ENM 531 Final Project Report: Materials Characterization from Non-equilibrium Process using Physics-informed Neural Networks

Shenglin Huang

May 15th, 2021

1 Introduction

Non-equilibrium process are ubiquitous in nature. In general, according to the underlying physics, we can separate a non-equilibrium process into two portions, the reversible portion and the irreversible portion. The reversible portion is the dynamics that has a time-symmetric structure when changing the sign of time. It can usually be described by the conservation law of some energy function, such as Helmholtz free energy for an isothermal system. On the contrary, the irreversible portion has no such time-symmetric structure and is related to the second law of thermodynamics. However, the second law of thermodynamics only provides an inequality to describe the direction of evolution (or time arrow). Although there has no general way to describe the evolution equation of the irreversible process, many of them, such as viscous force, phase transformation [1] and some simple plasticity phenomena [2], can be characterized by the dissipation potential in analogue to the free energy for reversible dynamics. Therefore, finding the free energy and dissipation potential is a crucial step for studying the non-equilibrium dynamics and materials characterization.

Besides traditional mechanics method, where people can use continuum based mechanics modeling [1] or information from particle level via fluctuation-dissipation relation [3] and fluctuation theorems [4] to extract material properties and evolution equations, machine learning becomes a powerful tool in material characterization. For example, people have successfully applied neural networks, including deep neural networks, integrable deep neural networks and recurrence neural networks, to obtain materials properties such as the configurational energy [5], free energy [6] and entropy production [7]. However, many of these methods consider little underlying physics and only use machine learning as a black-box regression model. In order to obtain the results with correct physics, one should encode more physics, such as physical symmetries, conservation laws or governing equations, into the machine learning method, which is now known as the physics-informed neural networks (PINNs) [8].

This project aim to use PINNs to achieve materials characterization from non-equilibrium process by extracting the free energy and dissipation potential. In Sec. 2, we will describe the system of interest, material characterization for a one-dimensional elastic bar, and the corresponding governing equations. Then Sec. 3 will introduce the numerical methods, including a PDE solver to simulate the pulling experiment for data generation and the PINNs architecture for materials characterization that extract free energy and dissipation potential. Two different models are tested for the proposed PINNs and the corresponding results are discussed in Sec. 4. Finally, conclusions and some outlooks are discussed in Sec. 5.

2 Mechanical Model for Materials Characterization

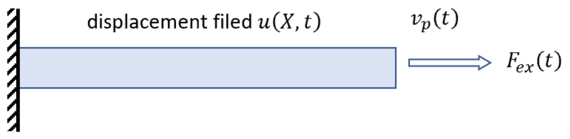


Figure 1: Schematics for the system.

Consider a non-equilibrium pulling process of a one-dimensional elastic bar as Fig. 1. The elastic proper of the bar is described by the free energy density f that is determined by the local strain ε, i.e., $f = f(\varepsilon)$. The Then the local stress can be simply given by the free energy derivative, i.e., $\sigma = f_{,\varepsilon}(\varepsilon) = \partial f(\varepsilon)/\partial\varepsilon$. The irreversible property of the system is described by the dissipation potential density ψ as a function of local velocity v, i.e., $\psi = \psi(v)$. The derivative of dissipation potential, i.e., $f_v(v) = \partial\psi(v)/\partial v$ describes the viscous force (per length) on the bar. The bar is fixed at the left end $X = 0$ and pulled with a velocity $v_p(t)$ at the right end $X = L$. During the evolution, the elastic force should balanced by the viscous force. Then the evolution of the system can be described by following equations,

$$\frac{\partial f_{,\varepsilon}(\varepsilon(X,t))}{\partial X} = \frac{\partial\psi(v)}{\partial v}, \tag{1}$$

$$\varepsilon(X,t) = \frac{\partial u(X,t)}{\partial X}, \tag{2}$$

$$v(X,t) = \frac{\partial u(X,t)}{\partial t}, \tag{3}$$

$$u(0,t) = 0, \tag{4}$$

$$v(L,t) = v_p(t), \tag{5}$$

$$u(X,0) = 0. \tag{6}$$

Equation 1 is the major governing equation for the system that describe the force balance. Equations 2 and 3 are two auxiliary equations to determine the local strain and velocity from the displacement field $u(X,t)$. Equations 4 and 5 are two boundary conditions and Eq. 6 is the initial condition. Therefore, given the free energy density $f(\varepsilon)$, the dissipation potential density $\psi(v)$ and the pulling velocity $v_p(t)$, we can solve the displacement field $u(X,t)$ from Eqs. 1-6. Furthermore, we can also find the external force for pulling the system as,

$$F_{ex}(t) = f_{,\varepsilon}(\varepsilon(L,t)). \tag{7}$$

Now we consider the inverse problem. If we know the displacement field $u(X,t)$ and the external force $F_{ex}(t)$ for a pulling process, we can obtain the strain and velocity fields, $\varepsilon(X,t)$ and $v(X,t)$ from Eqs. 2-3. Then, in principle, by using the governing PDE Eq. 1 together with Eq. 7 as the boundary condition, we should be able to extract the function forms of free energy density $f(\varepsilon)$ and dissipation potential density $\psi(v)$. In the following section, we will propose a numerical method to achieve this materials characterization process via physics-informed neural networks (PINNs).

3 Numerical Model Description

3.1 Data Generation

Before materials characterization process, we need to create a PDE solver to generate training data for the pulling process. Now we discretize the space into $N_X + 1$ points and the time into $N_t + 1$ points. We use superscript to infer the time index and subscript for the space index. For example, the displacement field $u(X,t)$ is now described by $u_i^n = u(X_i, t^n)$ with $i = 0, 1, \ldots, N_X$ and

$n = 0, 1, \ldots, N_t$. Equations 1-7 are discretized as following,

$$\frac{f_{,\varepsilon}(\varepsilon_{i+1}^n) - f_{,\varepsilon}(\varepsilon_i^n)}{\Delta X} = \psi_{,v}(v_i^n), \tag{8}$$

$$\varepsilon_i^n = \frac{u_i^n - u_{i-1}^n}{\Delta X}, \tag{9}$$

$$v_i^n = \frac{u_i^{n+1} - u_i^n}{\Delta t}, \tag{10}$$

$$u_0^n = 0, \tag{11}$$

$$v_{N_X}^n = v_p(t^n), \tag{12}$$

$$u_i^0 = 0, \tag{13}$$

$$F_{ex}^n = f_{,\varepsilon}(\varepsilon_{N_X}^i). \tag{14}$$

where $\Delta X = L/(N_X - 1)$, $\Delta t = T/(N_t - 1)$ and T is the total pulling time. Notice that a forward scheme is used for calculating the space gradient for stress, and a backward scheme is used for calculating the space gradient for strain. When solving the PDEs, in order to avoid numerical instability, the timestep Δt should be small enough. This indicates the output data is very fine in time. To avoid high computational cost for the later training process, I choose a coarser timestep Δt_{train} with corresponding $N_{t_{train}}$ for the training dataset, which includes the displacement field u_i^n and external force F_{ex}^n.

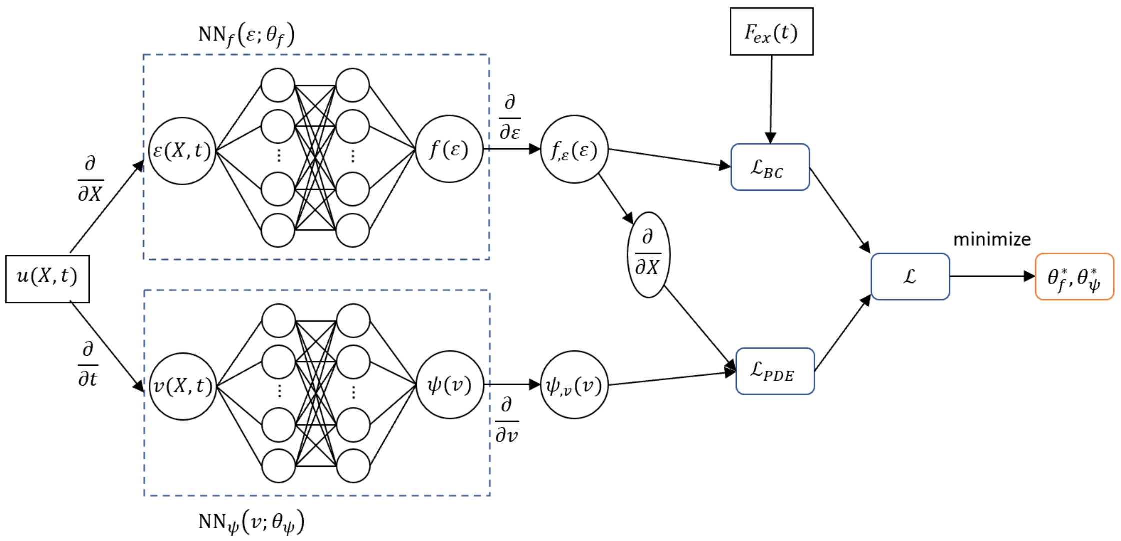
3.2 PINNs

With the generated training dataset, the following physics-informed neural networks, shown as Fig. 2, are applied for obtaining the free energy density $f(\varepsilon)$ and dissipation potential density $\psi(v)$. The whole PINN architecture consists of two neural networks (NNs) that fit the free energy density $f(\varepsilon)$ and dissipation potential density $\psi(v)$, respectively. With the fitted results, the loss function is constructed as the summation of the contribution from the PDE ($\mathcal{L}_{PDE}$) and the contribution from the boundary condition ($\mathcal{L}_{BC}$),

$$\mathcal{L} = \mathcal{L}_{PDE} + \mathcal{L}_{BC}, \tag{15}$$

$$\mathcal{L}_{PDE} = \alpha_{PDE} \left\| \frac{\partial f_{,\varepsilon}(\varepsilon(X,t))}{\partial X} - \frac{\partial \psi(v)}{\partial v} \right\|^2, \tag{16}$$

$$\mathcal{L}_{BC} = \alpha_{BC} \left\| f_{,\varepsilon}(\varepsilon(L,t)) - F_{ex}(t) \right\|^2. \tag{17}$$



$\mathrm{NN}_f(\varepsilon;\theta_f)$
$\varepsilon(X,t)$
$f(\varepsilon)$
$\frac{\partial}{\partial \varepsilon}$
$f_{,\varepsilon}(\varepsilon)$
$F_{ex}(t)$
$\mathcal{L}_{BC}$
$\frac{\partial}{\partial X}$
$u(X,t)$
$\frac{\partial}{\partial X}$
$\mathcal{L}$
minimize
$\theta_f^*, \theta_\psi^*$
$\frac{\partial}{\partial t}$
$v(X,t)$
$\psi(v)$
$\psi_{,v}(v)$
$\frac{\partial}{\partial v}$
$\mathcal{L}_{PDE}$
$\mathrm{NN}_\psi(v;\theta_\psi)$

Figure 2: PINNs architecture.

Here $\|\cdot\|^2$ refers to the L_2 norm (the average, not the summation) over space and time for the PDE term $\mathcal{L}_{PDE}$ and over time for the boundary term $\mathcal{L}_{BC}$. α_{PDE} and α_{BC} are two factors that normalize both $\mathcal{L}_{PDE}$ and $\mathcal{L}_{BC}$ to scale one.

The training process of the PINNs can be described as following. First, with given input data $u(X,t)$, the strain $\varepsilon(X,t)$ and velocity $v(X,t)$ are calculated according to Eqs. 9-10. Then, two neural networks NN_f and NN_ψ will fit the free energy and dissipation potential, respectively. Notice that a normalization process is applied such that each neural network only map a scale one parameter to another scale one parameter, i.e., $\tilde{f}(\tilde{\varepsilon})$ and $\tilde{\psi}(\tilde{v})$. The strain and velocity are normalized by there standard deviation, $\tilde{\varepsilon} = \varepsilon/\sigma_\varepsilon$, $\tilde{v} = v/\sigma_v$. According to the dimensional analysis, the rescaled output (free energy and dissipation potential) are calculated by,

$$f(\varepsilon) = \sigma_{F_{ex}}\sigma_\varepsilon \left[\tilde{f}(\tilde{\varepsilon}) - \tilde{f}(0)\right], \tag{18}$$

$$\psi(v) = \frac{\sigma_{F_{ex}}\sigma_v}{L}\left[\tilde{\psi}(\tilde{v}) - \tilde{\psi}(0)\right], \tag{19}$$

where the subtraction of $\tilde{f}(0)$ and $\tilde{\psi}(0)$ can force the free energy and dissipation potential across zero at zero input, i.e., $f(0) = 0$ and $\psi(0) = 0$. After two neural networks, the loss function is calculated by another input dataset $F_{ex}(t)$ and the derivatives (and spacial gradient) of the fitted functions. Finally, using stochastic gradient descent, we can find the optimized for the training parameters of two neural networks.

There are following notices need to be mentioned for the PINNs: (1) The derivatives with respect to strain ε and velocity v can be done by automatic differentiation. However, since the fitted functions $f(\varepsilon)$ and $\psi(v)$ do not explicitly depend on X and t, the derivatives with respective to X and t before and after the neural networks cannot be calculated by automatic differentiation. (2) Since a spatial gradient is required after the neural networks, the spacial and temporal order should be reassigned to the fitted function after the neural networks. (3) In the loss function, only the derivatives (not the absolute value) of the fitted function are required. According to the studies for Integrable deep neural networks (IDNN) [6], the derivative of the chosen activation function should also be an activation function. Therefore, the softplus function $g(x) = \ln(1 + e^x)$, whose derivative is the sigmoid function $g'(x) = 1/(1 + e^{-x})$, is selected.

4 Training Models and Results

4.1 Model One

To verify the proposed numerical method, we first present a simple model with both the free energy density and dissipation potential density are quadratic functions,

$$f(\varepsilon) = \frac{1}{2}EA\varepsilon^2, \tag{20}$$

$$\psi(v) = \frac{1}{2}\nu v^2, \tag{21}$$

such that the corresponding elastic stress and viscous force are linear,

$$f_{,\varepsilon}(\varepsilon) = EA\varepsilon, \tag{22}$$

$$\psi_{,v}(v) = \nu v, \tag{23}$$

where EA is the product of Young's modulus and ν is the viscosity. In this model, we choose $EA = 1$, $\nu = 5$. The total length of the bar is $L = 5$. It is pulled under a sinusoidal velocity,

$$v_p(t) = v_0 \left[1 - \cos\left(\frac{2\pi t}{T_{period}}\right)\right], \tag{24}$$

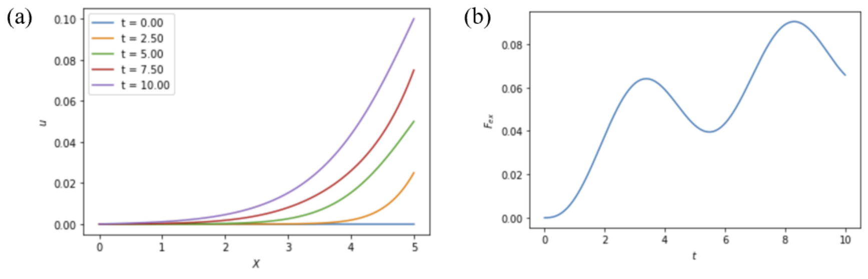


Figure 3: Training data for model 1: (a) displacement field, (b) external force.

with $v_0 = 0.02$ and $T_{period} = 5$ for total pulling time $T = 10$.

For solving the PDEs (Eqs. 8-13), we choose $\Delta X = 0.05$ (or $N_X = 100$) and $\Delta t = 0.001$. The coarser timestep for the training data is chosen as $\Delta t_{train} = 0.05$, which indicate $N_{t_{train}} = 200$. The resulted displacement profile and external force are shown in Fig. 3, which are used as the training dataset.

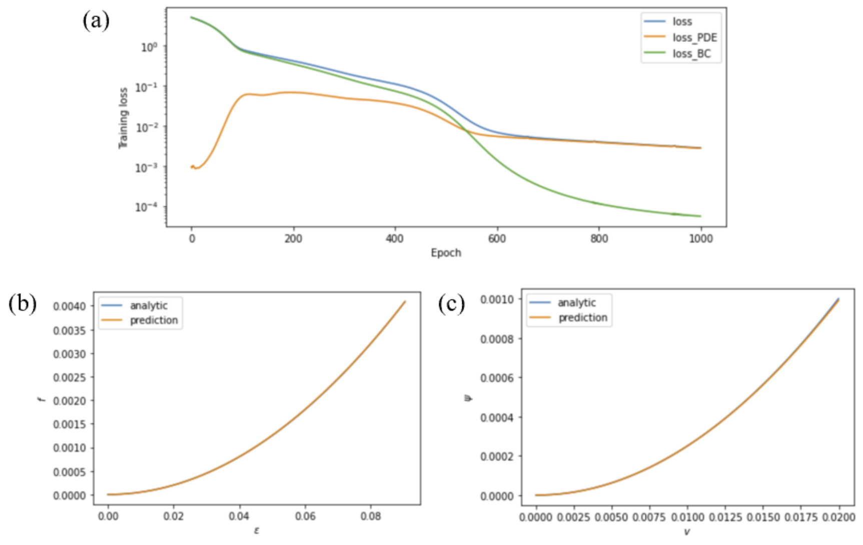


Figure 4: The (a) loss function, (b) free energy density and (c) dissipation potential density for the training result of the PINNs for model 1 with layer dimension as $[1, 25, 25, 1]$, learning rate as 10^{-3} and softplus activation function.

In the PINNs, both neural networks are set as two layers with 25 nodes in each layer. The Adam upgrade method is used for the stochastic gradient descent with a learning rate as 10^{-3}. Figure 4 shows the corresponding training results. After 1000 epochs, the total loss value reduced to 2.80^{-3}, in which the portion from the PDE and that from the boundary condition are 2.74×10^{-3} and 5.57^{-5}.

Model 1	$\mathcal{L}$	$\mathcal{L}_{PDE}$	$\mathcal{L}_{BC}$	err_f	err_ψ
reference	2.80×10^{-3}	2.74×10^{-3}	5.57×10^{-5}	1.48×10^{-3}	5.54×10^{-3}
10^{-2} learning rate	1.75×10^{-3}	1.70×10^{-3}	4.70×10^{-5}	1.73×10^{-3}	3.82×10^{-3}
10 nodes per layer	1.96×10^{-3}	1.93×10^{-3}	6.06×10^{-5}	1.38×10^{-3}	3.65×10^{-3}
tanh activation function	3.02×10^{-2}	7.16×10^{-3}	2.31×10^{-2}	8.90×10^{-2}	1.68×10^{-1}
without normalization	8.73×10^{-1}	7.29×10^{-2}	8.00×10^{-1}	5.24×10^{-1}	9.56×10^{-1}

Table 1: Results of Model 1.

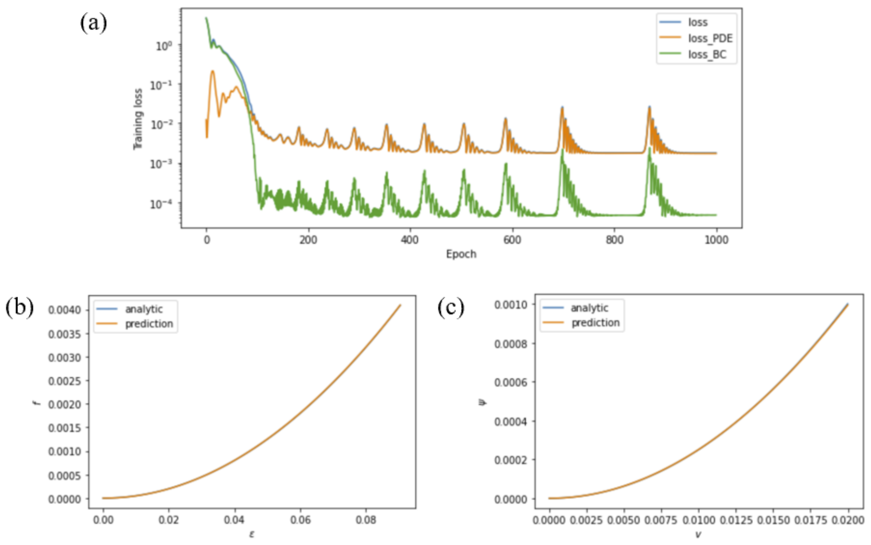


Figure 5: The (a) loss function, (b) free energy density and (c) dissipation potential density for the training result of the PINNs for model 1 with layer dimension as $[1, 25, 25, 1]$, learning rate as 10^{-2} and softplus activation function.

Although we can find that the loss function can decrease to a lower value, the predicted results are in good agreement with the analytic value. The L_2 relative errors for the free energy density and dissipation potential, denoted as err_f and err_ψ in Table 1, are 1.48×10^{-3} and 5.54×10^{-3}.

In comparison to the reference model setting showed above, Figs. 5-8 demonstrate the training results after 1000 epochs for several different parameter settings: increase learning rate from 10^{-3} to 10^{-2} (Fig. 5), decrease the node number per layer from 25 to 10 for both neural networks (Fig. 6), changing the activation function from softplus to tanh (Fig.7) and applying two neural networks without normalization process for the inputs and outputs (Fig. 8). The corresponding data are listed in Table 1. Notice that each result only changed one model parameter compared to the reference model. From the results we can find that increase the learning rate or decrease the node number per layer can result a faster training process with similar loss value and training error comparing with the reference setting. However, when the optimization process is approaching the desired result, huge

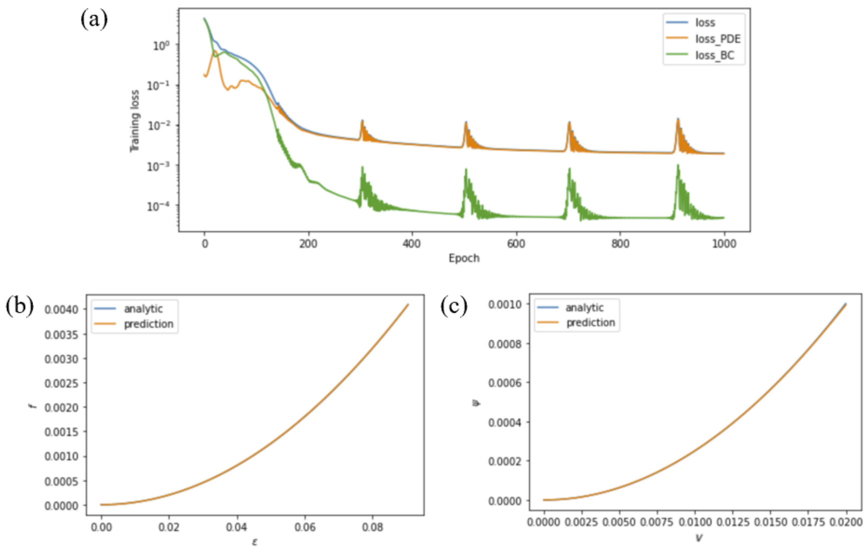


Figure 6: The (a) loss function, (b) free energy density and (c) dissipation potential density for the training result of the PINNs for model 1 with layer dimension as $[1, 10, 10, 1]$, learning rate as 10^{-3} and softplus activation function.

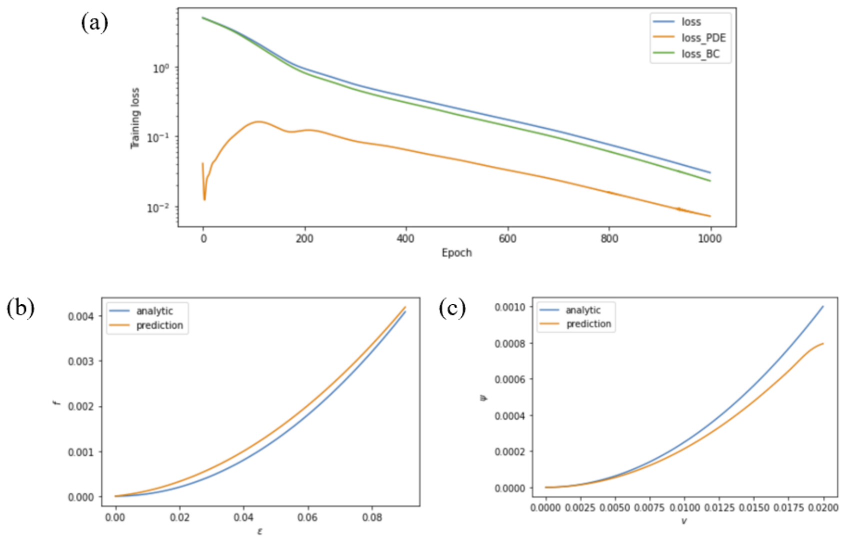


Figure 7: The (a) loss function, (b) free energy density and (c) dissipation potential density for the training result of the PINNs for model 1 with layer dimension as $[1, 25, 25, 1]$, learning rate as 10^{-3} and tanh activation function.

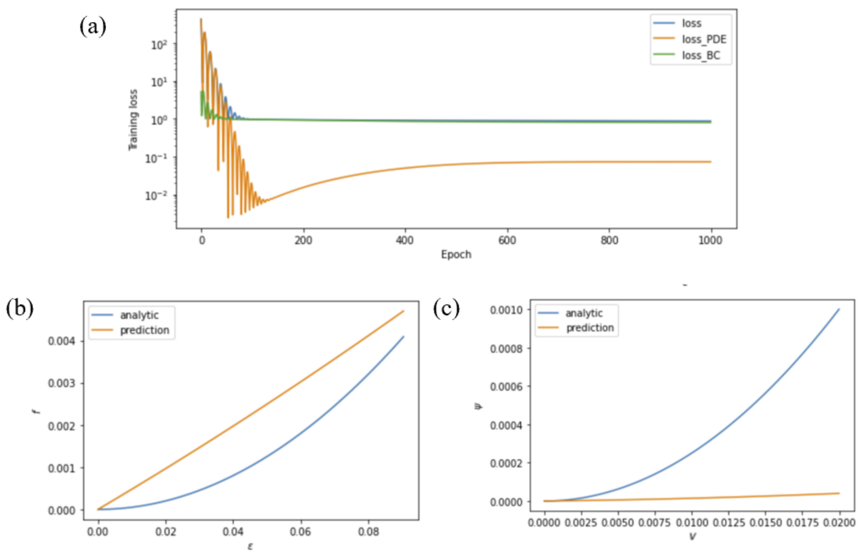


Figure 8: The (a) loss function, (b) free energy density and (c) dissipation potential density for the training result of the PINNs for model 1 with layer dimension as $[1, 25, 25, 1]$, learning rate as 10^{-3}, softplus activation function but without the normalization process.

oscillation appears, which indicates the training parameters goes beyond the optimum value during the stochastic gradient descent. Therefore one should make sure the loss function is true converged to a relatively steady stage, not the oscillation region during the training. From Fig. 7 we can find that using hyperbolic tangent as the activation function can in principle fit the free energy and dissipation potential well, but requires a longer training time compared to using softmax function. Finally, Fig. 8 shows that without the normalization, the optimization is trapped by a local minimum. At this local minimum, the fitted free energy is linear such that the stress is a constant around the average pulling external force. This leads to a zero stress gradient and a zero dissipation potential. That's why $\mathcal{L}_{PDE}$ is much smaller than $\mathcal{L}_{BC}$. Therefore, the normalization process for the neural networks is a crucial for the training.

4.2 Model Two

The above section shows that the presented method can successfully extract the free energy density and dissipation potential when both of them are quadratic functions. In order to test the model in a more complex system, the second model is applied.

The second model is described by a quartic free energy density and a power-law dissipation poten-

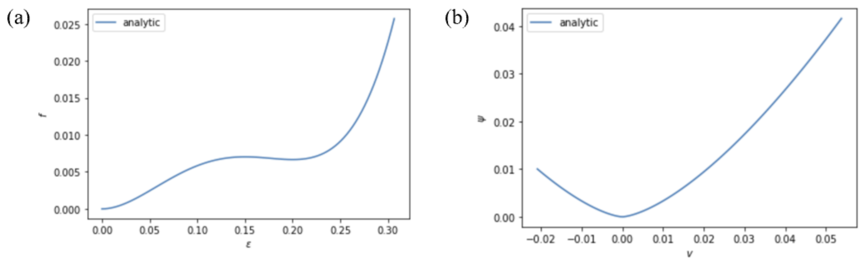


Figure 9: Analytic (a) free energy density and (b) dissipation potential density for model 2.

tial,

$$f(\varepsilon)=\frac{1}{12}k_4\varepsilon^2\left[3\varepsilon^2-(\varepsilon_1+\varepsilon_{cr}\varepsilon+6\varepsilon_1\varepsilon_{cr})\right], \tag{25}$$

$$\psi(v)=\nu v^{n+1}, \tag{26}$$

such that the elastic stress and viscous force are given as,

$$f_{,\varepsilon}(\varepsilon)=k_4\varepsilon\left(\varepsilon-\varepsilon_1\right)\left(\varepsilon-\varepsilon_{cr}\right), \tag{27}$$

$$\psi_{,v}(v)=\nu v^n, \tag{28}$$

where $k_4=100$ is an effective spring constant for the cubic elastic stress, $\varepsilon_1=0.2$ is the strain of the second minimum in free energy, ε_{cr} is the strain of the local maximum in free energy, $\nu=5$ is the drag coefficient and $n=0.5$ is the viscous power. The profiles for the free energy density and dissipation potential density are shown in Fig. 9, where the free energy is obviously non-convex and the minimum of the dissipation potential is sharper than the previous quadratic one. The total length of the bar is $L=1$, shorter than model 1. The pulling velocity is set as a constant $v_p=0.02$. The whole pulling process is $T=10$.

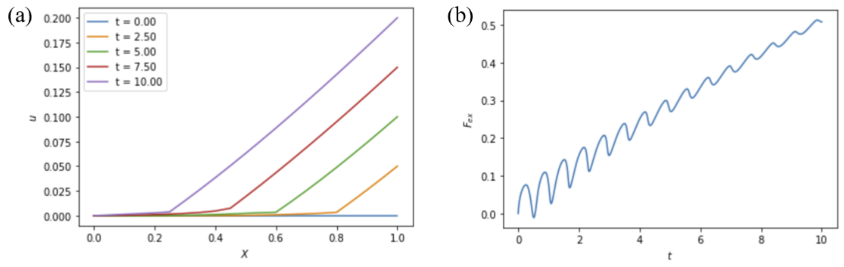


Figure 10: Training data for model 2: (a) displacement field, (b) external force.

For data generation, we choose $\Delta X = 0.05$ (or $N_X = 20$) and $\Delta t = 0.001$. The coarser timestep for the training dataset is chosen as $\Delta t_{train} = 0.05$ with $N_{t_{train}} = 200$, which is the same as model 1. Figure 10 shows the displacement profile and external force for model 2. Compare to model 1, the current model involves phase transformation process. We can observe a sharp front propagating from right to left in the displacement profile and the external force is oscillating while the propagation.

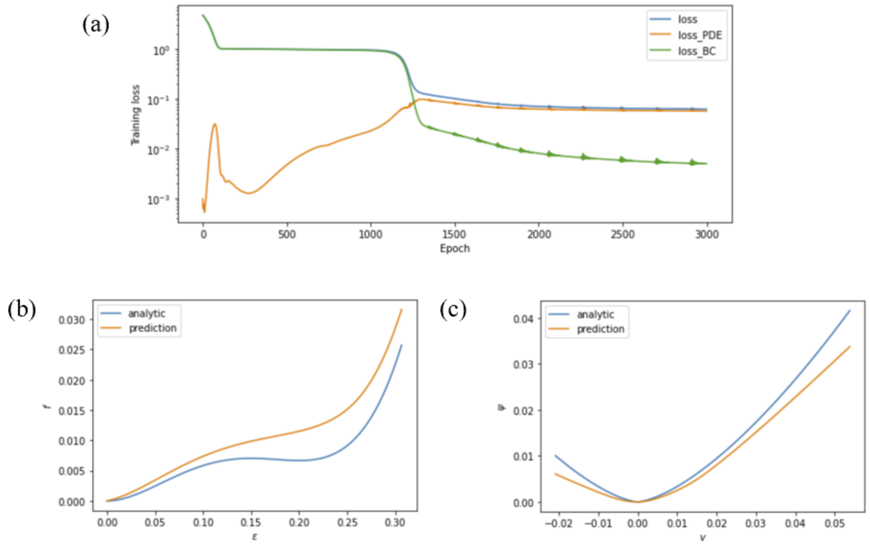


Figure 11: The (a) loss function, (b) free energy density and (c) dissipation potential density for the training result of the PINNs for model 2 with layer dimension as $[1, 25, 25, 1]$, learning rate as 10^{-3} and softplus activation function.

The training process is tested by two different parameter settings. In the first case, shown as Fig. 11, each neural network has two layers with 25 nodes in each layer. The learning rate is chosen as 10^{-3}. In the second case, shown as Fig. 12 each neural network has three layers with 50 nodes in each layer. The learning rate is chosen as 4×10^{-3}. The corresponding results for loss functions and L_2 relative errors are shown in Table 2. From the results, we can find that the optimization process was trapped in a local minimum at the beginning and then reached the second local minimum with lower loss value. However, the predictions in both cases deviate from the the analytic results systematically. The decrease trend of the loss functions in both cases is not stopping, while the optimization process for the one with less nodes looks much smoother than the other one with more nodes. It is unclear that whether the optimization can reach another local minimum with lower loss value. This unpleasant result can be partially explain by the non-convexity in the testing free energy. Once the evolving local strain reach the local maximum in free energy profile, it will jump to the second minimum in a very

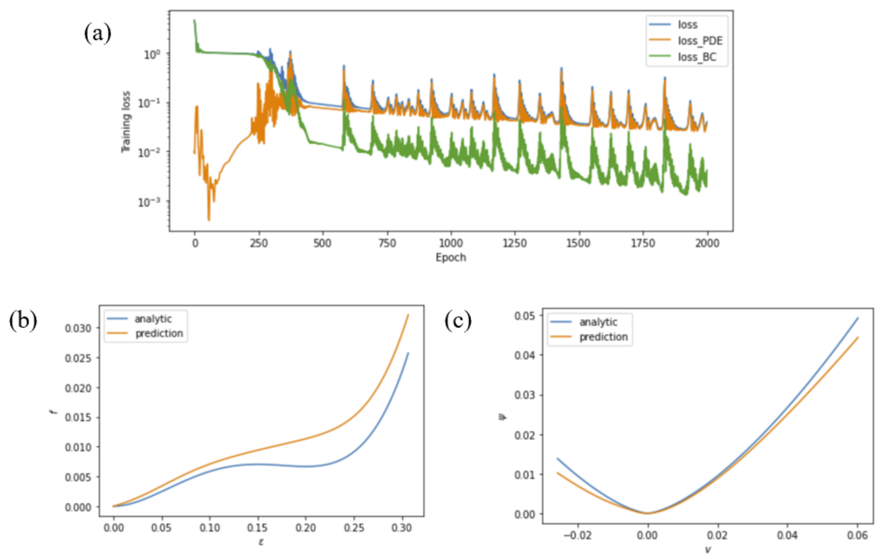


Figure 12: The (a) loss function, (b) free energy density and (c) dissipation potential density for the training result of the PINNs for model 2 with layer dimension as $[1, 50, 50, 50, 1]$, learning rate as 4×10^{-3} and softplus activation function.

Model 1	$\mathcal{L}$	$\mathcal{L}_{PDE}$	$\mathcal{L}_{BC}$	err_f	err_ψ
less nodes	6.12×10^{-2}	5.63×10^{-2}	4.92×10^{-3}	4.39×10^{-1}	1.70×10^{-1}
more nodes	3.90×10^{-2}	3.52×10^{-2}	4.11×10^{-3}	2.53×10^{-1}	9.41×10^{-2}

Table 2: Results of Model 2.

short time. Then a phase front (sharp corner) occur in the displacement profile and the external force decrease dramatically. This will result in less training data around the non-convex region in free energy profile. Therefore the prediction is much worse than the model 1 with convex free energy.

5 Conclusions and Outlooks

This project is aimed to achieve material characterization from non-equilibrium process via physics-informed neural networks (PINNs). First, I generated a PDE solver that simulate the pulling experiment for an 1D elastic-viscous bar with given free energy, dissipation potential and pulling velocity. Next, I proposed a PINNs model for material characterization (free energy and dissipation potential) with the generated dataset (displacement profile and external force) from the pulling experiment. Finally, two physics models are applied to test the PINNs. The PINNs successfully extracted the free energy and dissipation potential from the first training model with both free energy dissipation potential are quadratic. For the second model with a non-convex free energy, the PINNs obtained the rough profiles for the free energy and dissipation potential, while the errors for the exact values are still significant.

To fix the training error in the second model, following strategies can be applied in the future works. First, since there are less data points in the non-convex region, we can try to use more data points with finer timestep for the training dataset. Second, since the current model only trains materials properties from one single pulling experiment, we can use different pulling experiments as multiple sets of training data to extract the same materials properties. Particularly, it is worth to train a compress process to explore the region that is hard to achieve in the current pulling process. Third, since the majority information of free energy comes from the boundary condition and all the information of dissipation potential is fitted from the PDE, we can try to apply different pre-factors for two portions of loss function. This may enhance training performance.

References

[1] Alejandro Torres-Sánchez, Juan M Vanegas, Prashant K Purohit, and Marino Arroyo. Combined molecular/continuum modeling reveals the role of friction during fast unfolding of coiled-coil proteins. *Soft matter*, 15(24):4961–4975, 2019.

[2] Michael Ortiz and Laurent Stainier. The variational formulation of viscoplastic constitutive updates. *Computer methods in applied mechanics and engineering*, 171(3-4):419–444, 1999.

[3] Xiaoguai Li, Nicolas Dirr, Peter Embacher, Johannes Zimmer, and Celia Reina. Harnessing fluctuations to discover dissipative evolution equations. *Journal of the Mechanics and Physics of Solids*, 131:240–251, 2019.

[4] Shenglin Huang, Chuanpeng Sun, Prashant K Purohit, and Celia Reina. Harnessing fluctuation theorems to discover free energy and dissipation potentials from non-equilibrium data. *Journal of the Mechanics and Physics of Solids*, 149:104323, 2021.

[5] Anirudh Raju Natarajan and Anton Van der Ven. Machine-learning the configurational energy of multicomponent crystalline solids. *npj Computational Materials*, 4(1):1–7, 2018.

[6] Gregory H Teichert, AR Natarajan, A Van der Ven, and Krishna Garikipati. Machine learning materials physics: Integrable deep neural networks enable scale bridging by learning free energy functions. *Computer Methods in Applied Mechanics and Engineering*, 353:201–216, 2019.

[7] Dong-Kyum Kim, Youngkyoung Bae, Sangyun Lee, and Hawoong Jeong. Learning entropy production via neural networks. *Physical Review Letters*, 125(14):140604, 2020.

[8] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.