A PROOF OF THEOREM 1

In this section we provide the detailed proof for Theorem 1, based on Assumptions 1 and 2. Intuitively, that means the underlying implicit problem is solved with a converging fixed point method. This condition is a basic requirement by numerical PDEs, and it generally holds true in many applications governed by nonlinear and complex PDEs, such as in our three experiments.

Here, we prove that the MetaP is universal, i.e., give a fixed point method satisfying Assumptions 1-2, one can find parameter sets θ^η whose output approximates $\mathbf{U}^{\eta,*}$ to a desired accuracy, $\varepsilon > 0$, for all $\eta = 1, \cdots, H$ tasks. For the task-wise parameters, with a slight abuse of notation, we denote $P^\eta \in \mathbb{R}^{d_h M \times (d_g+s)M}$ as the collection of the pointwise weight matrices at each discretization point in χ for the η-th task, and $\mathbf{p}^\eta \in \mathbb{R}^{d_h M}$ for the bias in the lifting layer. Then, for the parameters shared among all tasks, in the iterative layer we denote $\mathbf{C} = [\mathbf{c}(\mathbf{x}_1), \cdots, \mathbf{c}(\mathbf{x}_M)] \in \mathbb{R}^{d_h M}$ as the collection of pointwise bias vectors $\mathbf{c}(\mathbf{x}_i)$, $W \in \mathbb{R}^{d_h \times d_h}$ for the local linear transformation, and $R = \mathcal{F}[\kappa(\cdot; \mathbf{v})] \in \mathbb{C}^{d_h \times d_h \times M} \in \mathbb{C}^{d_h \times d_h \times M}$ for the Fourier coefficients of the kernel κ. For simplicity, here we have assumed that the Fourier coefficient is not truncated, and all available frequencies are used. Then, for the projection layer we seek $Q_1 \in \mathbb{R}^{d_Q M \times d_h M}$, $Q_2 \in \mathbb{R}^{d_u M \times d_Q M}$, $\mathbf{q}_1 \in \mathbb{R}^{d_Q M}$ and $\mathbf{q}_2 \in \mathbb{R}^{d_u M}$. For the simplicity of notation, in this section we organize the feature vector $\mathbf{H} \in \mathbb{R}^{d_h M}$ in a way such that the components corresponding to each discretization point are adjacent, i.e., $\mathbf{H} = [\mathbf{H}(\mathbf{x}_1), \cdots, \mathbf{H}(\mathbf{x}_M)]$ and $\mathbf{H}(\mathbf{x}_i) \in \mathbb{R}^{d_h}$.

We point out that under this circumstance, we have the (discretized) iterative layer can be written as

$$\begin{aligned}\mathcal{J}[\mathbf{H}(l\Delta t)] =& \mathbf{H}(l\Delta t) + \Delta t \sigma \left(\tilde{W} \mathbf{H}(l\Delta t) + \mathrm{Re}(\mathcal{F}_{\Delta x}^{-1}(R \cdot \mathcal{F}_{\Delta x}(\mathbf{H}(l\Delta t)))) + \mathbf{C} \right) \\ =& \mathbf{H}(l\Delta t) + \Delta t \sigma \left(V \mathbf{H}(l\Delta t) + \mathbf{C} \right),\end{aligned}$$

with

$$V := \mathrm{Re} \begin{bmatrix} \sum_{n=0}^{M-1} R_{n+1} + W & \sum_{n=0}^{M-1} R_{n+1} \exp(\frac{2i\pi\Delta xn}{M}) & \dots & \sum_{n=0}^{M-1} R_{n+1} \exp(\frac{2i\pi(M-1)\Delta xn}{M}) \\ \sum_{n=0}^{M-1} R_{n+1} \exp(\frac{2i\pi\Delta xn}{M}) & \sum_{n=0}^{M-1} R_{n+1} + W & \dots & \sum_{n=0}^{M-1} R_{n+1} \exp(\frac{2i\pi(M-2)\Delta xn}{M}) \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{n=0}^{M-1} R_{n+1} \exp(\frac{2i\pi(M-1)\Delta xn}{M}) & \sum_{n=0}^{M-1} R_{n+1} \exp(\frac{2i\pi(M-2)\Delta xn}{M}) & \dots & \sum_{n=0}^{M-1} R_{n+1} + W \end{bmatrix}.$$

Here, $R \in \mathbb{C}^{M \times d_h \times d_h}$ with $R_i \in \mathbb{C}^{d_h \times d_h}$ being the component associated with each discretization point $\mathbf{x}_i \in \chi$, $V \in \mathbb{R}^{d_h M \times d_h M}$, $\mathbf{C} \in \mathbb{R}^{d_h M}$, $\tilde{W} := W \oplus W \oplus \cdots \oplus W$ is a $d_h M \times d_h M$ block diagonal matrix formed by $W \in \mathbb{R}^{d_h \times d_h}$, $\mathcal{F}_{\Delta x}$ and $\mathcal{F}_{\Delta x}^{-1}$ denote the discrete Fourier transform and its inverse, respectively. By further taking $R_2 = \cdots = R_M = W = 0$, a $d_h \times d_h$ matrix with all its elements being zero, it suffices to show the universal approximation property for an iterative layer as follows:

$$\mathcal{J}(\mathbf{H}(l\Delta t)) := \mathbf{H}(l\Delta t) + \Delta t \sigma \left(\tilde{V} \mathbf{H}(l\Delta t) + \mathbf{C} \right)$$

where $\tilde{V} := \mathbf{1}_{[M,M]} \otimes V$ with $V \in \mathbb{R}^{d_h \times d_h}$ and $\mathbf{1}_{[m,n]}$ being an m by n all-ones matrix.

To be more precise, we will prove the following theorem:

Theorem 1 (Universal approximation)**.** *Let $\mathbf{U}^{\eta,*} = [\mathbf{u}^\eta(\mathbf{x}_1), \mathbf{u}^\eta(\mathbf{x}_2), \dots, \mathbf{u}^\eta(\mathbf{x}_M)]$ be the ground-truth solution of η-th task that satisfies Assumptions 1-2, the activation function σ for all iterative kernel integration layers be the ReLU function, and the activation function in the projection layer be the identity function. Then for any $\varepsilon > 0$, there exist sufficiently large layer number $L > 0$ and feature dimension number $d_h > 0$, such that one can find a parameter set for the multi-task problem, $\theta^\eta = [\theta_P^\eta, \theta_I, \theta_Q]$ with the corresponding MetaP model satisfies*

$$\left\| \mathcal{Q}_{\theta_Q} \circ (\mathcal{J}_{\theta_I})^L \circ \mathcal{P}_{\theta_P^\eta}([\mathbf{U}^0, \mathbf{G}^\eta]^{\mathrm{T}}) - \mathbf{U}^{\eta,*} \right\| \leq \varepsilon, \quad \forall \mathbf{G}^\eta \in \mathbb{R}^M.$$

For the proof of this main theorem, we need the following approximation property of a shallow neural network, with its detailed proof provided in You et al. (2022c):

Lemma 1. *Given a continuous function $\mathcal{T} : \mathbb{R}^{2M} \mapsto \mathbb{R}^M$, and a non-polynomial and continuous activation function σ, for any constant $\hat{\varepsilon} > 0$ there exists a shallow neural network model $\hat{\mathcal{T}} := S\sigma\left(B\mathbf{X} + A\right)$ such that*

$$||\mathcal{T}(\mathbf{X}) - \hat{\mathcal{T}}(\mathbf{X})||_{l^2(\mathbb{R}^M)} \leq \hat{\varepsilon}, \quad \forall \mathbf{X} \in \mathbb{R}^{2M},$$

for sufficiently large feature dimension $\hat{d} > 0$. Here, $S \in \mathbb{R}^{M\times \hat{d}M}$, $B \in \mathbb{R}^{\hat{d}M\times 2M}$, and $A \in \mathbb{R}^{\hat{d}M}$ are matrices/vectors which are independent of $\mathbf{X}$.

We now proceed to the proof of Theorem 1:

Proof. Since all $\mathbf{U}^{\eta,*}$ satisfies Assumptions 1-2, for any $\varepsilon > 0$, we first pick a sufficiently large integer L such that the L-th layer iteration result of this fixed point formulation satisfies $||\mathbf{U}^L - \mathbf{U}^{\eta,*}||_{l^2(\mathbb{R}^M)} \leq \frac{\varepsilon}{2}$ for all tasks. By taking $\hat{\varepsilon} := \frac{m\varepsilon}{2(1+m)^L}$ in Lemma 1, there exists a sufficiently large feature dimension $\hat{d}$ and one can find $S \in \mathbb{R}^{M\times \hat{d}M}$, $B \in \mathbb{R}^{\hat{d}M\times 2M}$, and $A \in \mathbb{R}^{\hat{d}M}$, such that $\hat{\mathcal{R}}(\mathbf{U}^\eta, \tilde{\mathbf{G}}^\eta) := S\sigma(B[\mathbf{U}^\eta, \tilde{\mathbf{G}}^\eta]^{\mathrm{T}} + A)$ satisfies

$$||\mathcal{R}(\mathbf{U}^\eta, \tilde{\mathbf{G}}^\eta) - \hat{\mathcal{R}}(\mathbf{U}^\eta, \tilde{\mathbf{G}}^\eta)||_{l^2(\mathbb{R}^M)} = ||\mathcal{R}(\mathbf{U}^\eta, \tilde{\mathbf{G}}^\eta) - S\sigma(B[\mathbf{U}^\eta, \tilde{\mathbf{G}}^\eta]^{\mathrm{T}} + A)||_{l^2(\mathbb{R}^M)} \leq \hat{\varepsilon} = \frac{m\varepsilon}{2(1+m)^L},$$

where m is the contraction parameter of $\mathcal{R}$, as defined in Assumption 1. By this construction, we know that S has independent rows. Denoting $\tilde{d} := \hat{d} + 1 > 0$, there exists the right inverse of S, which we denote as $S^+ \in \mathbb{R}^{(\tilde{d}-1)M\times M}$, such that

$$SS^+ = I_M, \quad S^+S := \tilde{I}_{(\tilde{d}-1)M},$$

where I_M is the M by M identity matrix, $\tilde{I}_{(\tilde{d}-1)M}$ is a $(\tilde{d}-1)M$ by $(\tilde{d}-1)M$ block matrix with each of its element being either 1 or 0. Hence, for any vector $Z \in \mathbb{R}(\tilde{d}-1)M$, we have $\sigma(\tilde{I}_{(\tilde{d}-1)M}Z) = \tilde{I}_{(\tilde{d}-1)M}\sigma(Z)$. Moreover, we note that S has a very special structure: from the $((i-1)(\tilde{d}-1)+1)$-th to the $(i(\tilde{d}-1))$-th column of S, all nonzero elements are on its i-th row. Correspondingly, we can also choose S^+ to have a special structure: from the $((i-1)(\tilde{d}-1)+1)$-th to the $(i(\tilde{d}-1))$-th row of S^+, all nonzero elements are on its i-th column. Hence, when multiplying S^+ with $\mathbf{U}$, there will be no entanglement between different components of $\mathbf{U}$. That means, S^+ can be seen as a pointwise weight function.

We now construct the MetaP as follows. In this construction, we choose the feature dimension as $d_h := \tilde{d}M$. With the input $[\mathbf{U}^0, \mathbf{G}^\eta] \in \mathbb{R}^{2M}$, for the lift layer we set

$$P^\eta := \mathbf{1}_{[M,1]} \otimes \begin{bmatrix} S^+ & \mathbf{0} \\ \mathbf{0} & D^\eta \end{bmatrix} = \underbrace{\begin{bmatrix} S^+ & \mathbf{0} & S^+ & \mathbf{0} & \cdots & S^+ & \mathbf{0} \\ \mathbf{0} & D^\eta & \mathbf{0} & D^\eta & \cdots & \mathbf{0} & D^\eta \end{bmatrix}^{\mathrm{T}}}_{\text{repeated for } M \text{ times}} \in \mathbb{R}^{d_h M\times 2M},$$

and $\mathbf{p}^\eta := \mathbf{0} \in \mathbb{R}^{d_h M}$. Here, $D^\eta := \mathrm{diag}[1/\mathbf{F}_1[\mathbf{b}^\eta](\mathbf{x}_1), \cdots, 1/\mathbf{F}_1[\mathbf{b}^\eta](\mathbf{x}_M)]$. As such, the initial layer of feature is then given by

$$\mathbf{H}(0) = P^\eta([\mathbf{U}^0, \mathbf{G}^\eta]^{\mathrm{T}}) = \mathbf{1}_{[M,1]} \otimes [S^+\mathbf{U}^0, D^\eta\mathbf{G}^\eta]^{\mathrm{T}} = \mathbf{1}_{[M,1]} \otimes [S^+\mathbf{U}^0, \tilde{\mathbf{G}}^\eta]^{\mathrm{T}} \in \mathbb{R}^{dM}.$$

Here, we point out that P^η and $\mathbf{p}^\eta$ can be seen as pointwise weight and bias functions, respectively.

Next we construct the shared iterative layer $\mathcal{J}$, by setting

$$V := \begin{bmatrix} \tilde{I}_{(\tilde{d}-1)M}B/M \\ 0 \end{bmatrix} \begin{bmatrix} S/\Delta t & \mathbf{0} \\ \mathbf{0} & I_M/\Delta t \end{bmatrix}, \ \tilde{V} := \mathbf{1}_{[M,M]} \otimes V, \ \text{ and } \boldsymbol{C} := \mathbf{1}_{[M,1]} \otimes \begin{bmatrix} \tilde{I}_{(\tilde{d}-1)M}A/\Delta t \\ \mathbf{0} \end{bmatrix}.$$

Note that $\tilde{V}$ is independent of η, and falls into the formulation of V, by letting $R_1 = V$ and $R_2 = R_2 = \cdots = R_M = W = 0$. For the $l+1$-th layer of feature vector, we then arrive at

$$\begin{aligned}
&\mathbf{H}((l+1)\Delta t) = \mathbf{H}(l\Delta t) + \Delta t\sigma\left(\tilde{V}\mathbf{H}(l\Delta t) + \mathbf{C}\right) \\
=&\mathbf{H}(l\Delta t) + \left(I_M \otimes \begin{bmatrix} S^+S & \mathbf{0} \\ \mathbf{0} & I_M \end{bmatrix}\right)\sigma\left(\left(\mathbf{1}_{[M,1]} \otimes \begin{bmatrix} B/M \\ \mathbf{0} \end{bmatrix}\right)\left(\mathbf{1}_{[1,M]} \otimes \begin{bmatrix} S & \mathbf{0} \\ \mathbf{0} & I_M \end{bmatrix}\right)\mathbf{H}(l\Delta t) + \mathbf{1}_{[M,1]} \otimes \begin{bmatrix} A \\ \mathbf{0} \end{bmatrix}\right),
\end{aligned}$$

where $\mathbf{H}(l\Delta t) = [\hat{\mathbf{h}}_1^{l\Delta t}, \hat{\mathbf{h}}_2^{l\Delta t}, \ldots, \hat{\mathbf{h}}_{2M-1}^{l\Delta t}, \hat{\mathbf{h}}_{2M}^{l\Delta t}]^{\mathrm{T}}$ denotes the (spatially discretized) hidden layer feature at the $l-$th iterative layer of the IFNO. Subsequently, we note that the second part of the feature vector, $\hat{\mathbf{h}}_{2j}^{l\Delta t} \in \mathbb{R}^M$, satisfies

$$\hat{\mathbf{h}}_{2j}^{(l+1)\Delta t} = \hat{\mathbf{h}}_{2j}^{l\Delta t} = \cdots = \hat{\mathbf{h}}_{2j}^{0} = \tilde{\mathbf{G}}^{\eta}, \quad \forall l = 0, \cdots, L-1, \forall j = 1, \cdots, M$$

Hence, the first part of the feature vector, $\hat{\mathbf{h}}_{2j-1}^{l\Delta t} \in \mathbb{R}^{(\tilde{d}-1)M}$, satisfies the following iterative rule:

$$\hat{\mathbf{h}}_{2j-1}^{(l+1)\Delta t} = \hat{\mathbf{h}}_{2j-1}^{l\Delta t} + S^{+}S\sigma(B[S\hat{\mathbf{h}}_{2j-1}^{l\Delta t}, \tilde{\mathbf{G}}^{\eta}]^{\mathrm{T}} + A), \quad \forall l = 0, \cdots, L-1, \forall j = 1, \cdots, M,$$

and

$$\hat{\mathbf{h}}_{1}^{(l+1)\Delta t} = \hat{\mathbf{h}}_{3}^{(l+1)\Delta t} = \cdots = \hat{\mathbf{h}}_{2M-1}^{(l+1)\Delta t}.$$

Finally, for the projection layer $\mathcal{Q}$, we set the activation function in the projection layer as the identity function, $Q_1 := I_{d_h M}$ (the identity matrix of size $d_h M$), $Q_2 := [S, \mathbf{0}] \in \mathbb{R}^{M \times d_h M}$, $\mathbf{q}_1 := \mathbf{0} \in \mathbb{R}^{d_h M}$, and $\mathbf{q}_2 := \mathbf{0} \in \mathbb{R}^{M}$. Denoting the output $\mathbf{U}^{\eta} := \mathcal{Q}_{\theta_Q} \circ (\mathcal{J}_{\theta_I})^{L} \circ \mathcal{P}_{\theta_P^{\eta}}([\mathbf{U}^0, \mathbf{G}^{\eta}]^{\mathrm{T}})$, we now show that $\mathbf{U}^{\eta}$ can approximate $\mathbf{U}^{\eta,*}$ with a desired accuracy ε:

$$\begin{aligned}
||\mathbf{U}^{\eta} - \mathbf{U}^{\eta,*}|| &\leq ||\mathbf{U}^{\eta} - \mathbf{U}^{L}||_{l^2(\mathbb{R}^M)} + ||\mathbf{U}^{L} - \mathbf{U}^{\eta,*}||_{l^2(\mathbb{R}^M)} \\
&\leq ||S\hat{\mathbf{h}}_1^{L\Delta t} - \mathbf{U}^{L}||_{l^2(\mathbb{R}^M)} + \frac{\varepsilon}{2} \quad \textit{(by Assumption 2)} \\
&\leq ||S\hat{\mathbf{h}}_1^{(L-1)\Delta t} - \mathbf{U}^{L-1}||_{l^2(\mathbb{R}^M)} + ||\hat{\mathcal{R}}(S\hat{\mathbf{h}}_1^{(L-1)\Delta t}, \tilde{\mathbf{G}}) - \mathcal{R}(\mathbf{U}^{L-1}, \tilde{\mathbf{G}})||_{l^2(\mathbb{R}^M)} + \frac{\varepsilon}{2} \\
&\leq ||S\hat{\mathbf{h}}_1^{(L-1)\Delta t} - \mathbf{U}^{L-1}||_{l^2(\mathbb{R}^M)} + ||\hat{\mathcal{R}}(S\hat{\mathbf{h}}_1^{(L-1)\Delta t}, \tilde{G}b) - \mathcal{R}(S\hat{\mathbf{h}}_1^{(L-1)\Delta t}, \tilde{G}b)||_{l^2(\mathbb{R}^M)} \\
&+ ||\mathcal{R}(S\hat{\mathbf{h}}_1^{(L-1)\Delta t}, \tilde{G}b) - \mathcal{R}(\mathbf{U}^{L-1}, \tilde{G}b)||_{l^2(\mathbb{R}^M)} + \frac{\varepsilon}{2} \\
&\leq (1+m)||S\hat{\mathbf{h}}_1^{(L-1)\Delta t} - \mathbf{U}^{L-1}||_{l^2(\mathbb{R}^M)} + \frac{m\varepsilon}{2(1+m)^L} + \frac{\varepsilon}{2} \quad \textit{(by Lemma 1 and Assumption 1)} \\
&\leq \frac{m\varepsilon}{2(1+m)^L}(1 + (1+m) + (1+m)^2 + \cdots + (1+m)^{L-1}) + \frac{\varepsilon}{2} \\
&\leq \frac{\varepsilon}{2} + \frac{\varepsilon}{2} = \varepsilon.
\end{aligned}$$

□

B Data Generation and Training Details

In the following we briefly describe the empirical process of generating datasets, and the settings employed in running of each algorithm. For a fair comparison, for each algorithm, we tune the hyperparameters, including the learning rate from $\{0.1, 0.01, 0.001, 0.0001, 0.00001, 0.000001\}$, the decay rate from $\{0.5, 0.7, 0.9\}$, the weight decay parameter from $\{0.01, 0.001, 0.0001, 0.00001, 0.000001\}$, and the inner loop learning rate for MAML and ANIL from $\{0.01, 0.001, 0.0001, 0.00001, 0.000001\}$, to minimize the error on a separate validation dataset. In all experiments we decrease the learning rate with a ratio of learning rate decay rate every 100 epochs. The code and the processed datasets will be publicly released at github for readers to reproduce the experimental results.

B.1 Example 1: Synthetic Data Sets

B.1.1 Data Generation

In the synthetic data example, we consider the modeling problem of a hyperelastic, anisotropic, fiber-reinforced material, and seek to find its displacement field $\mathbf{u} : [0, 1]^2 \to \mathbb{R}^2$ under different boundary loadings. In this problem, the specimen is assumed to be subject to a uniaxial tension $T_y(\mathbf{x})$ on the top edge (see Figure 4(a)). To generate training and test samples, the Holzapfel-Gasser-Odgen (HGO) model (Holzapfel et al., 2000) was employed to describe the constitutive behavior of the material in this example, with its strain energy density function given as:

$$\begin{aligned}
\eta = &\frac{E}{4(1+\nu)}(\bar{I}_1 - 2) - \frac{E}{2(1+\nu)}\ln(J) \\
&+ \frac{k_1}{2k_2}\left(\exp\left(k_2\langle S(\alpha)\rangle^2\right) + \exp\left(k_2\langle S(-\alpha)\rangle^2\right) - 2\right) + \frac{E}{6(1-2\nu)}\left(\frac{J^2-1}{2} - \ln J\right).
\end{aligned}$$

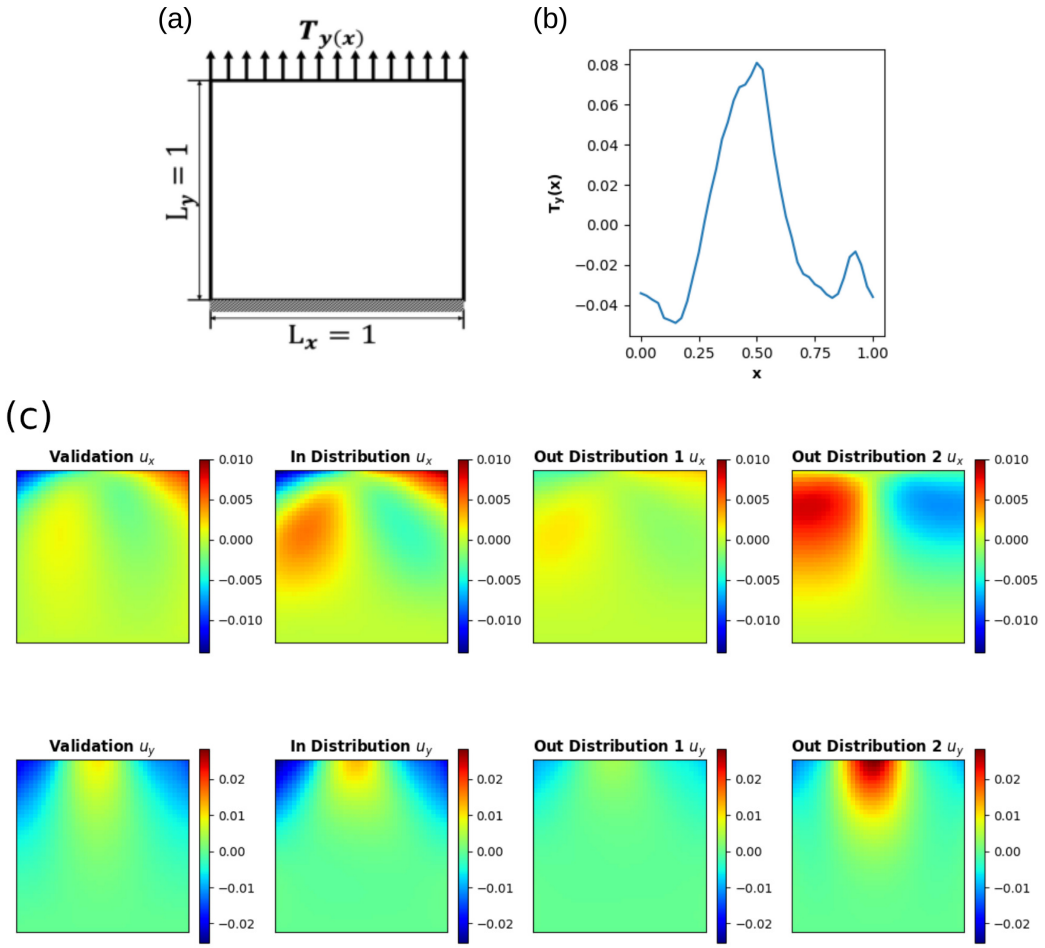

Figure 4: Problem setup of example 1: the synthetic data sets. (a) A unit square specimen subject to uniaxial tension with Neumann-type boundary condition. (b) & (c) Visualization of an instances of the loading field $T_y(x)$, and the corresponding ground-truth solutions $\mathbf{u}^\eta(\mathbf{x})$ from the in-distribution and out-of-distribution tasks, showing the solution diversity across different tasks, due to the change of underlying hidden material parameter set.

Here, $\langle\cdot\rangle$ denotes the Macaulay bracket, and the fiber strain of the two fiber groups is defined as:

$$S(\alpha) = \frac{\overline{I}_4(\alpha) - 1 + |\overline{I}_4(\alpha) - 1|}{2}.$$

where k_1 and k_2 are fiber modulus and the exponential coefficient, respectively, c_{10} is the moduli for the non-fibrous ground matrix, E is the Young's modulus, and ν is the Poisson ratio. Moreover, $\overline{I}_1 = \text{tr}(\mathbf{C})$ is the is the first invariant of the right Cauchy-Green tensor $\mathbf{C} = \mathbf{F}^T\mathbf{F}$, $\mathbf{F}$ is the deformation gradient, and J is related with $\mathbf{F}$ such that $J = \det \mathbf{F}$. For the fiber group with angle direction α from the reference direction, $\overline{I}_4(\alpha) = \mathbf{n}^T(\alpha)\mathbf{C}\mathbf{n}(\alpha)$ is the fourth invariant of the right Cauchy-Green tensor $\mathbf{C}$, where $\mathbf{n}(\alpha) = [\cos(\alpha), \sin(\alpha)]^T$. To generate samples for different specimens,different specimens (tasks) correspond to different material parameter sets, $\{k_1, k_2, E, \nu, \alpha\}$. For the training tasks, the validation task, and the in-distribution (ID) test task, their physical parameters are sampled from: $k_1, k_2 \sim \mathcal{U}[0.1, 1]$, $E \sim \mathcal{U}[0.55, 1.5]$, $\nu \sim \mathcal{U}[0.01, 0.49]$, and $\alpha \sim \mathcal{U}[\pi/10, \pi/2]$. For the two out-of-distribution (OOD) test tasks, we sample their parameters following $k_1, k_2 \sim \mathcal{U}[1, 1.9]$, $E \sim \mathcal{U}[1.5, 2] \cup \mathcal{U}[0.5, 0.55]$, $\nu \sim \mathcal{U}[0.01, 0.49]$[4], and $\alpha \sim \mathcal{U}[\pi/2, 3\pi/4] \cup [0, \pi/10]$. To generate the high-fidelity (ground-truth) dataset, we sampled 500 different vertical traction conditions $T_y(\mathbf{x})$ on the top edge from a random field, following the algorithm in Lang & Potthoff (2011); Yin et al. (2022b). In particular, $T_y(\mathbf{x})$ is taken as the restriction of a 2D random field, $\phi(\mathbf{x}) = \mathcal{F}^{-1}(\gamma^{1/2}\mathcal{F}(\Gamma))(\mathbf{x})$, on the top edge. Here, $\Gamma(\mathbf{x})$ is a Gaussian white noise random field on $\mathbb{R}^2$, $\gamma = (w_1^2 + w_2^2)^{-\frac{5}{4}}$ represents a correlation function, and w_1, w_2 are the wave numbers on x and y directions, respectively. Then, for each sampled traction loading, we solved the displacement field on the entire domain by minimizing potential energy using the finite element method implemented in FEniCS (Alnæs et al., 2015). In particular, the displacement filed was approximated by continuous piecewise linear finite elements with triangular mesh, and the grid size was taken as 0.025. Then, the finite element solution was interpolated onto χ, a structured 41×41 grid which will be employed as the discretization in our neural operators.

To visualize the domain characteristics for tasks, the distribution of each parameter for training, validation and test tasks are demonstrated in Figure 5, and the corresponding solution fields are plotted in Figure 4(c), showing the diversity across different tasks due to the change of underlying hidden material parameter set, $\{k_1, k_2, E, \nu, \alpha\}$. From Figures 5 and 4(c), one can see that OOD Task1 corresponds a stiffer material (with large Young's modulus E) and hence smaller deformation subject to the same loading $T_y(\mathbf{x})$. On the other hand, OOD Task2 corresponds a softer material (with small Young's modulus E) and larger deformation. Therefore, the material response of OOD Task1 specimen is more likely to lie in a linear region, which is easier to learn and explains the relatively small test error on this task. On the other hand, the material response of OOD Task2 is more nonlinear and hence complex due to larger deformation, as shown in Figure 4(c), and results in the relatively larger test error in Figure 2.

B.1.2 Algorithm Settings

Base model: As the base model for all algorithms, we construct an architecture for IFNO (You et al., 2022c) as follows. First, the input loading field instance $\mathbf{g}(\mathbf{x}) \in \mathcal{A}$ is lifted to a higher dimensional representation via lift layer $\mathcal{P}[\mathbf{g}](\mathbf{x})$, which is parameterized as a 1-layer feed forward linear layer with width (3,32). Then for the iterative layer in equation 1, we implement $\mathcal{F}^{-1}[\mathcal{F}[\kappa(\cdot; \mathbf{v})] \cdot \mathcal{F}[\mathbf{h}(\cdot, l\Delta t)]](\mathbf{x})$ with 2D fast Fourier transform (FFT) with input channel and output channel widths both set as 32 and the truncated Fourier modes set as 8. The local linear transformation parameter, W, is parameterized as a 1-layer feed forward network with width (32,32). In the projection layer, a 2-layer feed forward network with width (32,128,2) is employed. To accelerate the training procedure, we apply the shallow-to-deep training technique to initialize the optimization problem. In particular, we start from the NN model with depth $L = 1$, train until the loss function reaches a plateau, then use the resultant parameters to initialize the parameters for the next depth, with $L = 2$, $L = 4$, and $L = 8$. In the synthetic experiments, we set the layer depth as $L = 8$.

MetaP: We split the total 60 training tasks to two groups: 59 tasks for the purpose of training and 1 task for the purpose of validation. During the meta-train phase, we train for the task-wise parameters

[4]Here we sample both ID and OOD tasks from the same range of ν, due to the fact that $[0.01, 0.49]$ is the range of Poisson ratio for common materials (Bischofs & Schwarz, 2005).

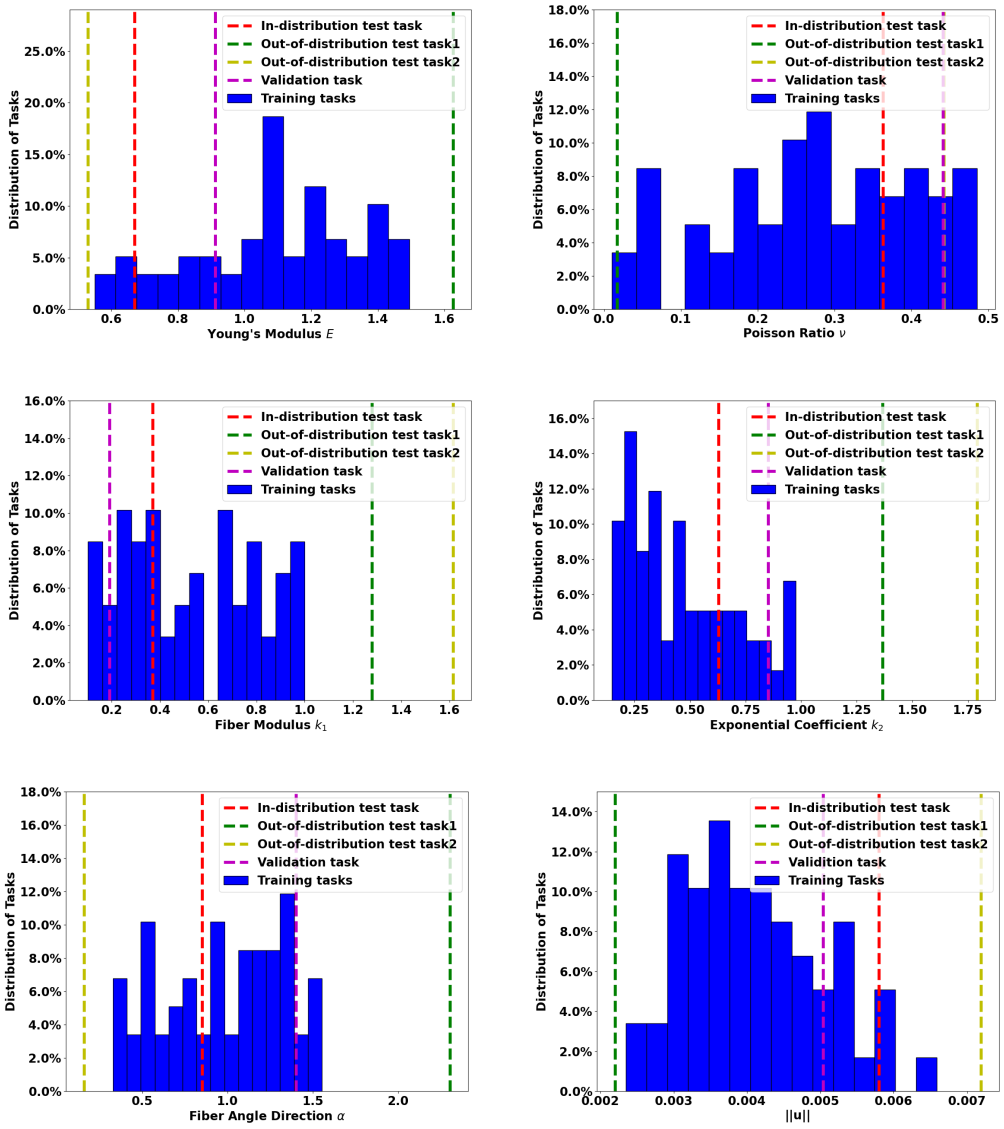

Figure 5: Distribution of physical parameters of different tasks, and the resultant magnitude of material response, $||u^{\eta}(\mathbf{x})||_{L^2(\Omega)}$, on an exemplar loading instance shown in Figure 4(b).

θ_P^η and the common parameters θ_I and θ_Q on all 59 tasks, with the context set of 500 samples on each task. After meta-train phase, we load θ_I and θ_Q and the averaged θ_P^η among all 59 tasks as initialization, then tune the hyperparameters based on the validation task. In particular, the 500 samples on the validation task is split into two parts: 300 samples are reserved for the purpose of training (as the context set) and the rest 200 samples are used for evaluation (as the target set). Then we train for the lift layer on the validation task, and tune the learning rate, the decay rate, and the weight decay parameter for different context set sizes (N^{test}), to minimize the loss on the target set. Based on the chosen hyperparameters, we perform the test on the test task by training for the lift layer on different numbers of samples on its context set, then evaluate and report the performance based on its target set. We repeat the procedure on the test task with selected hyperparemeters with different 5 random seeds, and calculate means and standard errors for the resultant test errors on target set.

MAML&ANIL: For MAML and ANIL, we use the same architecture as the base model, and also split the training tasks for the purpose of training and validation as in MetaP. During the meta-train phase, for each task we randomly split the available 500 samples to two sets: 250 samples in the support set used for inner loop updates, and the rest in the target set for outer loop updates. During the inner loop update, we train for the task-wise parameter with one epoch, following the standard settings of MAML and ANIL (Finn et al., 2017; Raghu et al., 2019). Then, the model hyperparameters, including the learning rate, weight decay, decay rate, and inner loop learning rate, are tuned. In the meta-test phase, we load the initial parameter and train for all parameters (in MAML) or the last-layer parameters (in ANIL) until the optimization algorithm converges. Similar as in MetaP, we first tune the hyperparameters on the validation task, then evaluate the performance on the test task.

B.2 Example 2: Mechnical MNIST

B.2.1 Data Settings

Mechanical MNIST is a benchmark dataset of heterogeneous material undergoing large deformation, modeld by the Neo-Hookean material with a varying modulus converted from the MNIST bitmap images (Lejeune, 2020). In this example, we randomly select 102 specimens corresponding to the hand-written number "1". On each specimen, we have 32 loading/response data pairs on a structured 27 by 27 grid, under the uniaxial extension, shear, equibiaxial extension, and confined compression load scenarios, respectively. All 102 specimens are splitted into three groups: 100 specimens for the purpose of training in the meta-train stage, 1 specimen for validation, and 1 specimen for test. On the validation and test tasks, we reserve a target set consisting of 20 data pairs for the purpose of evaluation, then use the rest as the context set.

B.2.2 Algorithm Settings

Base model: As the base model for all algorithms, we construct two IFNO architectures, for the prediction of u_x and u_y, the displacement fields in the x- and y-directions, respectively. On each architecture, the input loading field instance $\mathbf{g}(\mathbf{x}) \in \mathcal{A}$ is mapped to a higher dimensional representation via a lifting layer $\mathcal{P}[\mathbf{g}](\mathbf{x})$ parameterized as a 1-layer feed forward linear layer with width (4,64). Then for the iterative layer in equation 1, we set the number of truncated Fourier mode as 13, and parameterize the local linear transformation parameter, W, as a 1-layer feed forward network with width (64,64). In the projection layer, a 2-layer feed forward network with width (64,128,1) is employed. In this example we also apply the shallow-to-deep technique to accelerate the training, and set the layer depth as $L = 8$.

MetaP: During the meta-train phase, we train for the task-wise parameters θ_P^η and the common parameters θ_I and θ_Q on all 100 training tasks, with the context set of 32 samples on each task. After the meta-train phase, we load θ_I and θ_Q and the averaged θ_P^η among all 100 tasks as initialization, then train for θ_P on the validation task. In particular, the 32 samples on the validation task is split into two parts: 12 samples are reserved for the purpose of training (as the context set) and the rest 20 samples are used for the purpose of evaluation (as the target set). Then we train for the lift layer on the validation task, and tune the learning rate, the decay rate, and the weight decay parameter for different context set sizes (N^{test}), to minimize the loss on the target set. Based on the chosen hyperparameters, we perform the meta-test phase on the test task by training for the lift layer on

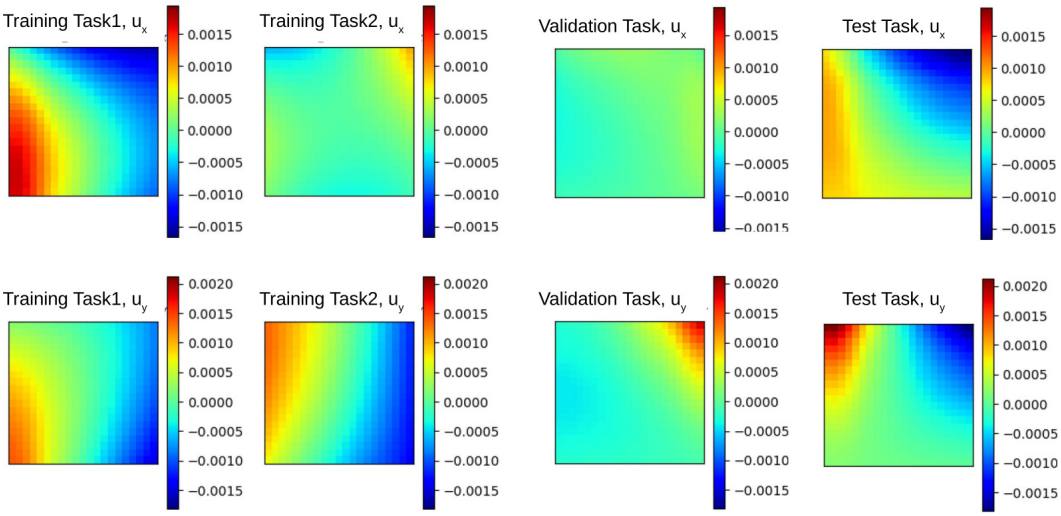

Figure 6: Visualization of the processed dataset in example 3: learning the biological tissue responses. Subject to the same loading instance, different columns show the corresponding ground-truth solutions $\mathbf{u}^n(\mathbf{x})$ from different tasks, showing the solution diversity across different tasks due to the change of underlying hidden material parameter field.

different numbers of samples on its context set, then evaluate and report the performance based on its target set.

MAML&ANIL: For MAML and ANIL, we use the same architecture as the base model, and also split the training tasks for the purpose of training and validation as in MetaP. During the meta-train phase, for each task we randomly split the available 32 samples to two sets: 16 samples in the support set used for inner loop updates, and the rest in the target set for outer loop updates. During the inner loop update, we also follow the standard settings of MAML and ANIL (Finn et al., 2017; Raghu et al., 2019), and tune the hyperparameters following the same procedure as elaborated above for Example 1.

B.3 Example 3: Experimental Measurements on Biological Tissues

B.3.1 Data Generation

We now briefly provide the data generation procedure for the tricuspid valve anterior leaflet (TVAL) response modeling example. In this problem, the constitutive equations and material microstructure are both unknown, and the dataset has unavoidable measurement noise. To generate the data, we firstly followed the established biaxial testing procedure, including acquisition of a healthy porcine heart and retrieval of the TVAL Ross et al. (2019); Laurence et al. (2019). Then, we sectioned the leaflet tissue and applied a speckling pattern to the tissue surface using an airbrush and black paint Zhang & Arola (2004); Lionello & Cristofolini (2014); Palanca et al. (2016). The painted specimen was then mounted to a biaxial testing device (BioTester, CellScale, Waterloo, ON, Canada). To generate samples for each specimen, we performed 7 protocols of displacement-controlled testing to target various biaxial stresses: $P_{11} : P_{22} = \{1:1, 1:0.66, 1:0.33, 0.66:1, 0.33:1, 0.05:1, 1:0.1\}$. Here, P_{11} and P_{22} denote the first Piola-Kirchhoff stresses in the x- and y-directions, respectively. Each stress ratio was performed for three loading/unloading cycles. Throughout the test, images of the specimen were captured by a CCD camera, and the load cell readings and actuator displacements were recorded at 5 Hz. After testing, the acquired images were analyzed using the digital image correlation (DIC) module of the BioTester's software. The pixel coordinate locations of the DIC-tracked grid were then exported and extrapolated to a 21 by 21 uniform grid.

In this example, we have the DIC measurements on 14 specimens, with 500 data pairs of loadings and material responses from the 7 protocols on each specimen. These specimens are divided into three groups: 12 for the purpose of meta-train, 1 for validation, and 1 for test. To demonstrate the

diversity of these specimens due to the material heterogeneity in biological tissues, in Figure 6 we plot the processed displacement field of two exemplar training specimens and the validation and test specimens.

B.3.2 ALGORITHM SETTINGS

Base model: As the base model, we first construct the lifting layer as a 1-layer feed forward linear layer with width (4,16). Then for the iterative layer in we keep 8 truncated Fourier modes and parameterize the local linear transformation parameter, W, a 1-layer feed forward network with width (16,16). In the projection layer, a 2-layer feed forward network with width (16,64,1) is employed. We construct two 4-layer IFNO architectures, for the prediction of u_x and u_y, the displacement fields in the x- and y-directions, respectively.

MetaP: During the meta-train phase, we train for the task-wise parameters θ_P^η and the common parameters θ_I and θ_Q on all 12 tasks, with the context set of 500 samples on each task. After meta-train phase, we load θ_I and θ_Q and the averaged θ_P^η among all 12 tasks as initialization, then tune the hyperparameters based on the validation task. In particular, the 500 samples on the validation task is splited into two parts: 300 samples are reserved for the purpose of training (as the context set) and the rest 200 samples are used for evaluation (as the target set). Based on the chosen hyperparameters, we perform the test on the test task by training for the lift layer on different numbers of samples on its context set, then evaluate and report the performance based on its target set.

MAML&ANIL: For MAML and ANIL, we use the same architecture as base model, and also split the training tasks for the purpose of training and validation as in MetaP. During the meta-train phase, for each task we randomly split the available 500 samples to two sets: 250 samples in the support set used for inner loop updates, and the rest in the target set for outer loop updates. During the inner loop update, we train for the task-wise parameter with one epoch, following the standard settings of MAML and ANIL (Finn et al., 2017; Raghu et al., 2019).