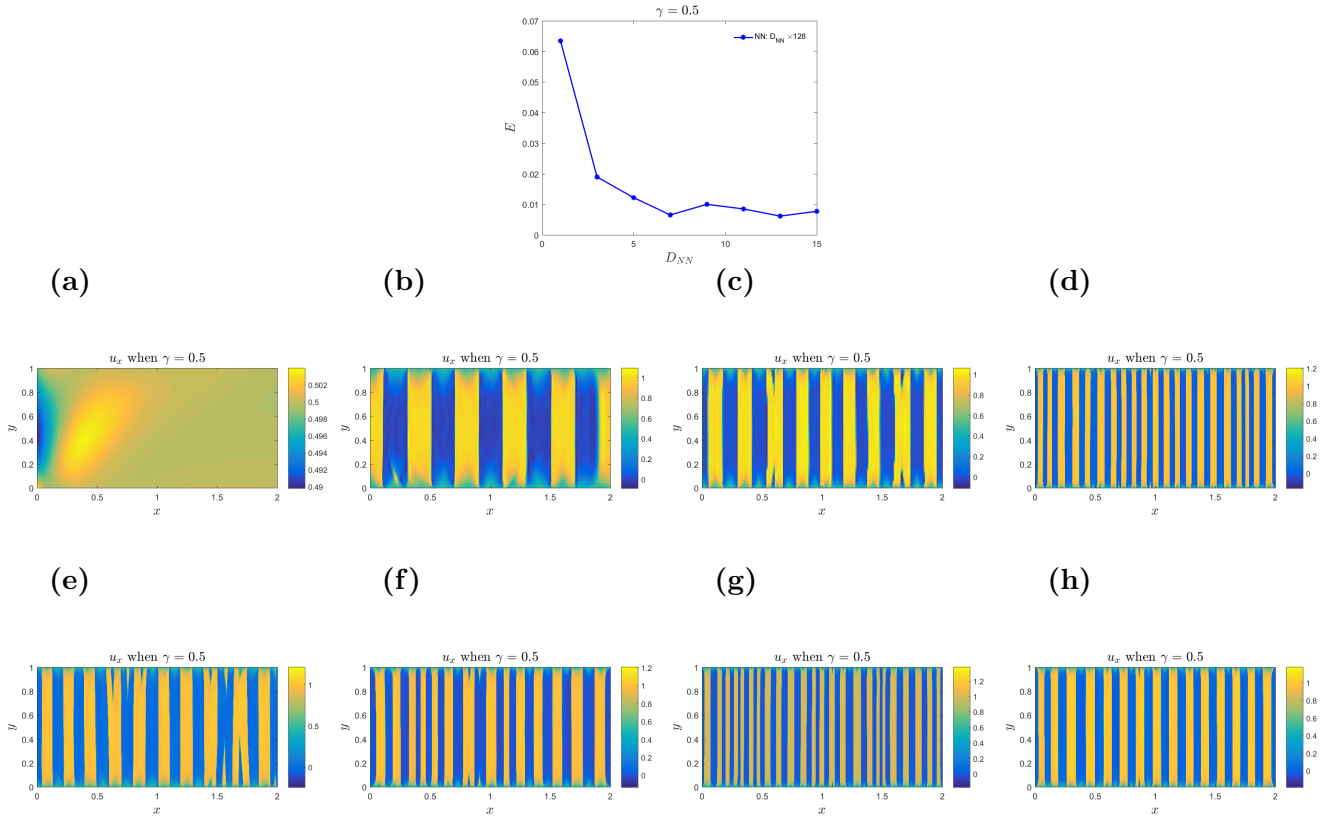

Figure 8: **2d Dirichlet boundary condition with SmReLU activation function.** Here $\gamma = 0.5$, $iteration = 300,000$, $\eta = 10^{-3}$.(a) NN: 1×128; (b) NN: 3×128; (c) NN: 5×128; (d) NN: 7×128; (e) NN: 9×128; (f) NN: 11×128; (g) NN: 13×128; (h) NN: 15×128.

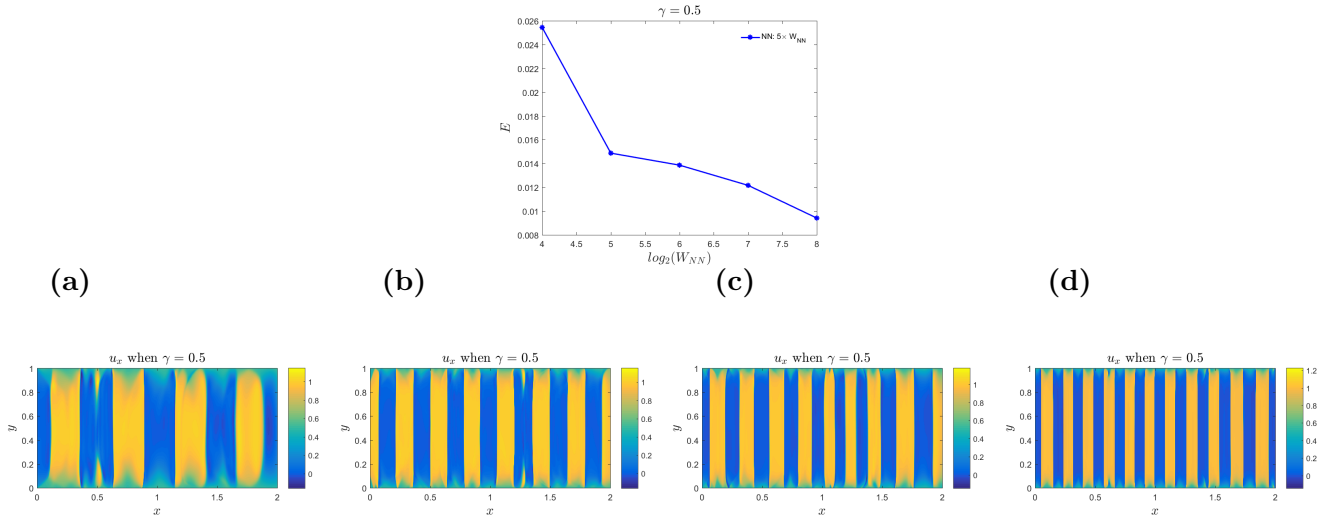

Figure 9: **2d Dirichlet boundary condition with SmReLU activation function.** Here $\gamma = 0.5$, $iteration = 300,000$, $\eta = 10^{-3}$. Top: the energy with different (a) NN: 5×16; (b) NN: 5×32; (c) NN: 5×64; (d) NN: 5×256.

3.2.3. Regularized Scalar Problem with Dirichlet Boundary Conditions

In orfer to study a problem that actually has a global minimum we use the common approach af adding higher gradients and studying the regularized problem

$$\min \int_\Omega \left[W(\nabla u(x,y)) + \frac{\varepsilon^2}{2} \left[\frac{\partial^2 u(x,y)}{\partial x^2} \right]^2 \right] dxdy, \tag{33}$$

where W is given by (29), $\Omega = [0,2] \times [0,1]$ and the boundary condition is the linear Dirichlet one (32). The higher gradients render local minima smooth and gradient discontinuities become smooth transition layers with width of order ε [6, 20, 24, 12, 11]. Also there is a global minimizer, but a careful bifurcation analysis [20] has revealed multitudes of stable local minima with multiple bands. A comparison of one of our solutions with one of the equilibria found in [20] is shown in Fig. 1(b), (c).

We explore how the depth and width of the neural network affect the energy. The results are shown in Fig. 10-12. where $\gamma = 0.25, 0.5, 0.75$ respectively. From the energy results, if we use a larger neural network, the energy will be lower, since the greater number of parameters allows capturing a local minimum with lower energy and eventually hopefully a global minimum.

The energy decreases when the regularization parameter ε is smaller, while solutions for smaller ε have a finer microstructure with more bands, e.g., Fig. 11, as also observed in [20].

3.2.4. Effect of the structure of the neural networks

Here, we fix the width of the neural network as 128 and $\varepsilon = \frac{1}{160}$. We explore how the depth affects the results, which are shown in Fig. 13. Once again, one layer is insufficient as it does not capture the banded microstructure. The energy decreases with the increasing depth of the neural network after some oscillations.

The number of bands more or less increases with increasing depth, the maximum observed being 7 yellow bands. This is probably because the global minimum is probably reached, although this is difficult to prove. More bands would increase the interfacial energy of transition layers, which is proportional to ε times the total length of transition layers between the blue and yellow phases.

We also see a refinement mechanism in Fig. 13(b4) where a yellow band near the middle is being split almost completely into two bands. The energy eventually plateaus out as the depth increases.

In order to explore how the width affects the results, we fix the depth of the neural network as 5 and $\varepsilon = \frac{1}{160}$. The results are shown in Fig. 14. We can see the energy and the number of the yellow bars will decrease with increasing width of the neural network. For a fixed depth of 5 layers of the DNN, increasing the width beyond 128 does not result in more than 5 yellow bands in our solutions.

3.2.5. Collocation Point Placement

In computed minimizers we observe that the energy density near the boundaries is higher, whereas we use a random choice of collocation points in the domain Ω. In order to minimize

the energy contribution near the boundary more effectively, we select adaptively a higher proportion of collocation points near the top and bottom parts of domain. We divide the domain Ω into three parts, $\Omega_1 = (0,2) \times (0,0.15)$, $\Omega_2 = (0,2) \times (0.15,0.85)$ and $\Omega_3 = (0,2) \times (0.85,1)$. And in each domain, we randomly choose N_1, N_2 and N_3 points, so that $N_1 = N_2$. For details see Fig. 15. In our computation, we set $N_1 = N_2$. When we increase the number $N_1 = N_2$, keeping the sum of N_1, N_2 and N_3 fixed, the energy decreases.

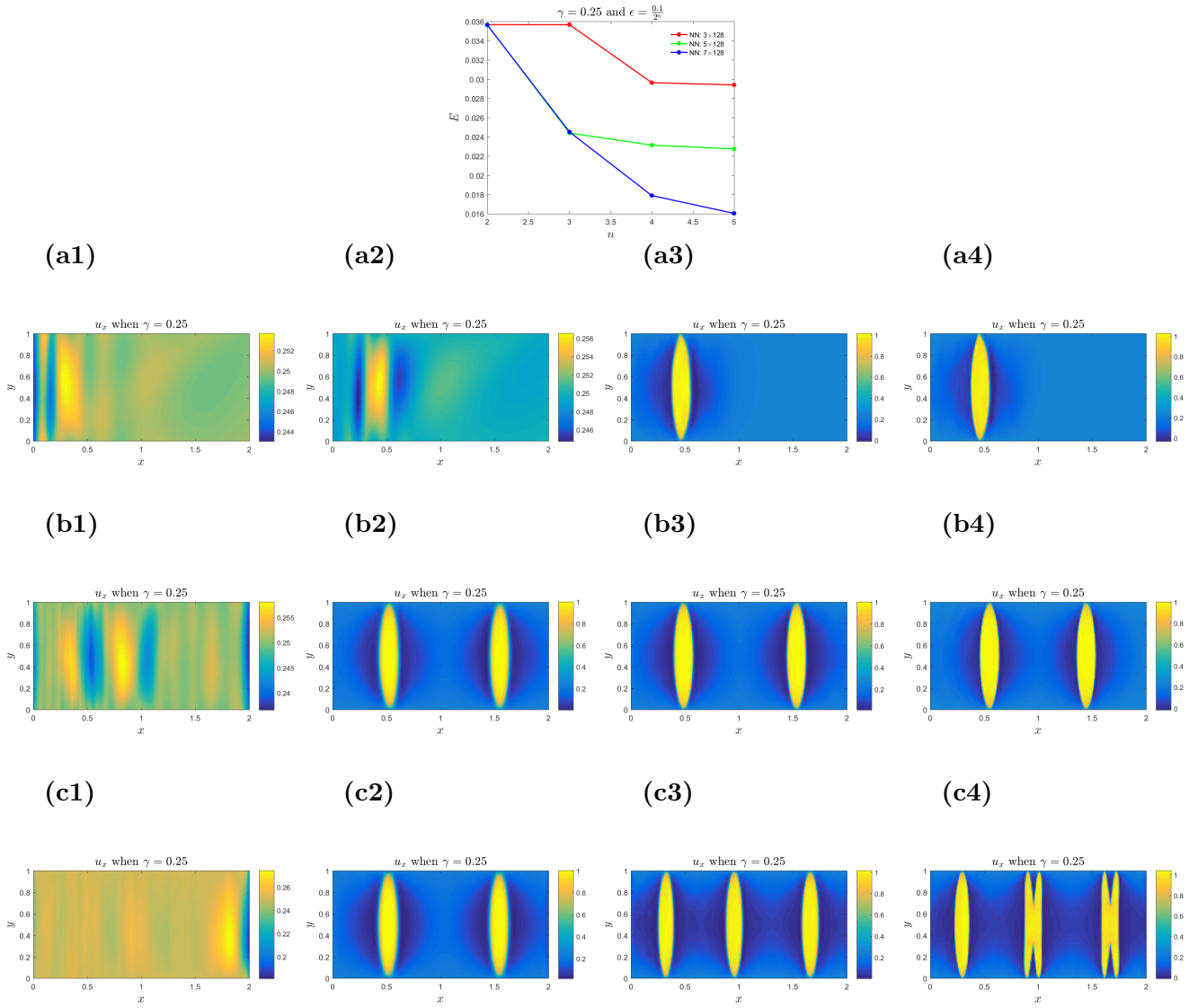

Figure 10: **2d regularized problem with SmReLU activation function.** Here $\gamma = 0.25$, *iteration* $= 300,000$ and learning rate $\eta = 10^{-3}$. (a) NN: 3×128; (b) NN: 5×128; (c) NN: 7×128. First column: $\varepsilon = \frac{0.1}{4}$; Second column: $\varepsilon = \frac{0.1}{8}$; Third column: $\varepsilon = \frac{0.1}{16}$; Last column: $\varepsilon = \frac{0.1}{32}$.

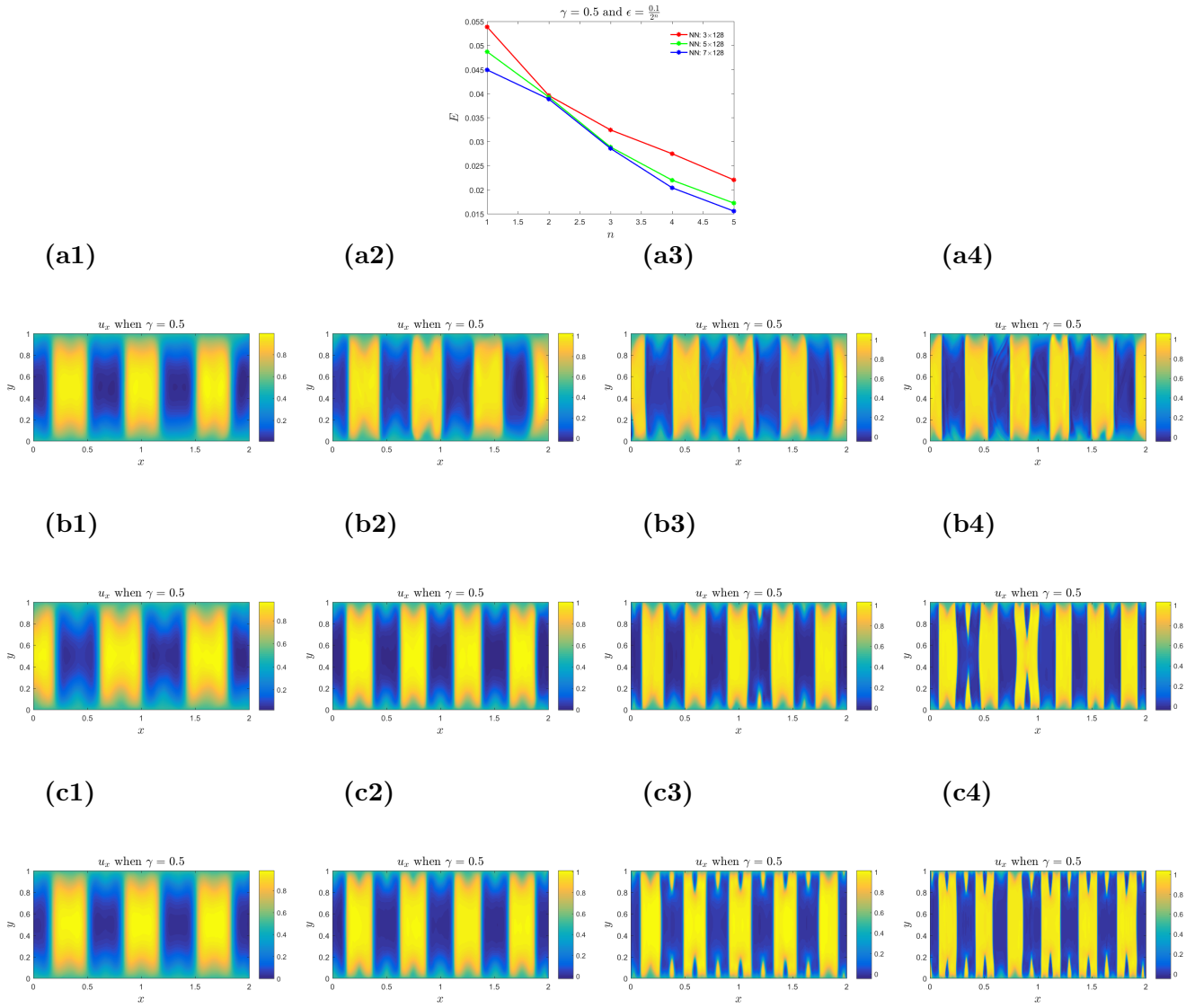

Figure 11: **2d regularized problem with SmReLU activation function.** Here $\gamma = 0.5$, *iteration* $= 300,000$ and learning rate $\eta = 10^{-3}$.(a) NN: 3×128; (b) NN: 5×128; (c) NN: 7×128. First column: $\varepsilon = \frac{0.1}{4}$; Second column: $\varepsilon = \frac{0.1}{8}$; Third column: $\varepsilon = \frac{0.1}{16}$; Last column: $\varepsilon = \frac{0.1}{32}$.

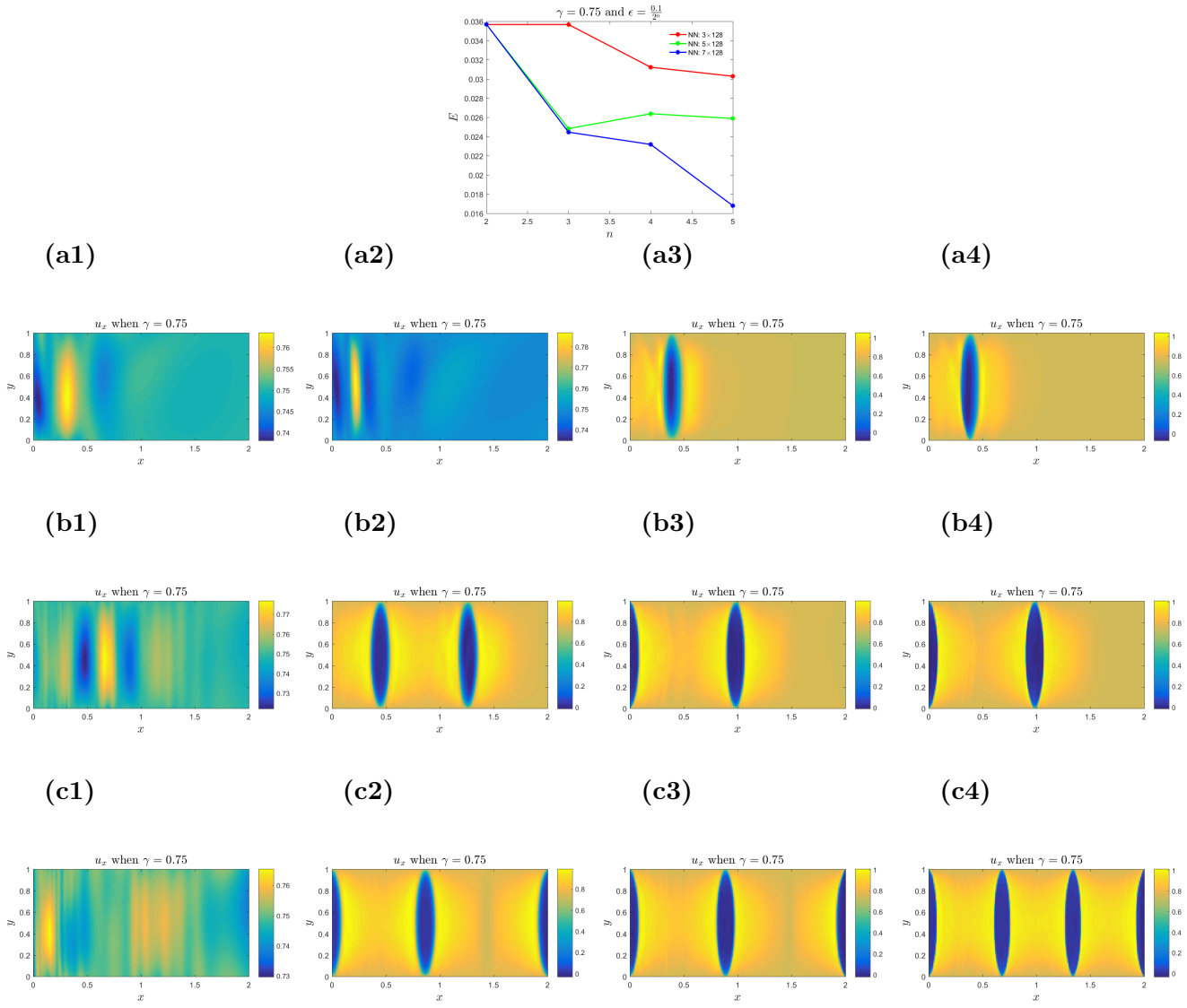

Figure 12: **2d regularized problem with SmReLU activation function.** Here $\gamma = 0.75$, *iteration* $= 200,000$ and learning rate $\eta = 10^{-3}$. (a) NN: 3×128; (b) NN: 5×128; (c) NN: 7×128. First column: $\varepsilon = \frac{0.1}{4}$; Second column: $\varepsilon = \frac{0.1}{8}$; Third column: $\varepsilon = \frac{0.1}{16}$; Last column: $\varepsilon = \frac{0.1}{32}$.

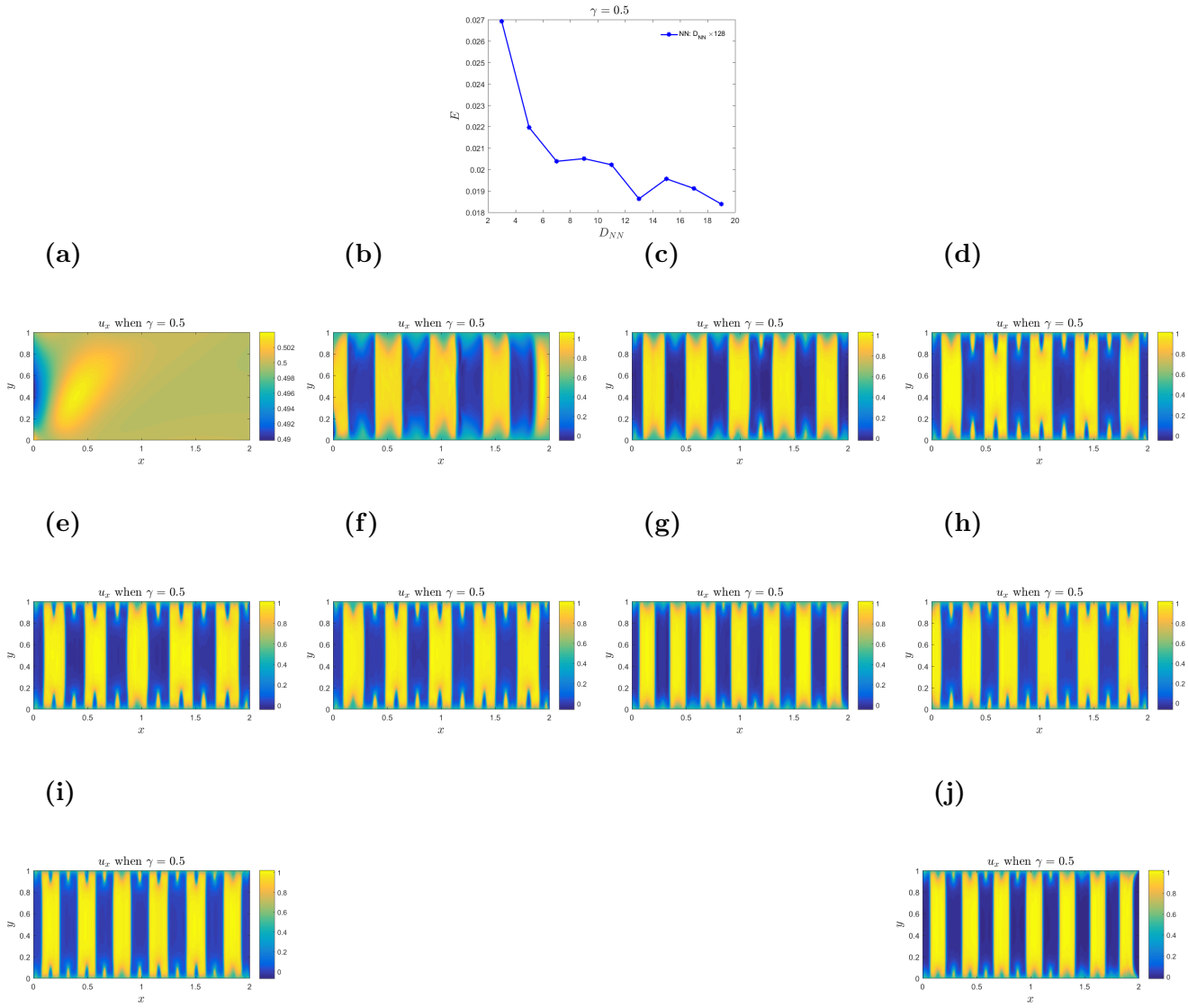

Figure 13: **2d regularized problem with SmReLU activation function.** Here $\gamma = 0.5$, *iteration* $= 300,000$, $\eta = 10^{-3}$ and $\varepsilon = \frac{0.1}{16}$. (a) NN: 1×128; (b) NN: 3×128; (c) NN: 5×128; (d) NN: 7×128; (e) NN: 9×128; (f) NN: 11×128; (g) NN: 13×128; (h) NN: 15×128; (i) NN: 17×128; (j) NN: 19×128.

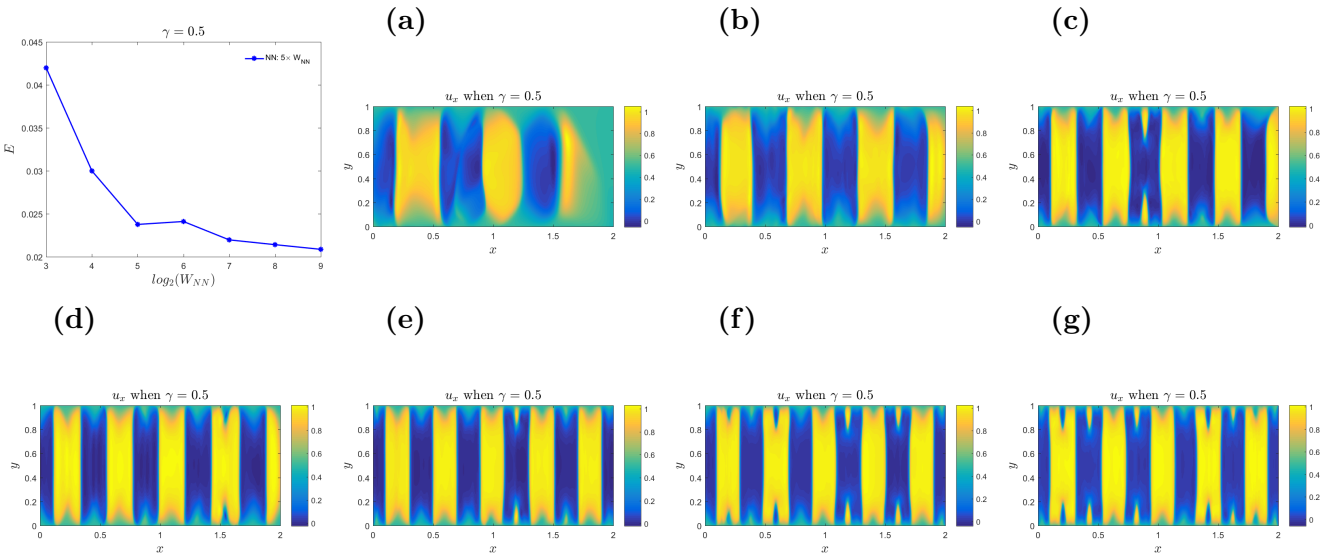

Figure 14: **2d regularized problem with SmReLU activation function.** Here $\gamma = 0.5$, *iteration* $= 300,000$, $\eta = 10^{-3}$ and $\varepsilon = \frac{0.1}{16}$. (a) NN: 5×8; (b) NN: 5×16; (c) NN: 5×32; (d) NN: 5×64; (e) NN: 5×128; (f) NN: 5×256; (f) NN: 5×512.

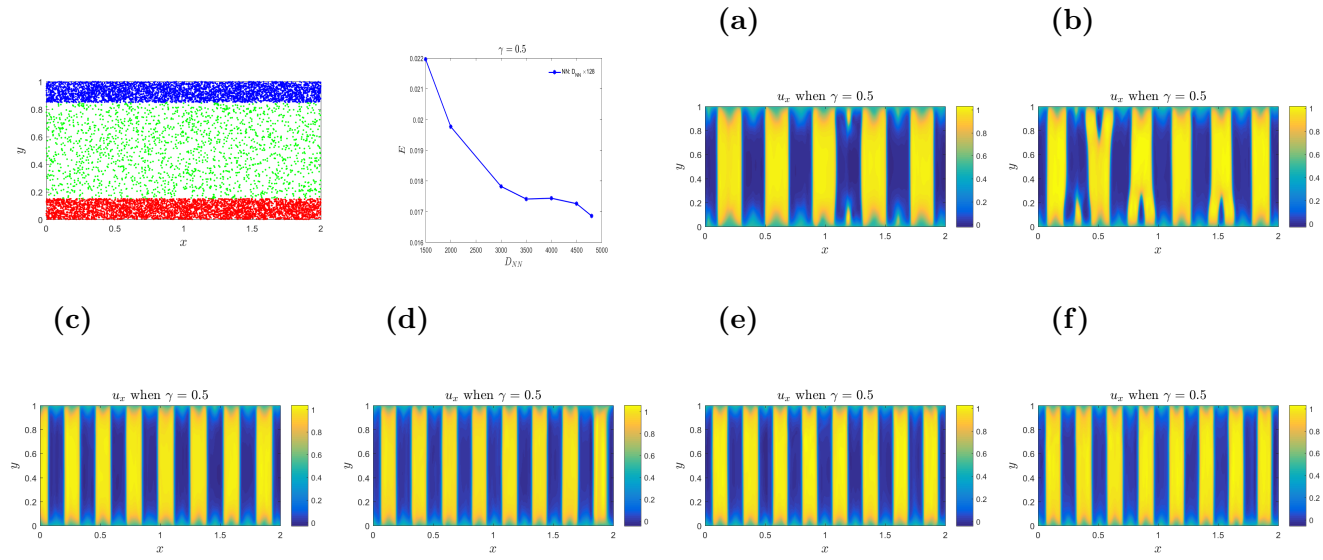

Figure 15: **Adapted collocation points** 2d regularized problem with SmReLU activation function, $\gamma = 0.5$, *iteration* $= 300,000$, $\eta = 10^{-3}$ and $\varepsilon = \frac{0.1}{16}$. Top: the position of collocation point; (a) $N_1 = N_3 = 1500$ and $N_2 = 7000$; (b) $N_1 = N_3 = 2000$ and $N_2 = 6000$; (c) $N_1 = N_3 = 3000$ and $N_2 = 4000$; (d) $N_1 = N_3 = 3500$ and $N_2 = 3000$; (e) $N_1 = N_3 = 4000$ and $N_2 = 2000$; (f) $N_1 = N_3 = 4500$ and $N_2 = 1000$.

3.3. Curved Inerfaces

We consider the following energy

$$W(\nabla u) = \frac{1}{2}[(\cos(\theta)u_x + \sin(\theta)u_y)^2(1 - (\cos(\theta)u_x + \sin(\theta)u_y))^2 + (-\sin(\theta)u_x + \cos(\theta)u_y)^2].$$

with Dirichlet boundary condition

$$u(x, y) = \gamma(\cos(\theta)x + \sin(\theta)y), \forall (x, y) \in \partial\Omega. \tag{34}$$

This corresponds to rotating one of the minima of W from $(0, 1)$ to $(\cos\theta, \sin\theta)$, where θ is a fixed parameter. Incompatibility with the boundary conditions now forces the twin boundaries (interfaces to bend and curve, demonstrating the mesh-free nature of the method. It is easy to see that previously we have considered the case with $\theta = 0$.

The results are shown in Fig. 16.

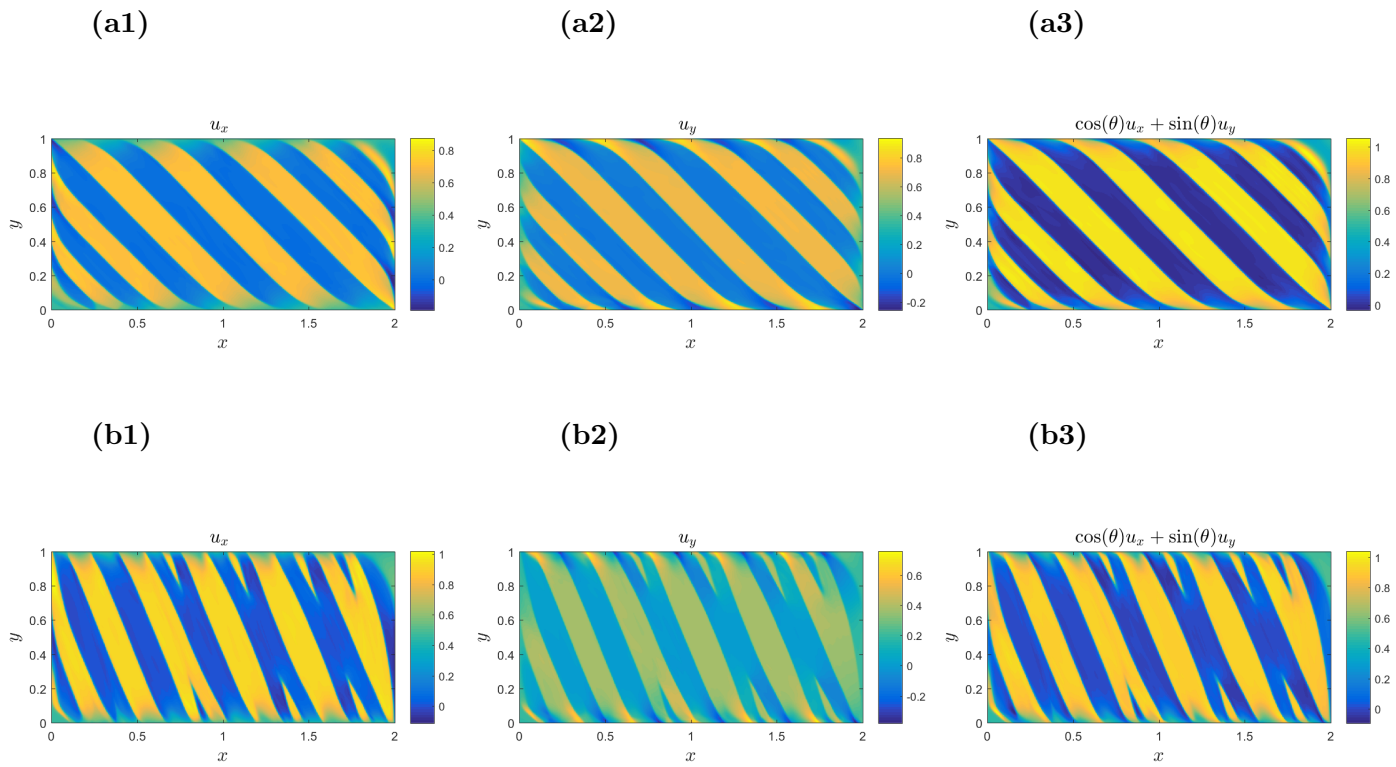

Figure 16: **Rotated minimum 2d problem with SmReLU activation function.** Here *iteration* $= 300,000$, $\eta = 10^{-3}$ and $\varepsilon = \frac{0.1}{32}$. (a) $\gamma = 0.5$ and $\theta = \frac{\pi}{4}$; (b) $\gamma = 0.5$ and $\theta = \frac{\pi}{8}$. (left) u_x; (middle) u_y; (right) $\cos(\theta)u_x + \sin(\theta)u_y$.

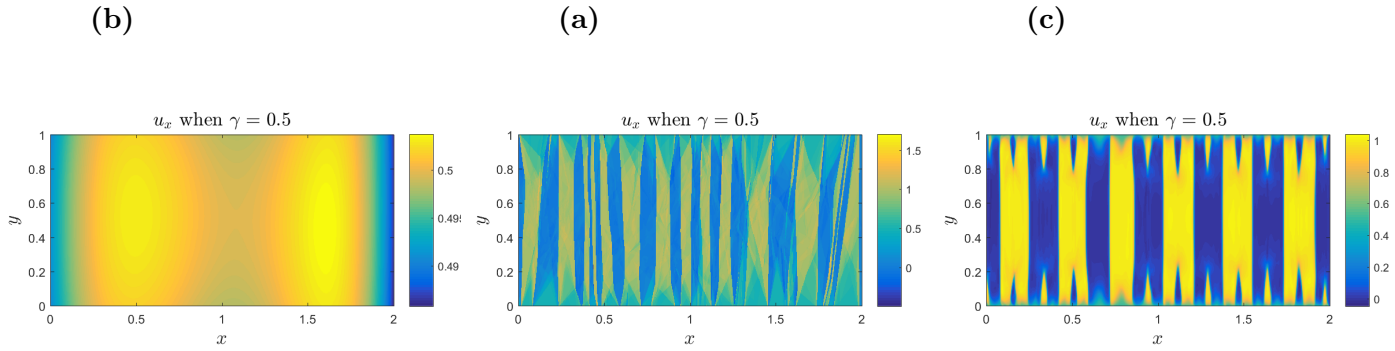

Figure 17: **Different activation functions in the 2d regularized problem.** Here $iteration = 300,000$, $\eta = 10^{-3}$ and $\varepsilon = \frac{0.1}{32}$. (a)Tanh activation function; (b) ReLU activation function; (c) SmReLU activation function.

4. Discussion

This work makes use of the Deep Ritz Method for the first time in order to solve a nonconvex gradient-energy minimization problem from materials science. Such problems are challenging in two ways. First, in general they do not possess a global minimum but only minimizing sequences, as well as a complex large set of local minima. Second, minimizers, including local ones, are characterized by complicated geometric microstructures, comprising multiple layers separated by twin interfaces, which are in effect free boundaries in the form of gradient discontinuities. These exhibit needle tapering and tip-splitting topological transitions near boundaries of the domain [20, 24, 11, 12, 38, 21].

A DNN represents the minimizer-candidate, which allows representation of the energy as a function of the weights and biases (the DNN parameters), and optimization with respect to them.

Our results confirm the ability of the Deep Ritz method to spontaneously capture the above microstructure complexities of local or global minimizers.

In contrast, the physics-informed neural network method does not work here because of the presence of nonsmooth weak solutions of the Euler-Lagrange PDEs whose high derivatives fail to exist at interfaces or even blow up.

A crucial new ingredient of our method is the activation function proposed here for the first time. To begin with, we observe that the ReLU activation function captures exact global minimizers in a simple unregularized 1D problem, due to its piecewise linear character. Other activation functions, such as Tanh and Sigmoid, struggle and are indeed unable to approximate these weak solutions with derivative jumps Fig. 3. For regularized problems, we introduce a new smoothened version of ReLU which we term SmReLU; see (27). It is quite successful in producing results that agree with careful FD computations [20] and captures the correct microstructure Fig. 1 (b), (c), while it outperforms other traditional activation functions, which fail at this task Fig. 17.

We find that it is essential for the DNN to have both sufficient width and depth to capture local minimizers. In general, increasing the number of layers (depth) increases the number of twin bands appearing in our solution until a point where further depth increases does not improve the solution. In that sense, depth plays a role analogous to mesh size in FE. We find that for fixed depth, it is not necessary to increase width past a certain point, so here depth plays a more important role.

Equipped with our new SmReLU activation function, our method has no problem capturing both sharp gradient discontinuities across twin boundaries and smoothened transition layers. It also approximates curved interfaces successfully Fig. 16.

This is because our approach is mesh-free, which is a great advantage over FEM and FD methods, especially in the presence of interfacial discontinuities carrying surface energy, where problems can arise due to mesh misorientation compared to interfaces [34].

Another advantage of this method is its simplicity, ease of programming, and the ability to naturally capture nonsmooth solutions with geometrically complex free interfaces in more

than one dimension without special treatment such as discontinuous Galerkin FE for higher gradients in nonconvex elasticity [18].

One usual complaint against the Deep Ritz Method is that the parametrization through DNNs as opposed to FE or FD would destroy any original convexity of the energy, a desirable property in optimization. Fortunately here the original energy is already nonconvex, so the DNN approach does not come at a disadvantage in this respect.

Acknowledgements

The research of Z. Zhang is supported by Hong Kong RGC grant (projects 17300318 and 17307921), National Natural Science Foundation of China (project 12171406), and Seed Funding Programme for Basic Research (HKU).

References

[1] R. Abeyaratne and J. Knowles, A continuum model of a thermoelastic solid capable of undergoing phase transitions, Journal of the Mechanics and Physics of Solids, 41 (1993), pp. 541–571.

[2] I. Babuška, The finite element method for elliptic equations with discontinuous coefficients, Computing, 5 (1970), pp. 207–213.

[3] J. Ball and R. James, Fine phase mixtures as minimizers of energy, in Analysis and continuum mechanics, Springer, 1989, pp. 647–686.

[4] L. Bottou, Large-scale machine learning with stochastic gradient descent, in Proceedings of COMPSTAT'2010, Springer, 2010, pp. 177–186.

[5] E. Bronstein, E. Faran, and D. Shilo, Analysis of austenite-martensite phase boundary and twinned microstructure in shape memory alloys: The role of twinning disconnections, Acta Materialia, 164 (2019), pp. 520–529.

[6] J. Carr, M. Gurtin, and M. Slemrod, Structured phase transitions on a finite interval, Archive for Rational Mechanics and Analysis, 86 (1984), pp. 317–351.

[7] X. Chen, J. Duan, and G. Karniadakis, Learning and meta-learning of stochastic advection–diffusion–reaction systems from sparse measurements, European Journal of Applied Mathematics, 32 (2021), pp. 397–420.

[8] Z. Chen and J. Zou, Finite element methods and their convergence for elliptic and parabolic interface problems, Numerische Mathematik, 79 (1998), pp. 175–202.

[9] C. Chu, I. Graham, and T. Y. Hou, A new multiscale finite element method for high-contrast elliptic interface problems, Math. Comp., 79 (2010), pp. 1915–1955.

[10] C. Chu and R. James, Biaxial loading experiments on Cu-Al-Ni single crystals, in Proceedings of the 1993 ASME Winter Annual Meeting, Publ by ASME, 1993, pp. 61–69.

[11] S. Conti, M. Lenz, N. Lüthen, M. Rumpf, and B. Zwicknagl, Geometry of martensite needles in shape memory alloys, Comptes Rendus. Mathématique, 358 (2020), pp. 1047–1057.

[12] P. Dondl, B. Heeren, and M. Rumpf, Optimization of the branching pattern in coherent phase transitions, Comptes Rendus Mathematique, 354 (2016), pp. 639–644.

[13] W. E, J. Han, and A. Jentzen, Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations, Communications in Mathematics and Statistics, 5 (2017), pp. 349–380.

[14] W. E and B. Yu, The deep Ritz method: A deep learning-based numerical algorithm for solving variational problems, Communications in Mathematics and Statistics, 6 (2018), pp. 1–12.

[15] J. Ericksen, Equilibrium of bars, Journal of elasticity, 5 (1975), pp. 191–201.

[16] E. Georgoulis, M. Loulakis, and A. Tsiourvas, Discrete gradient flow approximations of high dimensional evolution partial differential equations via deep neural networks, arXiv preprint arXiv:2206.00290, (2022).

[17] Y. Gong, B. Li, and Z. Li, Immersed-interface finite-element methods for elliptic interface problems with nonhomogeneous jump conditions, SIAM Journal on Numerical Analysis, 46 (2008), pp. 472–495.

[18] G. Grekas, K. Koumatos, C. Makridakis, and P. Rosakis, Approximations of energy minimization in cell-induced phase transitions of fibrous biomaterials: γ-convergence analysis, SIAM Journal on Numerical Analysis, 60 (2022), pp. 715–750.

[19] J. He, L. Li, J. Xu, and C. Zheng, Relu deep neural networks and linear finite elements, Journal of Computational Mathematics, 38 (2020), pp. 502–527.

[20] T. Healey and A. Sipos, Computational stability of phase-tip splitting in the presence of small interfacial energy in a simple two-phase solid, Physica D: Nonlinear phenomena, 261 (2013), pp. 62–69.

[21] T. Hou, P. Rosakis, and P. LeFloch, A level-set approach to the computation of twinning and phase-transition dynamics, Journal of Computational Physics, 150 (1999), pp. 302–331.

[22] Y. Khoo, J. Lu, and L. Ying, Solving parametric PDE problems with artificial neural networks, European Journal of Applied Mathematics, Special Issue 3: Connections between Deep Learning and Partial Differential Equations, 32 (2021), pp. 421–435.

[23] D. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv:1412.6980, (2014).

[24] R. Kohn and S. Müller, Branching of twins near an austenite¡ªtwinned-martensite interface, Philosophical Magazine A, 66 (1992), pp. 697–715.

[25] I. Lagaris, A. Likas, and D. Fotiadis, Artificial neural networks for solving ordinary and partial differential equations, IEEE Trans. Neural Netw., 9 (1998), pp. 987–1000.

[26] Y. LeCun, Y. Bengio, and G. Hinton, Deep learning, Nature, 521 (2015), p. 436.

[27] R. Leveque and Z. Li, The immersed interface method for elliptic equations with discontinuous coefficients and singular sources, SIAM Journal on Numerical Analysis, 31 (1994), pp. 1019–1044.

[28] B. Li and M. Luskin, Theory and computation for the microstructure near the interface between twinned layers and a pure variant of martensite, Materials Science and Engineering: A, 273 (1999), pp. 237–240.

[29] Z. Li, T. Lin, and X. Wu, New cartesian grid methods for interface problems using the finite element formulation, Numerische Mathematik, 96 (2003), pp. 61–98.

[30] Y. Liao and P. Ming, Deep nitsche method: Deep ritz method with essential boundary conditions, arXiv preprint arXiv:1912.01309, (2019).

[31] Q. Lou, X. Meng, and G. Karniadakis, Physics-informed neural networks for solving forward and inverse flow problems via the Boltzmann-BGK formulation, Journal of Computational Physics, 447 (2021), p. 110676.

[32] R. Mattey and S. Ghosh, A novel sequential method to train physics informed neural networks for Allen Cahn and Cahn Hilliard equations, Computer Methods in Applied Mechanics and Engineering, 390 (2022), p. 114474.

[33] H. Montanelli and Q. Du, New error bounds for deep ReLU networks using sparse grids, SIAM Journal on Mathematics of Data Science, 1 (2019), pp. 78–92.

[34] M. Negri, The anisotropy introduced by the mesh in the finite element approximation of the Mumford-Shah functional, Numerical Functional Analysis and Optimization, 20 (1999), pp. 957–982.

[35] C. Peskin, Numerical analysis of blood flow in the heart, Journal of Computational Physics, 25 (1977), pp. 220–252.

[36] M. Raissi, P. Perdikaris, and G. Karniadakis, Multistep Neural Networks for Data-driven Discovery of Nonlinear Dynamical Systems, arXiv:1801.01236, (2018).

[37] ——, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, Journal of Computational Physics, 378 (2019), pp. 686–707.

[38] H. Seiner, P. Plucinsky, V. Dabade, B. Benešová, and R. James, Branching of twins in shape memory alloys revisited, Journal of the Mechanics and Physics of Solids, 141 (2020), p. 103961.

[39] Y. Shin, J. Darbon, and G. Karniadakis, On the convergence of physics informed neural networks for linear second-order elliptic and parabolic type PDEs, arXiv preprint arXiv:2004.01806, (2020).

[40] S. Wang and P. Perdikaris, Deep learning of free boundary and Stefan problems, Journal of Computational Physics, 428 (2021), p. 109914.

[41] Z. Wang and Z. Zhang, A mesh-free method for interface problems using the deep learning approach, Journal of Computational Physics, 400 (2020), p. 108963.

[42] Z. Zhang, P. Rosakis, T. Hou, and G. Ravichandran, A minimal mechanosensing model predicts keratocyte evolution on flexible substrates, Journal of the Royal Society Interface, 17 (2020), p. 20200175.