TYPE Original Research
PUBLISHED 03 August 2023
DOI 10.3389/fams.2023.1206500

OPEN ACCESS

EDITED BY
Haizhao Yang,
Purdue University, United States

REVIEWED BY
Kent-Andre Mardal,
University of Oslo, Norway
Qifeng Liao,
ShanghaiTech University, China

*CORRESPONDENCE
Mingchao Cai
✉ Mingchao.Cai@morgan.edu

RECEIVED 15 April 2023
ACCEPTED 14 July 2023
PUBLISHED 03 August 2023

CITATION
Cai M, Gu H, Hong P and Li J (2023) A combination of physics-informed neural networks with the fixed-stress splitting iteration for solving Biot's model. *Front. Appl. Math. Stat.* 9:1206500. doi: 10.3389/fams.2023.1206500

A combination of physics-informed neural networks with the fixed-stress splitting iteration for solving Biot's model

Mingchao Cai[1]*, Huipeng Gu[2], Pengxiang Hong[2] and Jingzhi Li[2]

[1]Department of Mathematics, Morgan State University, Baltimore, MD, United States, [2]Department of Mathematics, Southern University of Science and Technology, Shenzhen, Guangdong, China

Introduction: Biot's consolidation model in poroelasticity describes the interaction between the fluid and the deformable porous structure. Based on the fixed-stress splitting iterative method proposed by Mikelic et al. (Computat Geosci, 2013), we present a network approach to solve Biot's consolidation model using physics-informed neural networks (PINNs).

Methods: Two independent and small neural networks are used to solve the displacement and pressure variables separately. Accordingly, separate loss functions are proposed, and the fixed stress splitting iterative algorithm is used to couple these variables. Error analysis is provided to support the capability of the proposed fixed-stress splitting-based PINNs (FS-PINNs).

Results: Several numerical experiments are performed to evaluate the effectiveness and accuracy of our approach, including the pure Dirichlet problem, the mixed partial Neumann and partial Dirichlet problem, and the Barry-Mercer's problem. The performance of FS-PINNs is superior to traditional PINNs, demonstrating the effectiveness of our approach.

Discussion: Our study highlights the successful application of PINNs with the fixed-stress splitting iterative method to tackle Biot's model. The ability to use independent neural networks for displacement and pressure offers computational advantages while maintaining accuracy. The proposed approach shows promising potential for solving other similar geoscientific problems.

KEYWORDS

physics-informed neural networks, the fixed-stress method, Biot's model, iterative algorithm, separated networks

1. Introduction

Biot's consolidation model in poroelasticity describes the interaction between fluid flow and the porous structure it saturates. This model was first proposed by Biot [1], and has a wide range of applications, including biomechanics [2] and petroleum engineering [3]. The partial differential equations (PDEs) for the quasi-static Biot system in a bounded domain $\Omega \subset \mathbb{R}^d$ (where $d = 2$ or 3) over the time interval $(0, T]$ are as follows:

$$\partial_t \left(\frac{1}{M} p + \alpha \nabla \cdot \boldsymbol{u} \right) - \nabla \cdot K \nabla p = f \qquad \text{in } \Omega \times (0, T], \tag{1}$$

$$-\nabla \cdot \sigma(\boldsymbol{u}) + \alpha \nabla p = \boldsymbol{g} \qquad \text{in } \Omega \times (0, T]. \tag{2}$$

Here, $\boldsymbol{u}$ is the displacement of solid, p is the fluid pressure, $\boldsymbol{g}$ is the body force, f is a source or sink term, $\sigma(\boldsymbol{u}) = 2\mu\varepsilon(\boldsymbol{u}) + \lambda \mathrm{div}\boldsymbol{u}\boldsymbol{I}$ with $\varepsilon(\boldsymbol{u}) = \frac{1}{2}\left[\nabla\boldsymbol{u} + (\nabla\boldsymbol{u})^T\right]$ being the strain tensor. The Lamé constants λ and μ are expressed in terms of the Young's modulus E and the Poisson ratio ν as

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)}, \quad \mu = \frac{E}{2(1+\nu)}. \tag{3}$$

Other physical parameters are the Biot-Willis constant $\alpha > 0$, which is close to 1, Biot's modulus $M > 0$, and hydraulic conductivity K. Equation (1) represents the mass conservation and Equation (2) means forces balance. For ease of presentation, we assume the following pure Dirichlet conditions.

$$\boldsymbol{u} = \boldsymbol{u}_d, \quad p = p_d \qquad \text{on } \partial\Omega \times (0, T]. \tag{4}$$

The initial conditions are

$$\boldsymbol{u} = \boldsymbol{u}_0,\ p = p_0 \qquad \text{in } \Omega \times \{0\}. \tag{5}$$

The discussion of the existence and uniqueness of the solution of Biot's system (1)–(5) can be found in [4–6]. In this work, we focus on the algorithm aspect.

Several classical numerical methods have been proposed to solve this problem, including finite volume methods [7], virtual element methods [8], and mixed finite element methods [9]. Biot's model is a multiphysics problem involving both linear elasticity and porous media flow. Numerical difficulties such as elastic locking and pressure oscillations can arise, especially for models based on two-field formulations [10–13]. To overcome these difficulties, various methods have been proposed, such as the discontinuous Galerkin method [14], stabilized finite element methods [11, 15], and three-field or four-field reformulations using inf-sup stable finite element pairs [12, 16–18]. These methods may face challenges in terms of large computational overhead. The fixed-stress splitting iterative method [19, 20] is proposed to address this issue. This method breaks down the original problem into two subproblems and solves them in an iterative manner, rather than solving the entire system at once. The method relies on the contraction mapping principle to prove its convergence [19] and has been shown to be efficient through various studies. Further studies have been carried out based on the fixed-stress splitting iterative method, such as the analysis of the relationship between the convergence rate and the stabilization parameter [21–23] and the implementation of a parallel-in-time strategy to speed up the computations [24].

In recent years, deep neural networks (DNNs) have demonstrated impressive potential in solving partial differential equations (PDEs) with a wide range of applications in various domains. Among DNNs, physics-informed neural networks (PINNs) have become a popular class due to their ability to solve PDEs without meshing. PINNs have proven successful in solving high-dimensional problems and interface problems, and they can tackle inverse problems with slight modifications of the loss function [25–27]. Several studies have used PINNs to tackle the Biot's model [28–30], where the key advantage of PINNs over traditional methods such as finite element methods is their ability to avoid numerical difficulties arising from meshing and not requiring inf-sup stability. Therefore, from a flexibility perspective, deep neural network methods are preferable. However, most of these studies employ a monolithic approach to train the neural network solution, with only one study using a sequential training approach [27], which lacks theoretical analysis and only presents numerical experiments. In this paper, we propose a combination of physics-informed neural networks with fixed-stress splitting method (FS-PINNs) to solve Biot's equations. We employ two PINNs and incorporate them into the fixed-stress splitting iterative method. Our method involves two separate neural networks, one for solid displacement and another for fluid pressure, leading to faster convergence and lower computational cost than classical PINNs [31, 32]. Through a detailed analysis of the monotonic convergence of the fixed-stress splitting method, we present an error estimate for the solution of the proposed FS-PINNs. Future work could include refining the neural network architecture to improve the accuracy and further reduce the computational cost. In addition, the potential application of our approach to other PDE problems and multiphysics problems could be explored. Overall, the proposed FS-PINNs represent a promising approach for solving poroelastic models with faster convergence and lower computational cost.

The remaining sections of the paper are organized as follows. Section 2 provides an overview of the fixed-stress splitting iterative method for Biot's model. In Section 3, we introduce the fixed-stress splitting iterative PINNs (FS-PINNs) and present theoretical analyses to demonstrate their approximation properties. Section 4 presents numerical experiments to demonstrate the effectiveness of the proposed method. Finally, conclusions are drawn in Section 5.

2. Fixed-stress splitting method for Biot's equations

In this section, we introduce an iterative scheme proposed in [19], called the fixed-stress splitting method. Instead of solving Biot's model (1)–(5) in a monolithic way, this method decouples the original problem into two subproblems and solves them in an iterative manner. Given a large enough stabilization parameter β_{FS} and an initial guess $(p^0, \boldsymbol{u}^0)$, the standard fixed-stress splitting method computes a sequence of approximations as follows:
The first step: Given p^n and $\boldsymbol{u}^n$, we solve for p^{n+1} satisfying

$$\begin{aligned} &\left(\frac{1}{M} + \beta_{FS}\right)\partial_t p^{n+1} - \nabla\cdot K\nabla p^{n+1} = \\ &f - \alpha\nabla\cdot\partial_t\boldsymbol{u}^n + \beta_{FS}\partial_t p^n \quad \text{in } \Omega\times(0,T], &(6)\\ &p^{n+1} = p_0 \qquad \text{in } \Omega\times\{0\}, &(7)\\ &p^{n+1} = p_d \qquad \text{on } \partial\Omega\times(0,T]. &(8)\end{aligned}$$

The second step: Using p^{n+1}, we solve for $\boldsymbol{u}^{n+1}$ satisfying

$$\begin{aligned} -\nabla\cdot\sigma(\boldsymbol{u}^{n+1}) &= \boldsymbol{g} - \alpha\nabla p^{n+1} \qquad \text{in } \Omega\times(0,T], &(9)\\ \boldsymbol{u}^{n+1} &= \boldsymbol{u}_0 \qquad \text{in } \Omega\times\{0\}, &(10)\\ \boldsymbol{u}^{n+1} &= \boldsymbol{u}_d \qquad \text{on } \partial\Omega\times(0,T]. &(11)\end{aligned}$$

To derive the variational formulation of the fixed-stress problem (6)–(11), we define the proper functional spaces with

$H^1(\Omega)$ denoting the Hilbert subspace as follows.

$$W = \{p \in H^1([0,T]; H^1(\Omega)) \mid p|_{\partial\Omega} = p_d\},$$
$$\boldsymbol{V} = \{\boldsymbol{u} \in H^1([0,T]; \boldsymbol{H}^1(\Omega)) \mid \boldsymbol{u}|_{\partial\Omega} = u_d\}.$$

Multiplying (6), (9) by test functions, and applying the integration by parts, we obtain the variational problems: for a given $t \geq 0$, find $\{(p^{n+1}, \boldsymbol{u}^{n+1})\} \subset W \times \boldsymbol{V}$ such that

$$\begin{aligned}&\left(\frac{1}{M} + \beta_{FS}\right)\int_\Omega (\partial_t p^{n+1})\psi dx + K\int_\Omega (\nabla p^{n+1})(\nabla\psi)dx \\ &= \int_\Omega f\psi dx - \alpha\int_\Omega (\nabla\cdot\partial_t \boldsymbol{u}^n)\psi dx + \beta_{FS}\int_\Omega (\partial_t p^n)\psi dx, \qquad \forall\psi \in W,\end{aligned} \tag{12}$$

$$\begin{aligned}&2\mu\int_\Omega \varepsilon(\boldsymbol{u}^{n+1}) : \varepsilon(\boldsymbol{v})dx + \lambda\int_\Omega (\nabla\cdot\boldsymbol{u}^{n+1})(\nabla\cdot\boldsymbol{v})dx \\ &= \int_\Omega \boldsymbol{g}\boldsymbol{v}dx + \alpha\int_\Omega (\nabla\cdot\boldsymbol{v})p^{n+1}dx, \qquad \forall\boldsymbol{v} \in \boldsymbol{V}.\end{aligned} \tag{13}$$

Assume that $(p, \boldsymbol{u}) \in W \times \boldsymbol{V}$ is the unique solution of Biot's system (1)–(5). Given a large enough stabilization parameter β_{FS}, the sequence $\{(p^{n+1}, \boldsymbol{u}^{n+1})\} \subset W \times \boldsymbol{V}$ generated by Equations (12, 13) converges to the solution $(p, \boldsymbol{u})$. More precisely, we can state the following:

Theorem 2.1. The sequence $\{(p^{n+1}, \boldsymbol{u}^{n+1})\} \subset W \times \boldsymbol{V}$ generated by Equations (6)–(11) converges to $(p, \boldsymbol{u}) \in W \times \boldsymbol{V}$ for any $\beta_{FS} \geq \frac{\alpha^2}{2\left(\frac{2\mu}{d}+\lambda\right)}$. If one denotes $e_{\boldsymbol{u}}^{n+1} := \boldsymbol{u}^{n+1} - \boldsymbol{u}$ and $e_p^{n+1} := p^{n+1} - p$, then there holds

$$\int_0^T \int_\Omega (\partial_t e_p^{n+1})^2 dxdt \leq L\int_0^T \int_\Omega (\partial_t e_p^n)^2 dxdt, \tag{14}$$

where $L = \frac{\beta_{FS}}{\beta_{FS}+\frac{2}{M}}$ is a positive constant strictly smaller than 1.

Remark 2.2. We have included a convergence analysis of the fixed-stress splitting iterative method in the Appendix. In our proof, we demonstrate that the differences e_p^{n+1} and $e_{\boldsymbol{u}}^{n+1}$ approach zeros as n tends to infinity. Our proof helps to illustrate the convergence of the fixed-stress splitting method.

3. The classical PINNs and the FS-PINNs for solving Biot's model

In this section, we introduce the training procedure of the classical PINNs for solving Biot's model, and propose an iterative deep learning method combining the fixed-stress splitting method, called FS-PINNs. The proposed FS-PINNs consist of two independent PINNs, one for pressure, and the other for displacement. We then present the training procedure and a theoretical analysis of the proposed FS-PINNs.

3.1. Training procedure of the classical PINNs

The idea of using the classical PINNs to solve PDEs is from [25], which can be easily extended to solve Biot's model. The main idea is to design loss functions for fully connected neural networks. In this context, we provide a brief overview of the training procedure for solving problem (1)–(5), which is illustrated in Figure 1.

The functional form of a classical PINN is given as follows.

$$\hat{\boldsymbol{n}}(\boldsymbol{x};\theta_{\boldsymbol{n}}) = \left(\hat{p}(\boldsymbol{x};\theta_{\boldsymbol{n}}); \hat{\boldsymbol{u}}(\boldsymbol{x};\theta_{\boldsymbol{n}})\right) = \Sigma_{\boldsymbol{n}}^{L_n} \circ \Sigma_{\boldsymbol{n}}^{L_n - 1} \circ \cdots \circ \Sigma_{\boldsymbol{n}}^1(\boldsymbol{x}). \tag{15}$$

Here, $\hat{\boldsymbol{n}}$ is the network approximation of the exact solution $(p; \boldsymbol{u})$ with L_n hidden layers with the input $\boldsymbol{x} = (x_1, \cdots, x_d, t) \in \mathbb{R}^d \times [0,T]$. The collection of parameter $\theta_{\boldsymbol{n}}$ is given by

$$\theta_{\boldsymbol{n}} = \{\boldsymbol{W}_{\boldsymbol{n}}^1, \boldsymbol{W}_{\boldsymbol{n}}^2, \cdots, \boldsymbol{W}_{\boldsymbol{n}}^{L_n}, \boldsymbol{b}_{\boldsymbol{n}}^1, \boldsymbol{b}_{\boldsymbol{n}}^2, \cdots, \boldsymbol{b}_{\boldsymbol{n}}^{L_n}\}. \tag{16}$$

The mapping function $\Sigma_{\boldsymbol{n}}^i$ is defined as

$$\boldsymbol{x}_{\boldsymbol{n}}^i := \Sigma_{\boldsymbol{n}}^i(\boldsymbol{x}_{\boldsymbol{n}}^{i-1}) = \sigma_{\boldsymbol{n}}^i(\boldsymbol{W}_{\boldsymbol{n}}^i \boldsymbol{x}_{\boldsymbol{n}}^{i-1} + \boldsymbol{b}_{\boldsymbol{n}}^i), \quad i = 1, 2, \cdots, L_{\boldsymbol{n}}. \tag{17}$$

Here, $(\boldsymbol{W}_{\boldsymbol{n}}^i, \boldsymbol{b}_{\boldsymbol{n}}^i, \sigma_{\boldsymbol{n}}^i)$ represent the weights, biases, and activation functions of the i-th layer for network $\hat{\boldsymbol{n}}$. It is assumed that the number of neurons in each hidden layer is set to be the same $N_{\boldsymbol{n}}$ and that all activation functions $\sigma_{\boldsymbol{n}}^i$ are identical.

After setting up the architecture, we can begin training the parameter $\theta_{\boldsymbol{n}}$ using the procedure:

$$\theta^* = \arg\min_{\theta_{\boldsymbol{n}}} \mathcal{L}_{\boldsymbol{n}}(\boldsymbol{x};\theta_{\boldsymbol{n}}), \tag{18}$$

where $\mathcal{L}_{\boldsymbol{n}}$ is the loss function. The loss function for the classical PINNs generally consists of three components: the PDEs loss, the initial condition loss, and the boundary condition loss. The PDEs loss represents the residual of the governing equations, while the initial and boundary condition losses ensure that the predicted solution satisfies the initial and boundary conditions. The loss function is denoted by $\mathcal{L}_{\boldsymbol{n}}(\boldsymbol{x};\theta_{\boldsymbol{n}})$ and has the following representation.

$$\begin{aligned}\mathcal{L}_{\boldsymbol{n}}(\boldsymbol{x};\theta_{\boldsymbol{n}}) =& \frac{1}{N_1}\sum_{i=1}^{N_1}\left(\left\|\hat{p}(\boldsymbol{x}_1^i;\theta_{\boldsymbol{n}}) - p^i\right\|_{L^2}^2 + \left\|\hat{\boldsymbol{u}}(\boldsymbol{x}_1^i;\theta_{\boldsymbol{n}}) - \boldsymbol{u}^i\right\|_{L^2}^2\right) \\ &+ \frac{1}{N_2}\sum_{i=1}^{N_2}\left(\left\|\frac{1}{M}\partial_t\hat{p}(\boldsymbol{x}_2^i;\theta_{\boldsymbol{n}}) - \nabla\cdot K\nabla\hat{p}(\boldsymbol{x}_2^i;\theta_{\boldsymbol{n}}) + \right.\right. \\ &\alpha\nabla\cdot\partial_t\hat{\boldsymbol{u}}(\boldsymbol{x}_2^i;\theta_{\boldsymbol{n}}) - f(\boldsymbol{x}_2^i)\Big\|_{L^2}^2 \quad + \Big\| -\nabla\cdot\sigma(\hat{\boldsymbol{u}}(\boldsymbol{x}_2^i;\theta_{\boldsymbol{n}})) + \\ &\left.\left.\alpha\nabla\hat{p}(\boldsymbol{x}_2^i;\theta_{\boldsymbol{n}}) - \boldsymbol{g}(\boldsymbol{x}_2^i)\right\|_{L^2}^2\right).\end{aligned}$$

The above expression uses the mean square error (MSE) to measure errors. In this expression, $\{\boldsymbol{x}_1^i\}_{i=1}^{N_1}$ represents the initial and boundary training data, while $\{\boldsymbol{x}_2^i\}_{i=1}^{N_2}$ represents the collocation points in the domain. We then optimize $\theta_{\boldsymbol{n}}$ to obtain θ^*. For cases involving mixed boundary conditions, one can refer to [28] for more details.

3.2. Training procedure of the proposed FS-PINNs

The primary innovation of our approach is incorporating the fixed-stress strategy into the training procedure of PINNs. The proposed method includes two independent PINNs to solve for

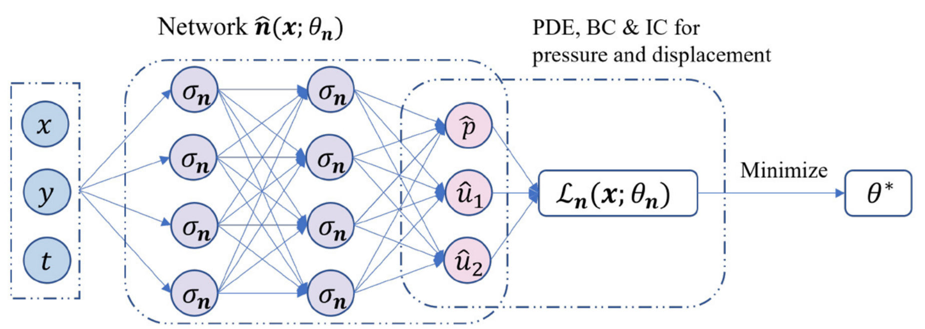
Network $\hat{\boldsymbol{n}}(\boldsymbol{x};\theta_{\boldsymbol{n}})$
PDE, BC & IC for pressure and displacement
x
y
t
$\sigma_{\boldsymbol{n}}$
$\hat{p}$
$\hat{u}_1$
$\hat{u}_2$
$\mathcal{L}_{\boldsymbol{n}}(\boldsymbol{x};\theta_{\boldsymbol{n}})$
Minimize
θ^*

FIGURE 1
The training procedure of the classical PINNs using unified activation functions for a 2D Biot's model.

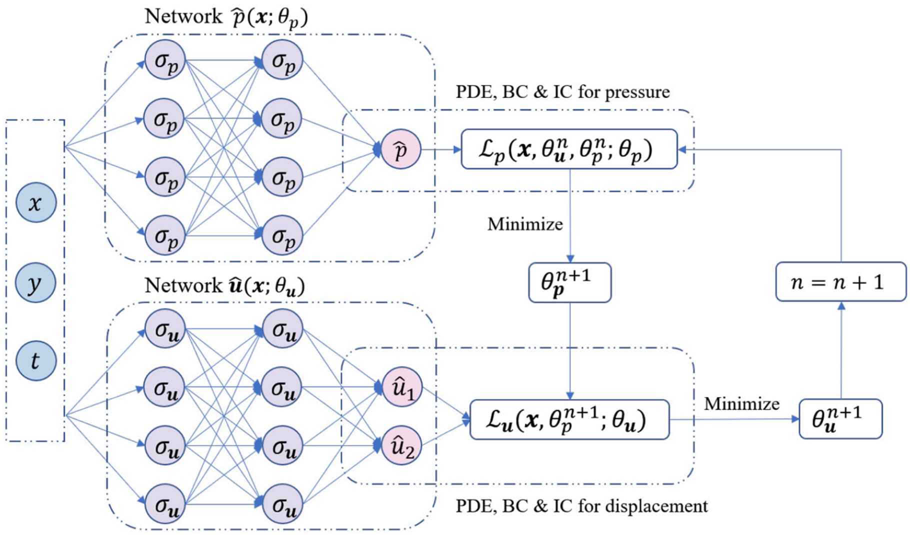
Network $\hat{p}(\boldsymbol{x};\theta_p)$
σ_p
PDE, BC & IC for pressure
$\hat{p}$
$\mathcal{L}_p(\boldsymbol{x},\theta_{\boldsymbol{u}}^n,\theta_p^n;\theta_p)$
Minimize
$\theta_{\boldsymbol{p}}^{n+1}$
$n=n+1$
x
y
t
Network $\hat{\boldsymbol{u}}(\boldsymbol{x};\theta_{\boldsymbol{u}})$
$\sigma_{\boldsymbol{u}}$
$\hat{u}_1$
$\hat{u}_2$
$\mathcal{L}_{\boldsymbol{u}}(\boldsymbol{x},\theta_p^{n+1};\theta_{\boldsymbol{u}})$
Minimize
$\theta_{\boldsymbol{u}}^{n+1}$
PDE, BC & IC for displacement

FIGURE 2
The training procedure of the proposed FS-PINNs. Two independent networks are trained iteratively with respect to the corresponding loss functions.

displacement and fluid pressure separately. We can then train our FS-PINNs iteratively with respect to their corresponding loss functions. Figure 2 illustrates the training procedure.

Similar to the classical PINNs, the functional forms of the two independent PINNs are as follows.

$$\hat{p}(\boldsymbol{x};\theta_p) = \Sigma_p^{L_p} \circ \Sigma_p^{L_p-1} \circ \cdots \circ \Sigma_p^1(\boldsymbol{x}), \tag{19}$$

$$\hat{\boldsymbol{u}}(\boldsymbol{x};\theta_{\boldsymbol{u}}) = \Sigma_{\boldsymbol{u}}^{L_{\boldsymbol{u}}} \circ \Sigma_{\boldsymbol{u}}^{L_{\boldsymbol{u}}-1} \circ \cdots \circ \Sigma_{\boldsymbol{u}}^1(\boldsymbol{x}). \tag{20}$$

Here, $\hat{p}$ refers to the network approximation of pressure p with L_p hidden layers, while $\hat{\boldsymbol{u}}$ refers to the network approximation of displacement $\boldsymbol{u}$ with $L_{\boldsymbol{u}}$ hidden layers. The collection of parameters of $\hat{p}$ and $\hat{\boldsymbol{u}}$ are denoted by θ_p and $\theta_{\boldsymbol{u}}$, respectively, and are given by:

$$\theta_p = \{\boldsymbol{W}_p^1, \boldsymbol{W}_p^2, \cdots, \boldsymbol{W}_p^{L_p}, \boldsymbol{b}_p^1, \boldsymbol{b}_p^2, \cdots, \boldsymbol{b}_p^{L_p}\}, \tag{21}$$

$$\theta_{\boldsymbol{u}} = \{\boldsymbol{W}_{\boldsymbol{u}}^1, \boldsymbol{W}_{\boldsymbol{u}}^2, \cdots, \boldsymbol{W}_{\boldsymbol{u}}^{L_{\boldsymbol{u}}}, \boldsymbol{b}_{\boldsymbol{u}}^1, \boldsymbol{b}_{\boldsymbol{u}}^2, \cdots, \boldsymbol{b}_{\boldsymbol{u}}^{L_{\boldsymbol{u}}}\}. \tag{22}$$

The mapping functions Σ_p^i and $\Sigma_{\boldsymbol{u}}^i$ are defined as

$$\boldsymbol{x}_p^i := \Sigma_p^i(\boldsymbol{x}_p^{i-1}) = \sigma_p^i(\boldsymbol{W}_p^i \boldsymbol{x}_p^{i-1} + \boldsymbol{b}_p^i), \quad i = 1, 2, \cdots, L_p, \tag{23}$$

$$\boldsymbol{x}_{\boldsymbol{u}}^i := \Sigma_{\boldsymbol{u}}^i(\boldsymbol{x}_{\boldsymbol{u}}^{i-1}) = \sigma_{\boldsymbol{u}}^i(\boldsymbol{W}_{\boldsymbol{u}}^i \boldsymbol{x}_{\boldsymbol{u}}^{i-1} + \boldsymbol{b}_{\boldsymbol{u}}^i), \quad i = 1, 2, \cdots, L_{\boldsymbol{u}}. \tag{24}$$

In this context, $(\boldsymbol{W}_p^i, \boldsymbol{b}_p^i, \sigma_p^i)$ and $(\boldsymbol{W}_{\boldsymbol{u}}^i, \boldsymbol{b}_{\boldsymbol{u}}^i, \sigma_{\boldsymbol{u}}^i)$ represent the weights, biases, and activation functions of the i-th layer for $\hat{p}$ and $\hat{\boldsymbol{u}}$, respectively. It is also assumed that the number of neurons in each hidden layer of $\hat{p}$ and $\hat{\boldsymbol{u}}$ is set to N_p and $N_{\boldsymbol{u}}$, respectively. Furthermore, all activation functions σ_p^i and $\sigma_{\boldsymbol{u}}^i$ are assumed to be identical.

From now, our attention turns to developing the loss functions. By utilizing the fixed-stress splitting method detailed in Equations (6)–(8) and Equations (9)–(11), we can impose constraints on the networks $\hat{p}$ and $\hat{\boldsymbol{u}}$, respectively. We begin by setting $n = 0$ and $\theta_p^n = \theta_{\boldsymbol{u}}^n = \boldsymbol{0}$. We can then formulate the optimization problem as follows. In the first step, we obtain θ_p^{n+1} by solving for the minimum of $\mathcal{L}_p(\boldsymbol{x}, \theta_{\boldsymbol{u}}^n, \theta_p^n; \theta_p)$:

$$\theta_p^{n+1} = \underset{\theta_p}{\arg\min}\, \mathcal{L}_p(\boldsymbol{x}, \theta_{\boldsymbol{u}}^n, \theta_p^n; \theta_p). \tag{25}$$

In the second step, we obtain $\theta_{\boldsymbol{u}}^{n+1}$ by solving for the minimum of $\mathcal{L}_{\boldsymbol{u}}(\boldsymbol{x}, \theta_p^{n+1}; \theta_{\boldsymbol{u}})$:

$$\theta_{\boldsymbol{u}}^{n+1} = \underset{\theta_{\boldsymbol{u}}}{\arg\min}\, \mathcal{L}_{\boldsymbol{u}}(\boldsymbol{x}, \theta_p^{n+1}; \theta_{\boldsymbol{u}}). \tag{26}$$

To be clear, $\mathcal{L}_p$ and $\mathcal{L}_{\boldsymbol{u}}$ denote the loss functions for networks $\hat{p}$ and $\hat{\boldsymbol{u}}$, respectively. These loss functions are defined as follows:

$$\begin{aligned}\mathcal{L}_p(\boldsymbol{x}, \theta_{\boldsymbol{u}}^n, \theta_p^n; \theta_p) &= \frac{1}{N_3}\sum_{i=1}^{N_3}\left\|\hat{p}(\boldsymbol{x}_p^i; \theta_p) - p(\boldsymbol{x}_p^i)\right\|^2 \\ &+ \frac{1}{N_4}\sum_{i=1}^{N_4}\Bigg\|\left(\frac{1}{M}+\beta_{FS}\right)\partial_t\hat{p}(\boldsymbol{x}_f^i; \theta_p) - \nabla\cdot K\nabla\hat{p}(\boldsymbol{x}_f^i; \theta_p) \\ &\qquad + \alpha\nabla\cdot\partial_t\hat{\boldsymbol{u}}(\boldsymbol{x}_f^i; \theta_{\boldsymbol{u}}^n) - \beta_{FS}\partial_t\hat{p}(\boldsymbol{x}_f^i; \theta_p^n) - f(\boldsymbol{x}_f^i)\Bigg\|^2,\end{aligned} \tag{27}$$

$$\begin{aligned}\mathcal{L}_{\boldsymbol{u}}(\boldsymbol{x}, \theta_p^{n+1}; \theta_{\boldsymbol{u}}) &= \frac{1}{N_5}\sum_{i=1}^{N_5}\left\|\hat{\boldsymbol{u}}(\boldsymbol{x}_{\boldsymbol{u}}^i; \theta_{\boldsymbol{u}}) - \boldsymbol{u}(\boldsymbol{x}_{\boldsymbol{u}}^i)\right\|^2 \\ &+ \frac{1}{N_6}\sum_{i=1}^{N_6}\left\|-\nabla\cdot\sigma(\hat{\boldsymbol{u}}(\boldsymbol{x}_{\boldsymbol{g}}^i; \theta_{\boldsymbol{u}})) + \alpha\nabla\hat{p}(\boldsymbol{x}_{\boldsymbol{g}}^i; \theta_p^{n+1}) - \boldsymbol{g}(\boldsymbol{x}_{\boldsymbol{g}}^i)\right\|^2.\end{aligned} \tag{28}$$

In the above expressions, $\{\boldsymbol{x}_p^i\}_{i=1}^{N_3}$ and $\{\boldsymbol{x}_{\boldsymbol{u}}^i\}_{i=1}^{N_5}$ denote the initial and boundary training data on $p(\boldsymbol{x})$ and $\boldsymbol{u}(\boldsymbol{x})$, respectively. Additionally, $\{\boldsymbol{x}_f^i\}_{i=1}^{N_4}$ and $\{\boldsymbol{x}_{\boldsymbol{g}}^i\}_{i=1}^{N_6}$ represent the collocation points for $f(\boldsymbol{x})$ and $\boldsymbol{g}(\boldsymbol{x})$ in the domain, respectively. By repeatedly updating θ_p and $\theta_{\boldsymbol{u}}$ using (25) and (26), we can get the network approximations of Biot's model: $\hat{p}(\boldsymbol{x}; \theta_p)$ and $\hat{\boldsymbol{u}}(\boldsymbol{x}; \theta_{\boldsymbol{u}})$.

3.3. Analysis of the proposed FS-PINNs

The classical PINNs have shown great potential in solving PDEs, but quantifying their errors remains an open problem [26]. The accuracy of neural network approximation is influenced by various factors, such as network architecture, training data, etc. In this section, we assume that PINNs can solve the two subproblems within a certain error tolerance. We then employ an iterative strategy for the proposed FS-PINNs. It is crucial to consider the possibility of error accumulation and divergence since network approximations can not guarantee convergence of the fixed-stress iteration. To address this issue, we observe that the fixed-stress splitting method, which is essentially equivalent to the fixed-point iterative scheme, has been shown to mitigate error accumulation. Therefore, we present a mathematical analysis to demonstrate the robustness of our proposed FS-PINNs. Given an initial guess $(\hat{p}^0, \hat{\boldsymbol{u}}^0)$, we describe the procedures of (25) and (26) as follows.

Step 1: Given $\hat{p}^n$ and $\hat{\boldsymbol{u}}^n$, we employ (25) to obtain a network solution $\hat{p}^{n+1}$ for the following problem.

$$\left(\frac{1}{M}+\beta_{FS}\right)\partial_t p^{n+1} - \nabla\cdot K\nabla p^{n+1} = f - \alpha\nabla\cdot\partial_t\hat{\boldsymbol{u}}^n + \beta_{FS}\partial_t\hat{p}^n \text{ in } \Omega\times(0,T], \tag{29}$$

$$p^{n+1} = p_0 \qquad \text{in } \Omega\times\{0\}, \tag{30}$$

$$p^{n+1} = p_d \qquad \text{on } \partial\Omega\times(0,T]. \tag{31}$$

The corresponding loss function for $\hat{p}^{n+1}$ imposes the initial condition and the boundary condition in the mean square sense.

Step 2: using $\hat{p}^{n+1}$, we employ (26) to obtain a network solution $\hat{\boldsymbol{u}}^{n+1}$ for the following problem.

$$-\nabla\cdot\sigma(\boldsymbol{u}^{n+1}) = \boldsymbol{g} - \alpha\nabla\hat{p}^{n+1} \qquad \text{in } \Omega\times(0,T], \tag{32}$$

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}_0 \qquad \text{in } \Omega\times\{0\}, \tag{33}$$

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}_d \qquad \text{on } \partial\Omega\times(0,T]. \tag{34}$$

The corresponding loss function for $\hat{\boldsymbol{u}}^{n+1}$ imposes the initial condition and the boundary condition in the mean square sense.

To analyze the numerical errors $\hat{p}^{n+1} - p$ and $\hat{\boldsymbol{u}}^{n+1} - \boldsymbol{u}$, we can express them as the sum of two terms: $(\hat{p}^{n+1} - p^{n+1}) + (p^{n+1} - p)$ and $(\hat{\boldsymbol{u}}^{n+1} - \boldsymbol{u}^{n+1}) + (\boldsymbol{u}^{n+1} - \boldsymbol{u})$, which can be dealt with separately. We assume that the Physics-Informed Neural Networks (PINNs) used to solve Equations (29) and (32) can achieve a certain error tolerance, denoted by δ, for the approximations $\hat{p}^{n+1}$ and $\hat{\boldsymbol{u}}^{n+1}$ obtained in each step. This is expressed as

$$\|\hat{p}^{n+1} - p^{n+1}\|_{H^1([0,T];H^1(\Omega))} \le \delta, \quad \|\hat{\boldsymbol{u}}^{n+1} - \boldsymbol{u}^{n+1}\|_{H^1([0,T];\boldsymbol{H}^1(\Omega))} \le \delta. \tag{35}$$

Such assumptions are reasonable if the neural networks are sufficiently deep and wide [33]. For estimating the terms $(p^{n+1} - p)$ and $(\boldsymbol{u}^{n+1} - \boldsymbol{u})$, we utilize integration by parts and employ the fixed-stress iterative method to analyze convergence. We then build on Theorem 2.1 to perform mathematical analysis.

Theorem 3.1. Assume that $(p, \boldsymbol{u}) \in W \times \boldsymbol{V}$ is the unique solution of the problem (1)–(5). Let $(\hat{p}^{n+1}, \hat{\boldsymbol{u}}^{n+1})$ be the network approximations of Equations (29)–(34), and $(p^{n+1}, \boldsymbol{u}^{n+1}) \in W \times \boldsymbol{V}$ be the exact solution of (6)–(11). We define $e_{\boldsymbol{u}}^{n+1} := \boldsymbol{u}^{n+1} - \boldsymbol{u}$ and $e_p^{n+1} := p^{n+1} - p$. Under the assumption (35) with a positive δ, if β_{FS} is large enough, then there exists a positive number $\tilde{L} < 1$ such that the following estimate holds for all $n \ge 0$.

$$\int_0^T \int_\Omega (\partial_t e_p^{n+1})^2 dxdt \le \tilde{L}\int_0^T \int_\Omega (\partial_t e_p^n)^2 dxdt + C\delta^2, \tag{36}$$

where C is a positive constant. Moreover, the limit superior exists and satisfies:

$$\limsup_{n\to\infty} \int_0^T \int_\Omega (\partial_t e_p^{n+1})^2 dxdt \le C\frac{\delta^2}{1-\tilde{L}}. \tag{37}$$

Proof. Firstly, we start with the second step of FS-PINNs. We subtract the original Equation (2) from the n-th iteration of (32). For the resulted equation, taking the derivative with respect to t, we obtain

$$-\nabla\cdot\sigma(\partial_t e_{\boldsymbol{u}}^n) = -\alpha\nabla\partial_t e_p^n + \alpha\nabla(\partial_t p^n - \partial_t\hat{p}^n). \tag{38}$$

Multiplying (38) by $\partial_t e_{\boldsymbol{u}}^n$, applying the divergence theorem, the Cauchy-Schwarz inequality, and the Poincaré inequality, we shall derive the following inequality with any $\epsilon_1 > 0$.

$$
\begin{aligned}
&2\mu \int_0^T \int_\Omega \varepsilon(\partial_t e_{\boldsymbol{u}}^n) : \varepsilon(\partial_t e_{\boldsymbol{u}}^n) dxdt + \lambda \int_0^T \int_\Omega (\nabla \cdot \partial_t e_{\boldsymbol{u}}^n)^2 dxdt \\
&= \alpha \int_0^T \int_\Omega (\nabla \cdot \partial_t e_{\boldsymbol{u}}^n)(\partial_t e_p^n) dxdt + \alpha \int_0^T \int_\Omega (\nabla(\partial_t \hat{p}^n - \partial_t p^n))(\partial_t e_{\boldsymbol{u}}^n) dxdt \\
&\le \frac{1}{2}\left(\frac{2\mu}{d} + \lambda\right) \int_0^T \int_\Omega (\nabla \cdot \partial_t e_{\boldsymbol{u}}^n)^2 dxdt + \frac{\alpha^2}{2\left(\frac{2\mu}{d} + \lambda\right)} \int_0^T \int_\Omega (\partial_t e_p^n)^2 dxdt \\
&+ \frac{\alpha^2}{4\epsilon_1} \int_0^T \int_\Omega (\nabla(\partial_t \hat{p}^n - \partial_t p^n))^2 dxdt + C_p \epsilon_1 \int_0^T \int_\Omega (\nabla \cdot \partial_t e_{\boldsymbol{u}}^n)^2 dxdt.
\end{aligned} \tag{39}
$$

On the other hand, there holds [23]

$$
\left(\frac{2\mu}{d} + \lambda\right) \int_0^T \int_\Omega (\nabla \cdot \partial_t e_{\boldsymbol{u}}^n)^2 dxdt \le 2\mu
\int_0^T \int_\Omega \varepsilon(\partial_t e_{\boldsymbol{u}}^n) : \varepsilon(\partial_t e_{\boldsymbol{u}}^n) dxdt + \lambda \int_0^T \int_\Omega (\nabla \cdot \partial_t e_{\boldsymbol{u}}^n)^2 dxdt.
$$

Based on the above inequality and (35), we can deduce the following inequality from (39).

$$
\begin{aligned}
&2\mu \int_0^T \int_\Omega \varepsilon(\partial_t e_{\boldsymbol{u}}^n) : \varepsilon(\partial_t e_{\boldsymbol{u}}^n) dxdt + \lambda \int_0^T \int_\Omega (\nabla \cdot \partial_t e_{\boldsymbol{u}}^n)^2 dxdt \\
&\le (1 + C\epsilon_1) \left[\frac{\alpha^2}{\frac{2\mu}{d} + \lambda} \int_0^T \int_\Omega (\partial_t e_p^n)^2 dxdt + \frac{\alpha^2}{2\epsilon_1} \delta^2 \right].
\end{aligned} \tag{40}
$$

Similar to the derivation of (38), subtracting the $(n+1)$-th iteration Equation (32) from (2) and taking the derivative with respect to t, we obtain

$$
-\nabla \cdot \sigma(\partial_t e_{\boldsymbol{u}}^{n+1}) = -\alpha \nabla \partial_t e_p^{n+1} + \alpha \nabla(\partial_t p^{n+1} - \partial_t \hat{p}^{n+1}). \tag{41}
$$

Subtracting (38) from (41) yields

$$
\begin{aligned}
-\nabla \cdot \sigma(\partial_t e_{\boldsymbol{u}}^{n+1} - \partial_t e_{\boldsymbol{u}}^n) &= -\alpha \nabla(\partial_t e_p^{n+1} - \partial_t e_p^n) + \\
&\alpha \nabla \left[(\partial_t p^{n+1} - \partial_t \hat{p}^{n+1}) - (\partial_t p^n - \partial_t \hat{p}^n) \right].
\end{aligned} \tag{42}
$$

We multiply (42) by $\partial_t e_{\boldsymbol{u}}^{n+1} - \partial_t e_{\boldsymbol{u}}^n$, and repeat the same argument as that for (40) to derive

$$
\begin{aligned}
&2\mu \int_0^T \int_\Omega \varepsilon(\partial_t e_{\boldsymbol{u}}^{n+1} - \partial_t e_{\boldsymbol{u}}^n) : \varepsilon(\partial_t e_{\boldsymbol{u}}^{n+1} - \partial_t e_{\boldsymbol{u}}^n) dxdt \\
&+ \lambda \int_0^T \int_\Omega (\nabla \cdot (\partial_t e_{\boldsymbol{u}}^{n+1} - \partial_t e_{\boldsymbol{u}}^n))^2 dxdt \\
&\le (1 + C\epsilon_1) \left[\frac{\alpha^2}{\frac{2\mu}{d} + \lambda} \int_0^T \int_\Omega (\partial_t e_p^{n+1} - \partial_t e_p^n)^2 dxdt + \frac{2\alpha^2}{\epsilon_1} \delta^2 \right].
\end{aligned} \tag{43}
$$

Next, we consider the entire system of the FS-PINNs. Subtracting the original Equation (1) from the $(n+1)$-th iteration Equation (29) yields

$$
\begin{aligned}
\left(\frac{1}{M} + \beta_{FS}\right) \partial_t e_p^{n+1} - \nabla \cdot K \nabla e_p^{n+1} &= -\alpha \nabla \cdot \partial_t e_{\boldsymbol{u}}^n \\
&+ \beta_{FS} \partial_t e_p^n + \alpha \nabla \cdot (\partial_t \boldsymbol{u}^n - \partial_t \hat{\boldsymbol{u}}^n) - \beta_{FS}(\partial_t p^n - \partial_t \hat{p}^n).
\end{aligned} \tag{44}
$$

Multiplying (44) by $\partial_t e_p^{n+1}$, and (41) by $\partial_t e_{\boldsymbol{u}}^n$, respectively, we then take a summation to derive the following equation.

$$
\begin{aligned}
\text{LHS} &:= \frac{1}{M} \int_0^T \int_\Omega (\partial_t e_p^{n+1})^2 dxdt + \frac{K}{2} \int_\Omega (\nabla e_p^{n+1}(T))^2 dx + \\
&2\mu \int_0^T \int_\Omega \varepsilon(\partial_t e_{\boldsymbol{u}}^{n+1}) : \varepsilon(\partial_t e_{\boldsymbol{u}}^n) dxdt \\
&+ \lambda \int_0^T \int_\Omega (\nabla \cdot \partial_t e_{\boldsymbol{u}}^{n+1})(\nabla \cdot \partial_t e_{\boldsymbol{u}}^n) dxdt + \\
&\beta_{FS} \int_0^T \int_\Omega (\partial_t e_p^{n+1})(\partial_t e_p^{n+1} - \partial_t e_p^n) dxdt \\
&= -\alpha \int_0^T \int_\Omega \left(\nabla \cdot (\partial_t \hat{\boldsymbol{u}}^n - \partial_t \boldsymbol{u}^n)\right)(\partial_t e_p^{n+1}) dxdt + \\
&\beta_{FS} \int_0^T \int_\Omega (\partial_t \hat{p}^n - \partial_t p^n)(\partial_t e_p^{n+1}) dxdt \\
&+ \alpha \int_0^T \int_\Omega (\nabla(\partial_t \hat{p}^{n+1} - \partial_t p^{n+1}))(\partial_t e_{\boldsymbol{u}}^n) dxdt =: \text{RHS}.
\end{aligned} \tag{45}
$$

We recall the following algebraic identities ["$(\cdot,\cdot)$" means the l^2 inner product].

$$
\begin{aligned}
(\xi - \eta, \xi) &= \frac{1}{2}\Big[(\xi, \xi) - (\eta, \eta) + (\xi - \eta, \xi - \eta) \Big], \\
(\xi, \eta) &= \frac{1}{4}\Big[(\xi + \eta, \xi + \eta) - (\xi - \eta, \xi - \eta) \Big].
\end{aligned} \tag{46}
$$

Applying (46) to the LHS of (45), and discarding some positive terms, we shall obtain

$$
\begin{aligned}
\text{LHS} \ge &\left(\frac{1}{M} + \frac{\beta_{FS}}{2}\right) \int_0^T \int_\Omega (\partial_t e_p^{n+1})^2 dxdt - \\
&\frac{\beta_{FS}}{2} \int_0^T \int_\Omega (\partial_t e_p^n)^2 dxdt \\
&- \frac{\mu}{2} \int_0^T \int_\Omega \varepsilon(\partial_t e_{\boldsymbol{u}}^{n+1} - \partial_t e_{\boldsymbol{u}}^n) : \varepsilon(\partial_t e_{\boldsymbol{u}}^{n+1} - \partial_t e_{\boldsymbol{u}}^n) dxdt \\
&- \frac{\lambda}{4} \int_0^T \int_\Omega (\nabla \cdot (\partial_t e_{\boldsymbol{u}}^{n+1} - \partial_t e_{\boldsymbol{u}}^n))^2 dxdt + \\
&\frac{\beta_{FS}}{2} \int_0^T \int_\Omega (\partial_t e_p^{n+1} - \partial_t e_p^n)^2 dxdt.
\end{aligned} \tag{47}
$$

We then deduce the following inequality based on (43).

$$
\begin{aligned}
\text{LHS} \ge &\left(\frac{1}{M} + \frac{\beta_{FS}}{2}\right) \int_0^T \int_\Omega (\partial_t e_p^{n+1})^2 dxdt - \\
&\frac{\beta_{FS}}{2} \int_0^T \int_\Omega (\partial_t e_p^n)^2 dxdt \\
&+ \left(\frac{\beta_{FS}}{2} - \frac{1 + C\epsilon_1}{2} \frac{\alpha^2}{\frac{2\mu}{d} + \lambda} \right) \int_0^T \int_\Omega (\partial_t e_p^{n+1} - \partial_t e_p^n)^2 dxdt \\
&- (1 + C\epsilon_1) \frac{\alpha^2}{\epsilon_1} \delta^2.
\end{aligned} \tag{48}
$$

For estimating an upper bound of the RHS term in (45), we use the Cauchy-Schwarz inequality with any $\epsilon_2 > 0$, and (40). We see that

$$\begin{aligned}\text{RHS} \le\ & \epsilon_2 \int_0^T \int_\Omega (\partial_t e_p^{n+1})^2 dxdt + \\ & C_p\epsilon_2 \int_0^T \int_\Omega (\nabla \cdot \partial_t e_{\boldsymbol{u}}^n)^2 dxdt + \frac{2\alpha^2 + \beta_{FS}}{4\epsilon_2}\delta^2 \\ \le\ & \epsilon_2 \int_0^T \int_\Omega (\partial_t e_p^{n+1})^2 dxdt + C_p\epsilon_2(1+C\epsilon_1)\frac{\alpha^2}{\frac{2\mu}{d}+\lambda}\int_0^T\int_\Omega \\ & (\partial_t e_p^n)^2 dxdt \\ & + \left(\frac{2\alpha^2+\beta_{FS}}{4\epsilon_2} + C_p\epsilon_2(1+C\epsilon_1)\frac{\alpha^2}{2\epsilon_1}\right)\delta^2. \qquad (49)\end{aligned}$$

We let $\beta_{FS} \ge (1+C\epsilon_1)\frac{\alpha^2}{\frac{2\mu}{d}+\lambda}$, by combining (48) and (49), we see that

$$\begin{aligned}&\left(\frac{1}{M} + \frac{\beta_{FS}}{2} - \epsilon_2\right)\int_0^T\int_\Omega (\partial_t e_p^{n+1})^2 dxdt \\ &\le \left(\frac{\beta_{FS}}{2} + C_1(\epsilon_1,\epsilon_2)\right)\int_0^T\int_\Omega (\partial_t e_p^n)^2 dxdt + C_2(\epsilon_1,\epsilon_2)\delta^2, \quad (50)\end{aligned}$$

where

$$\begin{aligned}C_1(\epsilon_1,\epsilon_2) &= C_p\epsilon_2(1+C\epsilon_1)\frac{\alpha^2}{\frac{2\mu}{d}+\lambda}, \\ C_2(\epsilon_1,\epsilon_2) &= \frac{2\alpha^2+\beta_{FS}}{4\epsilon_2} + \left(\frac{C_p\epsilon_2}{2}+1\right)(1+C\epsilon_1)\frac{\alpha^2}{\epsilon_1}.\end{aligned}$$

We choose ϵ_1 and ϵ_2 small enough, such that the contraction coefficient $\tilde{L}$, defined as follows, is strictly smaller than 1.

$$\tilde{L} := \frac{\beta_{FS} + 2C_1(\epsilon_1,\epsilon_2)}{\frac{2}{M} + \beta_{FS} - 2\epsilon_2} < 1.$$

Then, we can rewrite (50) in the following form.

$$\int_0^T\int_\Omega (\partial_t e_p^{n+1})^2 dxdt \le \tilde{L}\int_0^T\int_\Omega (\partial_t e_p^n)^2 dxdt + C\delta^2. \qquad (51)$$

This leads to the limit supremum.

$$\begin{aligned}&\limsup_{n\to\infty}\int_0^T\int_\Omega (\partial_t e_p^{n+1})^2 dxdt \\ &\le \limsup_{n\to\infty}\left[\tilde{L}^{n+1}\int_0^T\int_\Omega (\partial_t e_p^0)^2 dxdt + (\tilde{L}^n + \cdots + 1)C\delta^2\right] \\ &\le C\frac{\delta^2}{1-\tilde{L}}. \qquad (52)\end{aligned}$$

This completes our proof.

4. Benchmark tests

In this section, the efficacy of the proposed FS-PINNs is presented through three case studies. The first study involves a pure Dirichlet boundary value problem, while the second study considers a problem with mixed Dirichlet-Neumann boundary conditions. Finally, the third study investigates Barry-Mercer's problem. All experiments were implemented in the open-source machine learning tool PyTorch [34] on a single NVIDIA GeForce RTX 3060 GPU. Our code is available at https://github.com/newpolarbear/FS-PINNs. We evaluate the computational efficiency of our proposed method by comparing its true errors and runtime overheads with those of classical PINNs.

4.1. Models generation

In practice, the training procedure of classical PINNs is heavily influenced by several factors. One of these factors is the design of the neural network architecture, which does not have a universal standard. It's well-known that the network's representational capability is affected by its depth and width. While deeper and wider networks have greater representation capacity, they may suffer from slower training, compromises can be made to reduce runtime overhead. Another factor is the choice of nonlinear activation functions and optimization methods. In this work, we used $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ as the activation function and Adam optimizer with a fixed learning rate of 0.0001. In addition, some factors are specific to the problem being solved, such as the number of epochs, collocation points, and initial and boundary training data. In this study, we randomly select 10, 000 points as collocation points within the domain. The initial training data and each boundary training data are set to 500 and 200, respectively.

In addition to the standard configurations of classical PINNs, there are other aspects that come into play when training FS-PINNs. One FS-PINN is composed of two separate networks, which directly impact the performance. Moreover, training FS-PINNs involves networks optimization for each fixed-stress iteration, as shown in Figure 2. In the n-th fixed-stress iteration, we carry out the optimization procedure of network parameters θ_p and $\theta_{\boldsymbol{u}}$. For simplicity, we assume that the number of epochs for optimizing θ_p equals that for optimizing $\theta_{\boldsymbol{u}}$. Since the fixed-stress splitting method is an iterative algorithm, achieving accurate solutions at each step is not so necessary. For simplicity, we set the fixed-stress stabilization parameter to $\beta_{FS} = \alpha^2 / \left(\frac{2\mu}{d} + \lambda\right)$.

The training procedure of FS-PINNs is dynamic, loss functions $\mathcal{L}_p$ and $\mathcal{L}_{\boldsymbol{u}}$ can only show the performance in the current fixed-stress iteration. To evaluate the proposed method, we consider the true errors between network solutions and exact solutions as follows.

$$\begin{aligned}&\text{MSE for pressure } p: && \text{MSE}_p = \frac{1}{N}\sum_{i=1}^N \|\hat{p}(\boldsymbol{x}^i) - p(\boldsymbol{x}^i)\|^2, \\ &\text{MSE for } x\text{-displacement } u_1: && \text{MSE}_{u_1} = \frac{1}{N}\sum_{i=1}^N \|\hat{u}_1(\boldsymbol{x}^i) - u_1(\boldsymbol{x}^i)\|^2, \\ &\text{MSE for } y\text{-displacement } u_2: && \text{MSE}_{u_2} = \frac{1}{N}\sum_{i=1}^N \|\hat{u}_2(\boldsymbol{x}^i) - u_2(\boldsymbol{x}^i)\|^2.\end{aligned}$$

Here, $\hat{p}$, $\hat{u}_1$, and $\hat{u}_2$ represent the neural network solutions of pressure, x-displacement, and y-displacement, respectively, while p, u_1, and u_2 denote their corresponding exact solutions. The total mean squared error can be computed by taking the summation of

TABLE 1 Runtime overheads of five different parameter selections for FS-PINNs architecture of Example 1.

Parameter selection	Runtime for network $\hat{p}$	Runtime for network $\hat{\boldsymbol{u}}$	Total runtime
$N_p = 5 : N_{\boldsymbol{u}} = 25$	514.9 s (35.4%)	939.1 s (64.6%)	1454.0 s
$N_p = 10 : N_{\boldsymbol{u}} = 20$	471.4 s (40.8%)	685.4 s (59.2%)	1156.8 s
$N_p = 15 : N_{\boldsymbol{u}} = 15$	497.9 s (42.0%)	688.0 s (58.0%)	1185.9 s
$N_p = 20 : N_{\boldsymbol{u}} = 10$	483.0 s (43.1%)	637.0 s (56.9%)	1120.0 s
$N_p = 25 : N_{\boldsymbol{u}} = 5$	569.9 s (47.4%)	632.5 s (52.6%)	1202.4 s

these errors:

$$\text{Total MSE}: \qquad \text{MSE}_{total} = \text{MSE}_p + \text{MSE}_{u_1} + \text{MSE}_{u_2}. \quad (53)$$

4.2. Example 1

The first numerical experiment is based on the example in [16]. The domain is $\Omega = [0, 1]^2$ and the final time is $T = 0.5$, where pure Dirichlet boundary conditions are applied. In this experiment, the source or sink term f, body force $\boldsymbol{g}$, initial conditions, and boundary conditions are chosen such that the exact solutions are provided as follows:

$$\boldsymbol{u}(x, y, t) = \frac{t}{2}\left(x^2, y^2\right)^T, \quad p(x, y, t) = \sin\left(x + y\right) e^t. \qquad (54)$$

The model parameters are set to $E = 1.0$, $\nu = 0.3$, $\alpha = 1.0$, $M = 1.0$, and $K = 1.0$.

4.2.1. Robustness of the FS-PINNs architecture

The performance of FS-PINNs is highly dependent on its architecture. We fixed the total width to $N_p + N_{\boldsymbol{u}} = 30$ and depths to $L_p = L_{\boldsymbol{u}} = 3$, where N_p, $N_{\boldsymbol{u}}$ are the number of neurons, and L_p, $L_{\boldsymbol{u}}$ are the number of layers. We then designed an experiment with five sets of parameter selection for $N_p : N_{\boldsymbol{u}}$. To ensure that the networks are well-trained, we applied 20 fixed-stress iterations and 3,000 epochs for each fixed-stress step. The experiment is tested repeatedly to ensure the reliability of the results.

Table 1 shows the runtime overheads for various parameter selections, revealing that a more uniform separation exhibits lower runtime overhead. Notably, the case $N_p = 5 : N_{\boldsymbol{u}} = 25$ performs poorly compared to other cases, because wider networks experience slower training, which leads to a significant increase in the runtime overhead of network $\hat{u}$. Conversely, the other four cases exhibit similar performance, suggesting that the proposed FS-PINNs with reasonable architecture are resilient to runtime overhead.

Figure 3 tracks the trends of all MSE terms during different parameter selections training procedures, revealing that the case $N_p = 25 : N_{\boldsymbol{u}} = 5$ performs worst, and the case $N_p = 5 : N_{\boldsymbol{u}} = 25$ is less stable. The inadequate representation ability of tiny networks leads to inaccurate solutions. Network approximations $\hat{p}$ and $\hat{\boldsymbol{u}}$ depend on the interdependent loss functions for FS-PINNs. On the other hand, the trends of the remaining three cases are quite similar, which implies that FS-PINNs with reasonable architecture can withstand accuracy issues.

4.2.2. Acceleration effect of the FS-PINNs training

This section demonstrates the acceleration effect of the FS-PINNs over the classical PINNs. Biot's equations are solved using both methods with the same setup as in Example 1. The number of neurons for the two networks in FS-PINNs is fixed to $N_p = N_{\boldsymbol{u}} = 15$ (as indicated by Table 1), while the total number of neurons in classical PINNs is set to $N_{\boldsymbol{n}} = 30$ with the number of layers $L_{\boldsymbol{n}} = 3$. To examine the effect of the number of epochs optimizing networks and the number of fixed-stress iterations, we then conduct the test with a constant number of the multiplication of the above two number, which is set to 20, 000.

Figure 4 shows the trends of MSE terms for solving the problem in Example 1. We present the results generated by one classical PINN and four FS-PINNs under identical conditions, which demonstrates the acceleration effect of FS-PINNs. By decoupling the original optimization problem into two small sub-problems, together with the fixed-stress coefficient $L < 1$, FS-PINNs increase efficiency and maintain certain accuracy. From this perspective, FS-PINNs offer more flexibility in solving the problem through different setting of optimizing epoch and fixed-stress iteration. In Figure 4, we can see that a smaller number of epochs lead to faster convergence initially but lower accuracy eventually. Since the fixed-stress splitting method is a numerical approach, high accuracy is not essential for each fixed-stress step. It implies that a large number of optimizing epochs is not so helpful.

Among all the results generated by FS-PINNs, here we consider in-depth observations for the case of $[400, 50]$. Here, 400 means the number of epochs, and 50 means the number of fixed-stress iterations. To test the cases with a larger parameter λ, we also include the testing result when the Poisson ratio $\nu = 0.49$ for Example 1. Figure 5 provides a comparison between the analytical solution and the numerical solution given by FS-PINNs with $[400, 50]$. Theses results show that the proposed FS-PINNs with $\nu = 0.3$ and $\nu = 0.49$ accurately predict the pressure p and displacement $\boldsymbol{u}$ with errors within an acceptable tolerance range. However, it is true that when λ is large, to obtain a more accurate solution, we need to put a relatively large weight for the first equation in the total loss function. And also, we need to increase the number of epochs and set a finer learning rate for the training of the network for $\boldsymbol{u}$.

4.3. Example 2

The Biot's problem with mixed boundary conditions presented in [2] is tested. We consider the square domain $\Omega = [0, 1]^2$, with

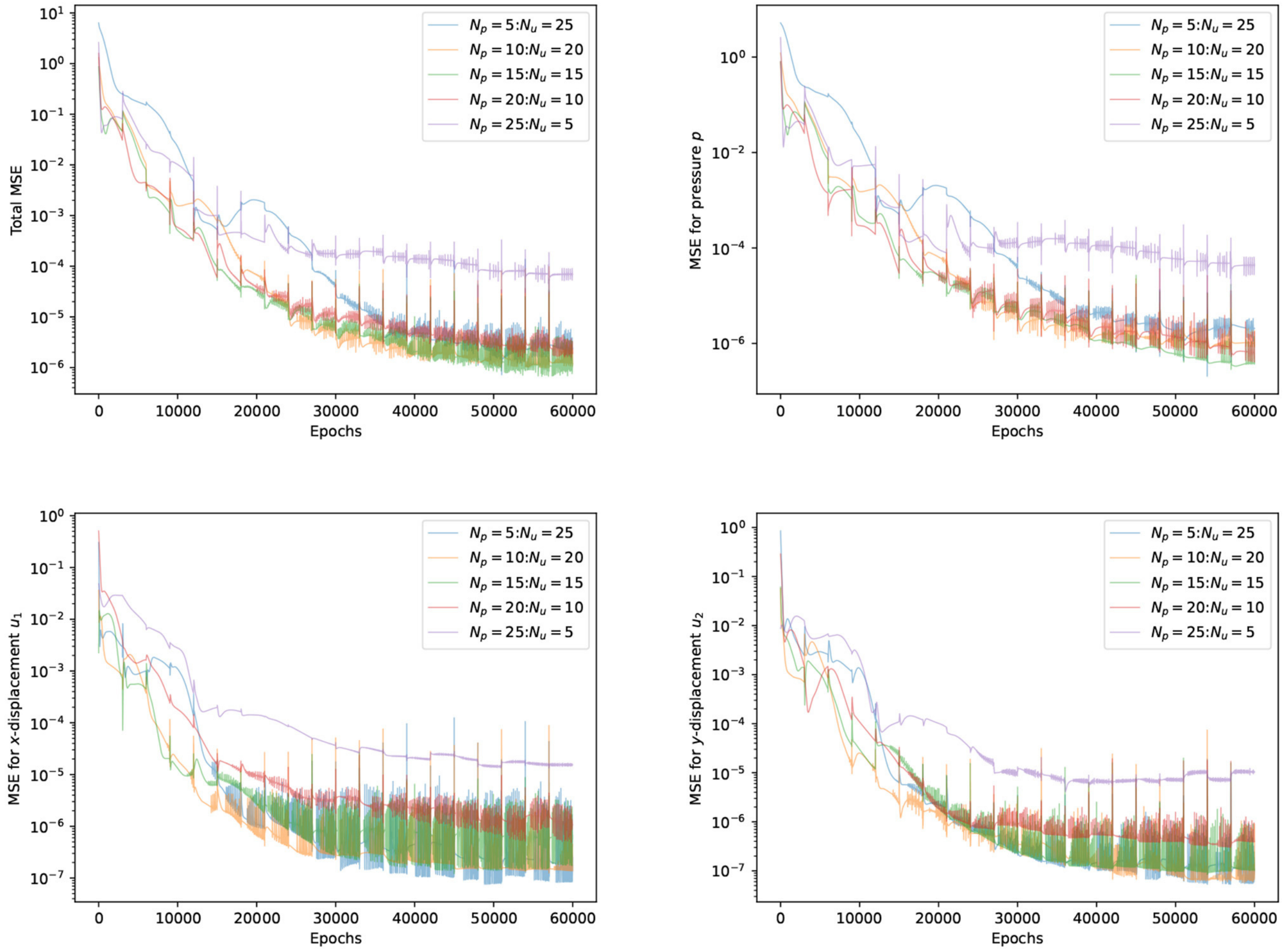

FIGURE 3
Trends of all the MSE terms at $T = 0.5$ for five different FS-PINNs architecture during training with respect to epoch.

Dirichlet boundary $\Gamma_1 \cup \Gamma_3$ and Neumann boundary $\Gamma_2 \cup \Gamma_4$. Specifically, $\Gamma_1 = (1, y); 0 \le y \le 1$, $\Gamma_2 = (x, 0); 0 \le x \le 1$, $\Gamma_3 = (0, y); 0 \le y \le 1$, and $\Gamma_4 = (x, 1); 0 \le x \le 1$. Setting the final time to $T = 0.2$, we choose the body force $\boldsymbol{g}$, the source or sink term f, as well as the initial and boundary conditions such that exact solutions are as follows.

$$u_1 = e^{-t}\left(\sin(2\pi y)(\cos(2\pi x) - 1) + \frac{1}{\mu+\lambda}\sin(\pi x)\sin(\pi y)\right),$$
$$u_2 = e^{-t}\left(\sin(2\pi x)(1 - \cos(2\pi y)) + \frac{1}{\mu+\lambda}\sin(\pi x)\sin(\pi y)\right),$$
$$p = e^{-t}\sin(\pi x)\sin(\pi y).$$

The parameters of the model are $E = 1.0$, $\nu = 0.3$, $\alpha = 1.0$, $M = 1.0$, $K = 1$.

In dealing with mixed boundary conditions, it is necessary to select suitable loss functions capable of handling both Dirichlet and Neumann boundary conditions. To this end, we follow the approach taken in [28, 32] and employ the corresponding loss functions $\mathcal{L}_p$ and $\mathcal{L}_{\boldsymbol{u}}$ for pressure and displacement, respectively. In this test, following the result in Section 4.2.2., we employ the FS-PINN with [400, 50] using the same architecture to solve Example 2.

In Figure 6, we compare the analytical solution with our FS-PINN solution and observe that our approach can approximate solutions for problems with mixed boundary conditions. However, its performance is not as good as that of solving the pure Dirichlet problem in Example 1. The network approximations in the boundary region are sub-optimal because all the parameters used were selected to solve Example 1. Hence, the complexity of Example 2 presents a challenge to our network, which could potentially be overcome by optimizing the parameters, such as by using additional collocation points in the boundary region.

4.4. Example 3

The Barry-Mercer's problem is a classical benchmark problem of porous media involving injection of fluids [14, 31, 35, 36]. It is a two-dimensional problem in the square domain $\Omega = [0, 1]^2$ that has a point source f located

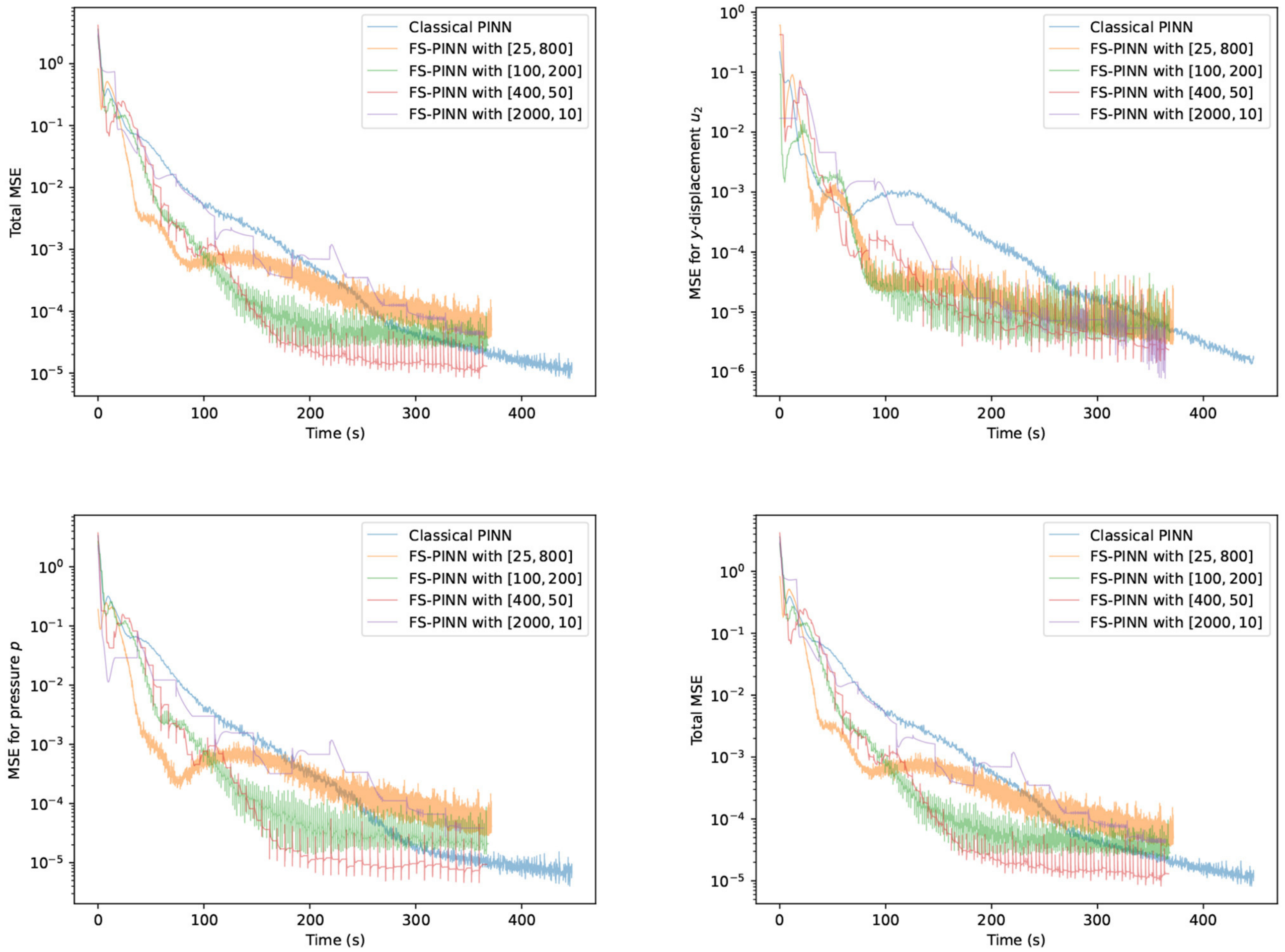

FIGURE 4
Trends of all the MSE terms at $T = 0.5$ for one classical PINN and four FS-PINNs with [the number of epochs training networks, the number of fixed-stress iterations] during training with respect to runtime overhead.

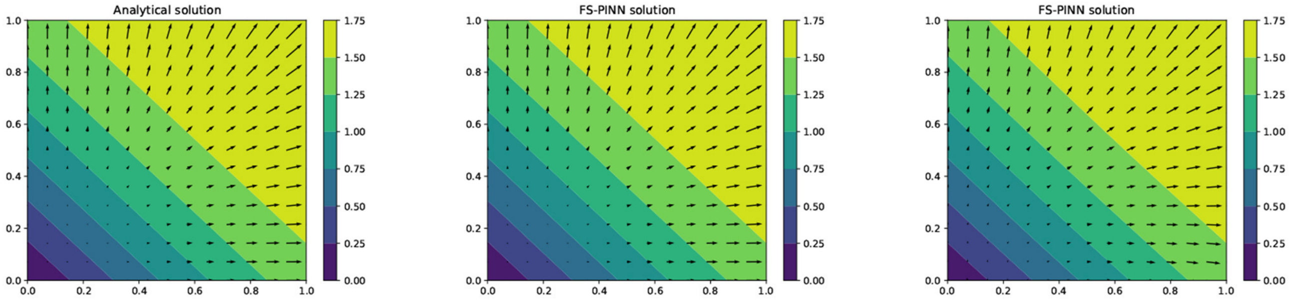

FIGURE 5
A comparison of the analytical solution **(left)**, the FS-PINN solution with $\nu = 0.3$ **(middle)**, and the FS-PINN solution $\nu = 0.49$ **(right)** at $T = 0.5$ for Example 1.

at $(x_0, y_0) = (0.25, 0.25)$, which makes it challenging to solve numerically. Figure 7 depicts the boundary conditions under consideration.

As for the initial conditions, we set

$$\boldsymbol{u} = 0, \quad p = 0 \quad \text{in } \Omega \times \{0\}. \tag{55}$$

To define the body force term $\boldsymbol{g}$ and the source/sink term f, we use the following expressions:

$$\boldsymbol{g} = \boldsymbol{0}, \quad f = 2\delta(x - x_0)\delta(y - y_0)\sin(t), \tag{56}$$

where δ denotes the Dirac function. We approximate the Dirac function using the Gaussian distribution [31], that is, $\delta(x) \approx$

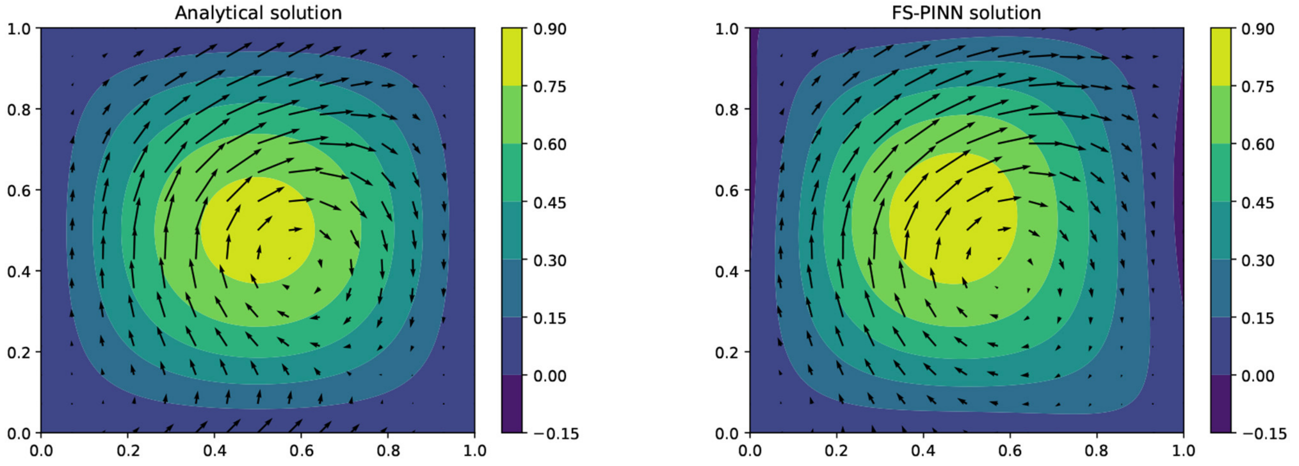

FIGURE 6
A comparison of the analytical solution **(left)** and the FS-PINN solution with [400, 50] **(right)** at $T = 0.2$ for Example 2.

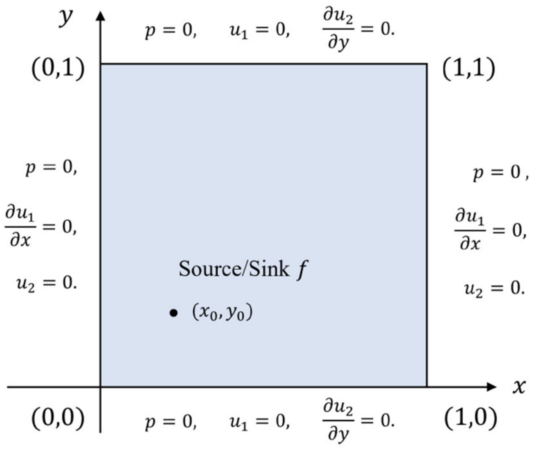
y
$p = 0, \quad u_1 = 0, \quad \frac{\partial u_2}{\partial y} = 0.$
(0,1)
(1,1)
$p = 0,$
$\frac{\partial u_1}{\partial x} = 0,$
$u_2 = 0.$
$p = 0\,,$
$\frac{\partial u_1}{\partial x} = 0,$
$u_2 = 0.$
Source/Sink f
(x_0, y_0)
x
(0,0)
$p = 0, \quad u_1 = 0, \quad \frac{\partial u_2}{\partial y} = 0.$
(1,0)

FIGURE 7
Boundary conditions for the Barry-Mercer's problem.

$\frac{1}{c_g\sqrt{\pi}}e^{-(x/c_g)^2}$ with $c_g = 0.04$. The physical parameters are specified as follows:

$$1/M = 0, \quad \alpha = 1.0, \quad \lambda = 0.2, \quad \mu = 0.4, \quad K = 1.0. \tag{57}$$

Denote $\lambda_n = n\pi$, $\lambda_q = q\pi$, and $\lambda_{nq} = \lambda_n^2 + \lambda_q^2$, then the analytical solution is as follows:

$$p(x,y,t) = -4\sum_{n=1}^{\infty}\sum_{q=1}^{\infty}\hat{p}(n,q,t)\sin(\lambda_n x)\sin(\lambda_q y),$$
$$u_1(x,y,t) = 4\sum_{n=1}^{\infty}\sum_{q=1}^{\infty}\hat{u}_1(n,q,t)\cos(\lambda_n x)\sin(\lambda_q y),$$
$$u_2(x,y,t) = 4\sum_{n=1}^{\infty}\sum_{q=1}^{\infty}\hat{u}_2(n,q,t)\sin(\lambda_n x)\cos(\lambda_q y),$$

where

$$\hat{p}(n,q,t) = -\frac{2\sin(\lambda_n x_0)\sin(\lambda_q y_0)}{\lambda_{nq}^2 + 1}\left(\lambda_{nq}\sin t - \cos t + e^{-\lambda_{nq}t}\right),$$
$$\hat{u}_1(n,q,t) = \frac{\lambda_n}{\lambda_{nq}}\hat{p}(n,q,t), \quad \hat{u}_2(n,q,t) = \frac{\lambda_q}{\lambda_{nq}}\hat{p}(n,q,t).$$

In this case, we use the same architecture as in Section 4.2.2, which is the FS-PINN with [400, 50], to solve Barry-Mercer's problem. As mentioned in the literature [37], networks training relies on the propagation of information from initial and boundary points to the interior points. To improve the performance of our FS-PINN, we apply weighted loss functions proposed in [32], and increase the weights of the initial and boundary loss terms. Specifically, we replace $\frac{1}{N_3}$, $\frac{1}{N_5}$ with $\frac{1000}{N_3}$, $\frac{100}{N_5}$ in the loss functions (27), (28), respectively. The solution obtained from our FS-PINN closely matches the analytic solution at $T = \pi/4$ for Example 3, as demonstrated in Figure 8. This indicates the effectiveness of our approach in approximating the analytic solution.

5. Conclusion

This paper presents a new approach to solving Biot's consolidation model by combining Physics-informed neural networks (PINNs) with the fixed-stress splitting method. We demonstrate that our method achieves faster convergence and lower computational costs than the neural network methods which solve Biot's problem in a monolithic way. We provide a theoretical analysis to show the convergence of our proposed FS-PINNs. The numerical experiments illustrate the effectiveness of our approach on various test cases with different boundary conditions and physical parameters.

While our approach has several advantages over some traditional methods for solving poroelastic models, such as being meshless, not requiring inf-sup stability, there are also some

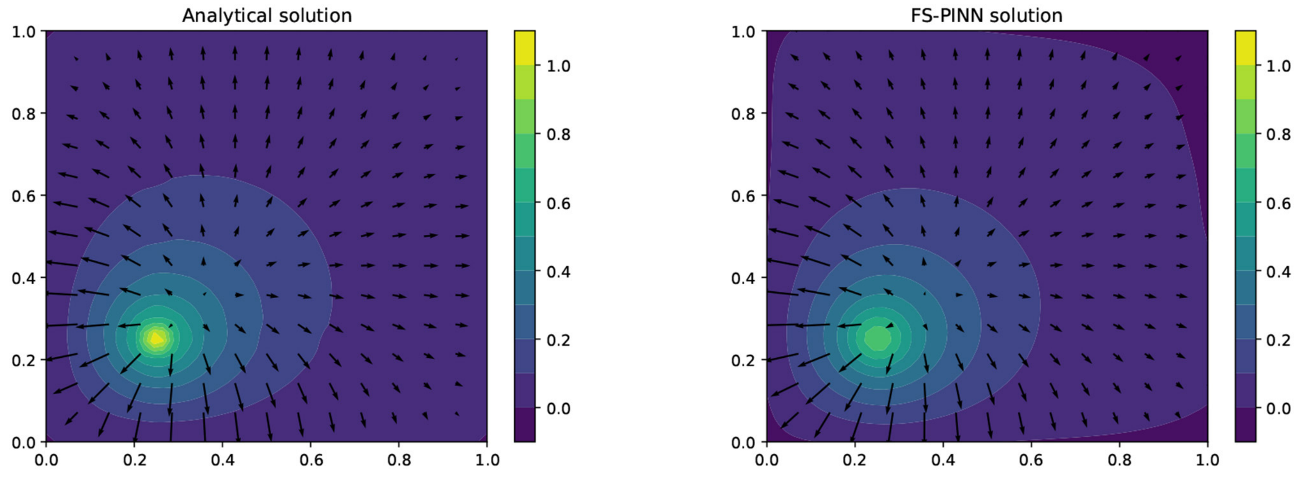

FIGURE 8
A comparison of the analytical solution **(left)** and the FS-PINN solution **(right)** at $T = \pi/4$ for Example 3.

limitations and challenges that need to be addressed in future work. These include choosing optimal network architectures and parameters for different problems, improving accuracy and reducing computational costs even further, and extending the approach to other PDE problems and multi-physics problems. We hope that our work can inspire more research on combining deep learning techniques with iterative methods for solving PDEs.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

MC: conceptualization, investigation, writing—original draft, validation, methodology, writing—review and editing, supervision, formal analysis, and funding acquisition. HG: investigation, writing—original draft, validation, visualization, writing—review and editing, formal analysis, and data curation. PH: investigation, writing—original draft, validation, visualization, writing—review and editing, software, and data curation. JL: conceptualization, investigation, funding acquisition, validation, methodology, project administration, and supervision. All authors contributed to the article and approved the submitted version.

Funding

The work of MC was partially supported by the NIH-RCMI grant through 347 U54MD013376, the affiliated project award from the Center for Equitable Artificial Intelligence and Machine Learning Systems (CEAMLS) at Morgan State University (project ID 02232301), and the National Science Foundation awards (1831950 and 2228010). The work of HG, PH, and JL were partially supported by the NSF of China No. 11971221, Guangdong NSF Major Fund No. 2021ZDZX1001, the Shenzhen Sci-Tech Fund Nos. RCJC20200714114556020, JCYJ20200109115422828, and JCYJ20190809150413261, National Center for Applied Mathematics Shenzhen, and SUSTech International Center for Mathematics.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

1. Biot MA. General theory of three-dimensional consolidation. *J Appl Phys*. (1941) 12:155–64. doi: 10.1063/1.1712886

2. Ju G, Cai M, Li J, Tian J. Parameter-robust multiphysics algorithms for Biot model with application in brain edema simulation. *Math Comput Simul*. (2020) 177:385–403. doi: 10.1016/j.matcom.2020.04.027

3. Kim J, Tchelepi HA, Juanes R. Stability and convergence of sequential methods for coupled flow and geomechanics: fixed-stress and fixed-strain splits. *Comput Methods Appl Mech Eng*. (2011) 200:1591–606. doi: 10.1016/j.cma.2010.12.022

4. Ženíšek A. The existence and uniqueness theorem in Biot's consolidation theory. *Aplik Matem*. (1984) 29:194–211. doi: 10.21136/AM.1984.104085

5. Showalter RE. Diffusion in poro-elastic media. *J Math Anal Appl*. (2000) 251:310–40. doi: 10.1006/jmaa.2000.7048

6. Phillips PJ, Wheeler MF. A coupling of mixed and continuous Galerkin finite element methods for poroelasticity I: the continuous in time case. *Comput Geosci*. (2007) 11:131–44. doi: 10.1007/s10596-007-9045-y

7. Nordbotten JM. Stable cell-centered finite volume discretization for Biot equations. *SIAM J Numer Anal*. (2016) 54:942–68. doi: 10.1137/15M1014280

8. Coulet J, Faille I, Girault V, Guy N, Nataf F. A fully coupled scheme using virtual element method and finite volume for poroelasticity. *Comput Geosci*. (2020) 24:381–403. doi: 10.1007/s10596-019-09831-w

9. Yi SY. A coupling of nonconforming and mixed finite element methods for Biot's consolidation model. *Numer Methods Part Diff Equat*. (2013) 29:1749–77. doi: 10.1002/num.21775

10. Yi SY. A study of two modes of locking in poroelasticity. *SIAM J Numer Anal*. (2017) 55:1915–36. doi: 10.1137/16M1056109

11. Rodrigo C, Gaspar F, Hu X, Zikatanov L. Stability and monotonicity for some discretizations of the Biot's consolidation model. *Comput Methods Appl Mech Eng*. (2016) 298:183–204. doi: 10.1016/j.cma.2015.09.019

12. Lee JJ, Mardal KA, Winther R. Parameter-robust discretization and preconditioning of Biot's consolidation model. *SIAM J Sci Comput*. (2017) 39:A1–24. doi: 10.1137/15M1029473

13. Cai M, Zhang G. Comparisons of some iterative algorithms for Biot equations. *Int J Evol Equat*. (2017) 10:267–82.

14. Phillips PJ, Wheeler MF. Overcoming the problem of locking in linear elasticity and poroelasticity: an heuristic approach. *Comput Geosci*. (2009) 13:5–12. doi: 10.1007/s10596-008-9114-x

15. Rodrigo C, Hu X, Ohm P, Adler JH, Gaspar FJ, Zikatanov L. New stabilized discretizations for poroelasticity and the Stoke's equations. *Comput Methods Appl Mech Eng*. (2018) 341:467–84. doi: 10.1016/j.cma.2018.07.003

16. Feng X, Ge Z, Li Y. Analysis of a multiphysics finite element method for a poroelasticity model. *IMA J Numer Anal*. (2018) 38:330–59. doi: 10.1093/imanum/drx003

17. Oyarzúa R, Ruiz-Baier R. Locking-free finite element methods for poroelasticity. *SIAM Journal on Numerical Anal*. (2016) 54:2951–73. doi: 10.1137/15M1050082

18. Yi SY, Bean ML. Iteratively coupled solution strategies for a four-field mixed finite element method for poroelasticity. *Int J Numer Analyt Methods Geomech*. (2017) 41:159–79. doi: 10.1002/nag.2538

19. Mikelić A, Wheeler MF. Convergence of iterative coupling for coupled flow and geomechanics. *Comput Geosci*. (2013) 17:455–61. doi: 10.1007/s10596-012-9318-y

20. Mikelić A, Wang B, Wheeler MF. Numerical convergence study of iterative coupling for coupled flow and geomechanics. *Comput Geosci*. (2014) 18:325–41. doi: 10.1007/s10596-013-9393-8

21. Both JW, Borregales M, Nordbotten JM, Kumar K, Radu FA. Robust fixed stress splitting for Biot's equations in heterogeneous media. *Appl Math Lett*. (2017) 68:101–8. doi: 10.1016/j.aml.2016.12.019

22. Bause M, Radu FA, Köcher U. Space-time finite element approximation of the Biot poroelasticity system with iterative coupling. *Comput Methods Appl Mech Eng*. (2017) 320:745–68. doi: 10.1016/j.cma.2017.03.017

23. Storvik E, Both JW, Kumar K, Nordbotten JM, Radu FA. On the optimization of the fixed-stress splitting for Biot's equations. *Int J Numer Methods Eng*. (2019) 120:179–94. doi: 10.1002/nme.6130

24. Borregales M, Kumar K, Radu FA, Rodrigo C, Gaspar FJ. A partially parallel-in-time fixed-stress splitting method for Biot's consolidation model. *Comput Math Appl*. (2019) 77:1466–78. doi: 10.1016/j.camwa.2018.09.005

25. Raissi M, Perdikaris P, Karniadakis GE. Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J Comput Phys*. (2019) 378:686–707. doi: 10.1016/j.jcp.2018.10.045

26. Lu L, Meng X, Mao Z, Karniadakis GE. DeepXDE: a deep learning library for solving differential equations. *SIAM Rev*. (2021) 63:208–28. doi: 10.1137/19M1274067

27. Amini D, Haghighat E, Juanes R. Inverse modeling of nonisothermal multiphase poromechanics using physics-informed neural networks. *arXiv preprint arXiv:220903276*. (2022). doi: 10.1016/j.jcp.2023.112323

28. Kadeethum T, Jørgensen TM, Nick HM. Physics-informed neural networks for solving nonlinear diffusivity and Biot's equations. *PLoS ONE*. (2020) 15:e0232683. doi: 10.1371/journal.pone.0232683

29. Millevoi C, Spiezia N, Ferronato M. *On Physics-Informed Neural Networks Architecture for Coupled Hydro-Poromechanical Problems*. (2022). doi: 10.2139/ssrn.4074416

30. Bekele YW. Physics-informed deep learning for flow and deformation in poroelastic media. *arXiv preprint arXiv:201015426*. (2020).

31. Haghighat E, Amini D, Juanes R. Physics-informed neural network simulation of multiphase poroelasticity using stress-split sequential training. *Comput Methods Appl Mech Eng*. (2022) 397:115141. doi: 10.1016/j.cma.2022.115141

32. Amini D, Haghighat E, Juanes R. Physics-informed neural network solution of thermo-hydro-mechanical (THM) processes in porous media. *arXiv preprint arXiv:220301514*. (2022). doi: 10.1061/(ASCE)EM.1943-7889.0002156

33. De Ryck T, Jagtap AD, Mishra S. Error estimates for physics informed neural networks approximating the Navier-Stokes equations. *arXiv preprint arXiv:220309346*. (2022). doi: 10.1093/imanum/drac085

34. Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, et al. PyTorch: An imperative style, high-performance deep learning library. In: Wallach H, Larochelle H, Beygelzimer A, Alche-Buc F, Fox E, Garnett R, editors. *Advances in Neural Information Processing Systems, Vol. 32*. Curran Associates (2019). Available online at: https://proceedings.neurips.cc/paper_files/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf

35. Barry S, Mercer G. Exact solutions for two-dimensional time-dependent flow and deformation within a poroelastic medium. *J Appl Mech*. (1999) 66:536–40. doi: 10.1115/1.2791080

36. Phillips PJ. *Finite Element Methods in Linear Poroelasticity: Theoretical and Computational Results*. Austin, TX: The University of Texas at Austin (2005).

37. Daw A, Bu J, Wang S, Perdikaris P, Karpatne A. Mitigating propagation failures in pinns using evolutionary sampling, conference paper at ICLR 2023?. (Under review).

Appendix

Proof of Theorem 2.1

Proof. Subtracting (6), (9) from (1), (2), respectively, we shall derive

$$\left(\frac{1}{M}+\beta_{FS}\right)\partial_t e_p^{n+1}-\nabla\cdot K\nabla e_p^{n+1}=-\alpha\nabla\cdot\partial_t e_{\boldsymbol{u}}^n+\beta_{FS}\partial_t e_p^n, \tag{58}$$

$$-\nabla\cdot\sigma(e_{\boldsymbol{u}}^{n+1})+\alpha\nabla e_p^{n+1}=0. \tag{59}$$

After differentiating (59) with respect to t, one has

$$-\nabla\cdot\sigma(\partial_t e_{\boldsymbol{u}}^{n+1})+\alpha\nabla\partial_t e_p^{n+1}=0. \tag{60}$$

We multiply (58), (60) by $\partial_t e_p^{n+1}$, $\partial_t e_{\boldsymbol{u}}^n$, respectively, and use integration by parts to obtain

$$\begin{aligned}&\left(\frac{1}{M}+\beta_{FS}\right)\int_0^T\int_\Omega(\partial_t e_p^{n+1})^2dxdt+\\&\frac{K}{2}\int_\Omega(\nabla e_p^{n+1}(T))^2dx\\&=-\alpha\int_0^T\int_\Omega(\nabla\cdot\partial_t e_{\boldsymbol{u}}^n)(\partial_t e_p^{n+1})dxdt+\\&\beta_{FS}\int_0^T\int_\Omega(\partial_t e_p^n)(\partial_t e_p^{n+1})dxdt,\end{aligned} \tag{61}$$

$$\begin{aligned}&2\mu\int_0^T\int_\Omega\varepsilon(\partial_t e_{\boldsymbol{u}}^{n+1}):\varepsilon(\partial_t e_{\boldsymbol{u}}^n)dxdt+\\&\lambda\int_0^T\int_\Omega(\nabla\cdot\partial_t e_{\boldsymbol{u}}^{n+1})(\nabla\cdot\partial_t e_{\boldsymbol{u}}^n)dxdt\\&=\alpha\int_0^T\int_\Omega(\nabla\cdot\partial_t e_{\boldsymbol{u}}^n)(\partial_t e_p^{n+1})dxdt.\end{aligned} \tag{62}$$

Combining Equations (61) and (62) results in the following equation.

$$\begin{aligned}&\frac{1}{M}\int_0^T\int_\Omega(\partial_t e_p^{n+1})^2dxdt+\frac{K}{2}\int_\Omega(\nabla e_p^{n+1}(T))^2dx+\\&2\mu\int_0^T\int_\Omega\varepsilon(\partial_t e_{\boldsymbol{u}}^{n+1}):\varepsilon(\partial_t e_{\boldsymbol{u}}^n)dxdt\\&+\lambda\int_0^T\int_\Omega(\nabla\cdot\partial_t e_{\boldsymbol{u}}^{n+1})(\nabla\cdot\partial_t e_{\boldsymbol{u}}^n)dxdt+\\&\beta_{FS}\int_0^T\int_\Omega(\partial_t e_p^{n+1})(\partial_t e_p^{n+1}-\partial_t e_p^n)dxdt=0.\end{aligned} \tag{63}$$

Applying the algebraic identities (46) to (63), we derive

$$\begin{aligned}&\left(\frac{1}{M}+\frac{\beta_{FS}}{2}\right)\int_0^T\int_\Omega(\partial_t e_p^{n+1})^2dxdt+\frac{K}{2}\int_\Omega(\nabla e_p^{n+1}(T))^2dx\\&+\frac{\mu}{2}\int_0^T\int_\Omega\varepsilon(\partial_t e_{\boldsymbol{u}}^{n+1}+\partial_t e_{\boldsymbol{u}}^n):\varepsilon(\partial_t e_{\boldsymbol{u}}^{n+1}+\partial_t e_{\boldsymbol{u}}^n)dxdt\\&+\frac{\lambda}{4}\int_0^T\int_\Omega\left(\nabla\cdot(\partial_t e_{\boldsymbol{u}}^{n+1}+\partial_t e_{\boldsymbol{u}}^n)\right)^2dxdt++\\&\frac{\beta_{FS}}{2}\int_0^T\int_\Omega(\partial_t e_p^{n+1}-\partial_t e_p^n)^2dxdt\\&=\frac{\mu}{2}\int_0^T\int_\Omega\varepsilon(\partial_t e_{\boldsymbol{u}}^{n+1}-\partial_t e_{\boldsymbol{u}}^n):\varepsilon(\partial_t e_{\boldsymbol{u}}^{n+1}-\partial_t e_{\boldsymbol{u}}^n)dxdt\\&+\frac{\lambda}{4}\int_0^T\int_\Omega\left(\nabla\cdot(\partial_t e_{\boldsymbol{u}}^{n+1}-\partial_t e_{\boldsymbol{u}}^n)\right)^2dxdt+\\&\frac{\beta_{FS}}{2}\int_0^T\int_\Omega(\partial_t e_p^n)^2dxdt.\end{aligned} \tag{64}$$

The difference of two successive iterations based on (59) will yield

$$-\nabla\cdot\sigma(\partial_t e_{\boldsymbol{u}}^{n+1}-\partial_t e_{\boldsymbol{u}}^n)+\alpha\nabla(\partial_t e_p^{n+1}-\partial_t e_p^n)=0. \tag{65}$$

Multiplying (65) by $(\partial_t e_{\boldsymbol{u}}^{n+1}-\partial_t e_{\boldsymbol{u}}^n)$, integration by parts, and then applying Cauchy-Schwarz inequality, we derive that

$$\begin{aligned}&2\mu\int_0^T\int_\Omega\varepsilon(\partial_t e_{\boldsymbol{u}}^{n+1}-\partial_t e_{\boldsymbol{u}}^n):\varepsilon(\partial_t e_{\boldsymbol{u}}^{n+1}-\partial_t e_{\boldsymbol{u}}^n)dxdt\\&+\lambda\int_0^T\int_\Omega\left(\nabla\cdot(\partial_t e_{\boldsymbol{u}}^{n+1}-\partial_t e_{\boldsymbol{u}}^n)\right)^2dxdt+\\&\leq\frac{\alpha^2}{\frac{2\mu}{d}+\lambda}\int_0^T\int_\Omega(\partial_t e_p^{n+1}-\partial_t e_p^n)^2dxdt.\end{aligned} \tag{66}$$

Discarding the second, third, and fourth terms in (64), we choose $\beta_{FS}\geq\frac{\alpha^2}{2\left(\frac{2\mu}{d}+\lambda\right)}$ to obtain

$$\int_0^T\int_\Omega(\partial_t e_p^{n+1})^2dxdt\leq\frac{\beta_{FS}}{\beta_{FS}+\frac{2}{M}}\int_0^T\int_\Omega(\partial_t e_p^n)^2dxdt, \tag{67}$$

which directly implies (14).