CHOICE OF INTERIOR PENALTY COEFFICIENT FOR INTERIOR PENALTY DISCONTINUOUS GALERKIN METHOD FOR BIOT'S SYSTEM BY EMPLOYING MACHINE LEARNING*

SANGHYUN LEE†, TEERATORN KADEETHUM‡, AND HAMID M. NICK‡

Abstract. In this paper, optimal choice of the interior penalty parameter for the elliptic problems and the Biot's systems are studied by utilizing the neural network and machine learning.

Key words. Discontinuous Galerkin, Interior Penalty, Neural Network, Machine Learning, Finite Element Methods

AMS subject classifications. 68Q25, 68R10, 68U05

1. Introduction. Discontinuous Galerkin finite element method (DG) is one of the most popular non conforming finite elements employed for various realistic applications, especially with discontinuous coefficients. The idea of DG finite element methods originated from [74] and extended by several authors including [38,118,84,4], which were so called Interior Penalty Galerkin Methods. Perhaps, the most popular and successful methods in terms of the local flux conservation is DG. DG can deal robustly with general partial differential equations as well as with equations whose type changes within the computational domain such as from advection dominated to diffusion dominated [**?**, **?**]. [77] They are naturally suited for multi-physics applications, and for problems with highly varying material properties [**?**, **?**, **?**]. [26,28]

However, one of the main disadvantage of DG is that the stability and the accuracy of the scheme depends on the interior penalty parameter that needs to be chosen. These numerical analyses of DG are proved under an assumption on the interior penalty parameter, and it is crucial to employ the optimal interior penalty parameter. Generally, if the parameter is too large, DG schemes converge to the continuous Galerkin finite element methods, and ofter suffer from the linear solver. If the parameter is too small, the stability of the scheme is not guaranteed.

Thus, several studies of the lower bounds for the penalty parameter have been obtained in [**?**, **?**, **?**, **?**, **?**]. A lower boundary for the interior penalty parameter for arbitrary nonuniform polynomial order was first shown in [**?**] and it was improved in [**?**] by assuming at most one hanging node per edge of each element. In [**?**, **?**], a weighted interior penalty parameters for the cases where the diffusion coefficient is discontinuous were studied. Moreover, specific illustrations on the selection of the penalty parameters are shown in [**?**].

In this paper, we propose the method to find the optimal interior penalty parameters for both elliptic problems and the poroelasitic Biot system. However, the novelty in our paper is that we employ the neural networks and machine learning processes. We need to discuss about the machine learning algorithm / neural networks here. When various numerical simulations are necessary, but with different numerical and physical parameters (e.g mesh size, permeability, viscosity, etc), this computational framework could be very efficient and robust.

*Submitted to the editors DATE.

Funding: This work was funded by XXX.

†Department of Mathematics, Florida State University (lee@math.fsu.edu, http://https://www.math.fsu.edu/~lee/).

‡Technical University of Denmark, (teekad@dtu.dk, hamid@dtu.dk).

The paper is organized as follows. Our governing system and finite element discretizations are in section 2. Details about the machine learning algorithm is discussed in section 4. Finally, numerical results are in section 5, and the conclusions follow in section 6.

2. Mathematical Model. In this section, we briefly recapitulate the Biot system for poro-elasticity that we will discuss in this paper. Let $\Omega \subset \mathbb{R}^d$ ($d \in \{1, 2, 3\}$) be the computational domain, which is bounded by the boundary, $\partial\Omega$. The time domain is denoted by $\mathbb{T} = (0, T]$ with $T > 0$. Then the coupling between the fluid flow and solid deformation can be captured through the application of Biot's equation of poroelasticity, which is composed of linear momentum and mass balance equations [**?**].

First, the mass balance equation is given as [**?**]:

$$\rho\left(\phi c_f + \frac{\alpha - \phi}{K_s}\right)\frac{\partial}{\partial t}p + \rho\alpha\frac{\partial}{\partial t}\nabla\cdot\mathbf{u} - \nabla\cdot\boldsymbol{\kappa}(\nabla p - \rho\mathbf{g}) = g \text{ in } \Omega\times\mathbb{T}, \tag{2.1}$$

where $p(\cdot, t) : \Omega \times (0; T] \to \mathbb{R}$ is a scalar-valued fluid pressure, $\mathbf{u}(\cdot, t) : \Omega \times (0; T] \to \mathbb{R}^d$ is a vector-valued displacement, ρ is a fluid density, ϕ is an initial porosity, c_f is a fluid compressibility, $\mathbf{g}$ is a gravitational vector, g is a sink/source. Here, $\nabla\cdot\mathbf{u}$ term represents the volumetric deformation and $\boldsymbol{\kappa}$ is defined as:

$$\boldsymbol{\kappa} := \frac{\rho\mathbf{k}_m}{\mu}, \tag{2.2}$$

where $\mathbf{k}_m$ is a matrix permeability tensor and μ is a fluid viscosity. By assuming that the rock volumetric displacement can cause the matrix permeability alteration, $\mathbf{k}_m$ is defined as [**?**, **?**]:

$$\mathbf{k}_m = \mathbf{k}_{m_0}\frac{\left(1 + \frac{\varepsilon_v}{\phi}\right)^3}{1 + \varepsilon_v}, \tag{2.3}$$

where $\mathbf{k}_{m_0}$ represent an initial rock matrix permeability. Here, ε_v is the total volumetric strain, which is defined as:

$$\varepsilon_v := \mathrm{tr}(\boldsymbol{\epsilon}(\mathbf{u})) = \sum_{i=1}^{d}\boldsymbol{\epsilon}(\mathbf{u})_{ii}, \tag{2.4}$$

where $\boldsymbol{\epsilon}(\mathbf{u})$ is a strain, which is defined as:

$$\boldsymbol{\epsilon}(\mathbf{u}) := \frac{1}{2}\left(\nabla\mathbf{u} + \nabla\mathbf{u}^T\right), \tag{2.5}$$

by assuming infinitesimal displacements. Throughout this paper, we assume ρ and μ are constants but $\boldsymbol{\kappa}$ varies as a function of $\mathbf{k}_m$.

The mass balance equation (the fluid flow problem) is supplemented by the following boundary and initial conditions:

$$p = p_D \text{ on } \partial\Omega_p \times \mathbb{T}, \tag{2.6}$$

$$-\nabla\cdot\boldsymbol{\kappa}(\nabla p - \rho\mathbf{g})\cdot\mathbf{n} = q_D \text{ on } \partial\Omega_q \times \mathbb{T}, \tag{2.7}$$

$$p = p_0 \text{ in } \Omega \text{ at } t = 0, \tag{2.8}$$

where p_D and q_D are specified pressure and flux, respectively, and $\partial\Omega$ is decomposed to pressure and flux boundaries, $\partial\Omega_p$ and $\partial\Omega_q$, respectively.

Secondly, the linear momentum balance equation can be written as follows:

$$\nabla \cdot \boldsymbol{\sigma}(\mathbf{u}, p) = \mathbf{f}. \tag{2.9}$$

For the simplicity, a body force $\mathbf{f}$ is neglected in this study. Here, $\boldsymbol{\sigma}$ is total stress, which is defined as:

$$\boldsymbol{\sigma} := \boldsymbol{\sigma}(\mathbf{u}, p) = \boldsymbol{\sigma}'(\mathbf{u}) - \alpha p \mathbf{I}, \tag{2.10}$$

where $\mathbf{I}$ is the identity tensor and α is Biot's coefficient defined as [**?**]:

$$\alpha := 1 - \frac{K}{K_s}, \tag{2.11}$$

with the bulk modulus of a rock matrix K and the solid grains modulus K_s. In addition, $\boldsymbol{\sigma}'$ is an effective stress written as:

$$\boldsymbol{\sigma}' := \boldsymbol{\sigma}'(\mathbf{u}) = 2\mu_l \boldsymbol{\epsilon}(\mathbf{u}) - \lambda_l \nabla \cdot \mathbf{u}\mathbf{I}, \tag{2.12}$$

where λ_l and μ_l are Lamé constants. Thus, we can write the linear momentum balance supplemented by its boundary and initial conditions as:

$$\nabla \cdot \boldsymbol{\sigma}'(\mathbf{u}) + \alpha \nabla \cdot p\mathbf{I} = \mathbf{f} \quad \text{in } \boldsymbol{\Omega} \times \mathbb{T}, \tag{2.13}$$

$$\mathbf{u} = \mathbf{u}_D \text{ on } \partial\Omega_u \times \mathbb{T}, \tag{2.14}$$

$$\boldsymbol{\sigma}' \cdot \mathbf{n} = \boldsymbol{\sigma}_D \text{ on } \partial\Omega_t \times \mathbb{T}, \tag{2.15}$$

$$\mathbf{u} = \mathbf{u}_0 \quad \text{in } \Omega \text{ at } t = 0, \tag{2.16}$$

where $\mathbf{u}_D$ and $\boldsymbol{\sigma}_D$ are prescribed displacement and traction at boundaries, respectively, and t is time. Here, $\partial\Omega$ can be decomposed to displacement and traction boundaries, $\partial\Omega_u$ and $\partial\Omega_t$, respectively, for the solid deformation problem.

3. Numerical Discretizations. For this paper, we employ the discontnious Galerkin (DG) finite element method for the spatial discretization. Let $\mathcal{T}_h$ be the shape-regular (in the sense of Ciarlet) triangulation by a family of partitions of Ω into d-simplices T (triangles/squares in $d = 2$ or tetrahedra/cubes in $d = 3$). We denote by h_T the diameter of T and we set $h = \max_{T\in\mathcal{T}_h} h_T$. Also we denote by $\mathcal{E}_h$ the set of all edges and by $\mathcal{E}_h^I$ and $\mathcal{E}_h^\partial$ the collection of all interior and boundary edges, respectively. In the following notation, we assume edges for two dimension but the results hold analogously for faces in three dimensional case. The space $H^s(\mathcal{T}_h)$ ($s \in \mathbb{R}$) is the set of element-wise H^s functions on $\mathcal{T}_h$, and $L^2(\mathcal{E}_h)$ refers to the set of functions whose traces on the elements of $\mathcal{E}_h$ are square integrable. Let $\mathbb{Q}_l(T)$ denote the space of polynomials of partial degree at most l. Throughout the paper, we use the standard notation for Sobolev spaces and their norms. For example, let $E \subseteq \Omega$, then $\|\cdot\|_{1,E}$ and $|\cdot|_{1,E}$ denote the $H^1(E)$ norm and seminorm, respectively. For simplicity, we eliminate the subscripts on the norms if $E = \Omega$.

Since we consider the nonconforming DG methods, let

$$e = \partial T^+ \cap \partial T^-, \quad e \in \mathcal{E}_h^I,$$

where T^+ and T^- be two neighboring elements and we denote by h_e the length of the edge e. Let $\mathbf{n}^+$ and $\mathbf{n}^-$ be the outward normal unit vectors to ∂T^+ and ∂T^-,

respectively ($\mathbf{n}^{\pm} := \mathbf{n}_{|T^{\pm}}$). For any given function ξ and vector function $\boldsymbol{\xi}$, defined on the triangulation $\mathcal{T}_h$, we denote $\xi^{\pm}$ and $\boldsymbol{\xi}^{\pm}$ by the restrictions of ξ and $\boldsymbol{\xi}$ to $T^{\pm}$, respectively.

Next, we define the weighted average operator $\{\cdot\}_{\delta_e}$ as follows: for $\zeta \in L^2(\mathcal{T}_h)$ and $\boldsymbol{\tau} \in L^2(\mathcal{T}_h)^d$,

$$\{\zeta\}_{\delta e} = \delta_e \zeta^+ + (1-\delta_e)\,\zeta^-, \quad \text{and} \quad \{\boldsymbol{\tau}\}_{\delta e} = \delta_e \boldsymbol{\tau}^+ + (1-\delta_e)\,\boldsymbol{\tau}^-, \quad \text{on } e \in \mathcal{E}_h^I, \tag{3.1}$$

where δ_e is calculated by [**?**, **?**].

$$\delta_e := \frac{\kappa_e^-}{\kappa_e^+ + \kappa_e^-}. \tag{3.2}$$

Here,

$$\kappa_e^+ := \left(\mathbf{n}^+\right)^T \cdot \boldsymbol{\kappa}^+ \cdot \mathbf{n}^+, \quad \text{and } \kappa_e^- := \left(\mathbf{n}^-\right)^T \cdot \boldsymbol{\kappa}^- \cdot \mathbf{n}^-, \tag{3.3}$$

where κ_e is a harmonic average of κ_e^+ and κ_e^- read as:

$$\kappa_e := \frac{2\kappa_e^+ \kappa_e^-}{\left(\kappa_e^+ + \kappa_e^-\right)}. \tag{3.4}$$

On the other hand, for $e \in \mathcal{E}_h^{\partial}$, we set $\{\zeta\}_{\delta_e} := \zeta$ and $\{\boldsymbol{\tau}\}_{\delta_e} := \boldsymbol{\tau}$. The jump across the interior edge will be defined as

$$[\![\zeta]\!] = \zeta^+ \mathbf{n}^+ + \zeta^- \mathbf{n}^- \quad \text{and} \quad [\![\boldsymbol{\tau}]\!] = \boldsymbol{\tau}^+ \cdot \mathbf{n}^+ + \boldsymbol{\tau}^- \cdot \mathbf{n}^- \quad \text{on } e \in \mathcal{E}_h^I.$$

For $e \in \mathcal{E}_h^{\partial}$, we let $[\![\zeta]\!] := \zeta \mathbf{n}$ and $[\![\boldsymbol{\tau}]\!] := \boldsymbol{\tau} \cdot \mathbf{n}$.

Finally, we introduce the finite element space for the discontinuous Galerkin method, which is the space of piecewise discontinuous polynomials of degree k by

$$V_{h,k}^{\mathsf{DG}}(\mathcal{T}_h) := \left\{\psi \in L^2(\Omega) |\ \psi_{|_T} \in \mathbb{Q}_k(T),\ \forall T \in \mathcal{T}_h\right\}. \tag{3.5}$$

Moreover, we use the notation:

$$(v,w)_{\mathcal{T}_h} := \sum_{T \in \mathcal{T}_h} \int_T v\, w dx, \quad \forall\ v, w \in L^2(\mathcal{T}_h),$$

$$\langle v,w\rangle_{\mathcal{E}_h} := \sum_{e \in \mathcal{E}_h} \int_e v\, w\, d\gamma, \quad \forall\, v, w \in L^2(\mathcal{E}_h).$$

3.1. Pressure equation. First, we introduce the backward Euler DG approximation to (2.1). We define a partition of the time interval $0 =: t^0 < t^1 < \cdots < t^N := \mathbb{T}$ and denote the uniform time step size by $\delta t := t^n - t^{n-1}$. The DG finite element space approximation of the pressure $p(\mathbf{x}, t)$ is denoted by $P(\mathbf{x}, t) \in V_{h,k}^{DG}$. Let $P^n := P(\mathbf{x}, t^n)$ for $0 \le n \le N$. We set a given initial condition for the pressure as P^0 and assume the displacement at time t, $\mathbf{u}(\cdot, t)$ is given. For the simplicity the gravity and the source/sink terms are neglected. Then, the time stepping algorithm reads as follows: Given P^{n-1},

$$\text{Find } P^n \in V_{h,k}^{DG} \text{ such that } \mathcal{S}_\theta(P^n, w) = \mathcal{F}_\theta(w), \quad \forall\, w \in V_{h,k}^{DG}, \tag{3.6}$$

where $\mathcal{S}_\theta$ and $\mathcal{F}_\theta$ are the bilinear form and linear functional as defined by

$$\mathcal{S}_\theta(v,w) := \frac{\rho}{\delta t}\left(\phi c_f + \frac{\alpha-\phi}{K_s}\right)(v,w)_{\mathcal{T}_h} + (\boldsymbol{\kappa}\nabla v, \nabla w)_{\mathcal{T}_h} - \left\langle \{\boldsymbol{\kappa}\nabla v\}_{\delta_e}, [\![w]\!]\right\rangle_{\mathcal{E}_h^1} - \theta\left\langle [\![v]\!], \{\boldsymbol{\kappa}\nabla w\}_{\delta_e}\right\rangle_{\mathcal{E}_h^1} + \beta(k)\langle h_e^{-1}\kappa_e [\![v]\!], [\![w]\!]\rangle_{\mathcal{E}_h^1}, \quad \forall v, w \in V_{h,k}^{DG}, \tag{3.7}$$

and

$$\mathcal{F}_\theta(w) := \frac{1}{\delta t}(P^{n-1}, w)_{\mathcal{T}_h} - \rho\alpha(\frac{\partial}{\partial t}\nabla\cdot\mathbf{u}, w)_{\mathcal{T}_h} - \langle q_D, [\![w]\!]\rangle_{\mathcal{E}_h^{N,\partial}} - \theta\left\langle p_D, \{\boldsymbol{\kappa}\nabla w\}_{\delta_e}\right\rangle_{\mathcal{E}_h^{D,\partial}} + \beta(k)\langle h_e^{-1}\kappa_e p_D, [\![w]\!]\rangle_{\mathcal{E}_h^{D,\partial}}, \quad \forall w \in V_{h,k}^{DG}. \tag{3.8}$$

(see Luigi's thesis to clarify the reference) The choice of θ leads to different DG algorithms. For example, i) $\theta = 1$ for SIPG(β)$-k$ methods [**?**, **?**], which later has been extended to the advection-diffusion problems in [**?**, **?**], ii) $\theta = -1$ for NIPG(β)$-k$ methods [**?**], and iii) $\theta = 0$ for IIPG(β)$-k$ method [**?**].

The interior penalty parameter, $\beta(k)$, is a function of polynomial degree approximation, k. Here, h_e is a characteristic length of the edge $e \in \mathcal{E}_h$ calculated as:

$$h_e := \frac{\text{meas}\,(T^+) + \text{meas}\,(T^-)}{2\,\text{meas}(e)}, \tag{3.9}$$

where meas(.) represents a measurement operator, measuring length, area, or volume. Several analysis for the choice of the interior penalty parameter are shown in [X] and this β is the quantatiy that we investigate in this paper.

3.2. Poroelasticity problem. For the displacement $\mathbf{u}$, we employ the classical continuous Galerkin (CG)finite element methods for the spatial discretizations as in [**?**, **?**] where the function space is defined as

$$W_{h,k}^{\mathsf{CG}}(\mathcal{T}_h) := \left\{\boldsymbol{\psi}_u \in \mathbb{C}^0(\Omega;\mathbb{R}^d) : \boldsymbol{\psi}_u|_T \in \mathbb{Q}_k(T;\mathbb{R}^d), \forall T \in \mathcal{T}_h\right\}, \tag{3.10}$$

where $\mathbb{C}^0(\Omega;\mathbb{R}^d)$ denotes the space of vector-valued piecewise continuous polynomials, $\mathbb{Q}_k(T;\mathbb{R}^d)$ is the space of polynomials of degree at most k over each element T.

The CG finite element space approximation of the displacement $\mathbf{u}(\mathbf{x},t)$ is denoted by $\mathbf{U}(\mathbf{x},t) \in W_{h,k}^{CG}$. Let $\mathbf{U}^n := \mathbf{U}(\mathbf{x},t^n)$ for $0 \le n \le N$. We set a given initial condition for the pressure as $\mathbf{U}^0$ and the pressure at time t, P^n is given from the previous section. Then, the time stepping algorithm reads as follows: Given ¶^{n},

$$\text{Find } \mathbf{U}^n \in W_{h,k}^{CG} \text{ such that } \mathcal{A}(\mathbf{U}^n, \mathbf{w}) = \mathcal{D}(\mathbf{w}), \quad \forall\, \mathbf{w} \in W_{h,k}^{CG}, \tag{3.11}$$

where $\mathcal{A}$ and $\mathcal{D}$ are the bilinear form and linear functional as defined as

$$\mathcal{A}(\mathbf{v},\mathbf{w}) := \sum_{T\in\mathcal{T}_h}\int_T \boldsymbol{\sigma}'(\mathbf{v}) : \nabla^s\mathbf{w}\, dV + \sum_{T\in\mathcal{T}_h}\int_T \alpha\nabla P^n.\nabla\mathbf{w}\, dV, \quad \forall\, \mathbf{v},\mathbf{w} \in W_{h,k}^{CG}, \tag{3.12}$$

and

$$\mathcal{D}(\mathbf{w}) := \sum_{T\in\mathcal{T}_h}\int_T \mathbf{f}\mathbf{w}\, dV + \sum_{e\in\mathcal{E}_h^N}\int_e \boldsymbol{\sigma}_D\mathbf{w}\, dS, \quad \forall\, \mathbf{w} \in W_{h,k}^{CG}, \tag{3.13}$$

4. Machine Learning Algorithm. In this section, we discuss the details for the neural network and machine learning algorithm that we employ for our numerical scheme to find the optimal choice of interior penalty parameter. The overall algorithm is based on [x]

Our analysis leads to the algorithm in Algorithm 4.1.

Algorithm 4.1 Build tree

Define $P := T := \{\{1\}, \dots, \{d\}\}$
while $\#P > 1$ **do**
  Choose $C' \in \mathcal{C}_p(P)$ with $C' := \operatorname{argmin}_{C \in \mathcal{C}_p(P)} \varrho(C)$
  Find an optimal partition tree $T_{C'}$
  Update $P := (P \backslash C') \cup \{\bigcup_{t \in C'} t\}$
  Update $T := T \cup \{\bigcup_{t \in \tau} t : \tau \in T_{C'} \backslash \mathcal{L}(T_{C'})\}$
end while
return T

5. Numerical results. In this section, several numerical examples are illustrated to show the capability of our proposed algorithm and computational framework.

5.1. Elliptic Problems. First, we consider the simplified equation of the flow problems (2.1). By assuming that the pressure doesn't depend on the time and $\mathbf{u}$ is a given constant, we obtain

$$-\nabla \cdot (\boldsymbol{\kappa} \nabla p) = g \text{ in } \Omega \times \mathbb{T}. \tag{5.1}$$

Moreover, we note that the gravity ($\mathbf{g}$) and the source/sink term (g) are neglected for the simplicity.

5.1.1. The effect of a polynomial degree approximation (k). We investigate the effect of a polynomial degree approximation on the choice of optimal β using different solvers and discretization schemes. We study five k values (1, 2, 3, 4, and 5) and use Algorithm 5.1 for each case, i.e. each case has different combination of solver, discretization, and k.

We take $\Omega = [0, 1]^2$ and choose the exact solution in Ω as:

$$p(x, y) := \sin(x + y), \tag{5.2}$$

where x and y represent points in x- and y-direction, respectively, and $\boldsymbol{\kappa}$ is read:

$$\boldsymbol{\kappa} := \kappa \boldsymbol{I}, \tag{5.3}$$

where κ represents a multiplied coefficient in a range of $[1.0, 1 \times 10^{-18}]$ for the whole domain, and $\boldsymbol{I}$ is an identity matrix. Subsequently, g is chosen as:

$$g(x, y) := 2.0 \cos(x + y) \tag{5.4}$$

to satisfy the exact solution. Furthermore, the homogeneous boundary conditions are applied to all boundaries using Equation (5.2). The results are presented in Table 1,

Algorithm 5.1 Investigation procedure for the elliptic problem

Initialize the set of $\boldsymbol{\kappa}$ that used in the investigation
for $i < n_{\boldsymbol{\kappa}}$, where $n_{\boldsymbol{\kappa}}$ is the size of the specified $\boldsymbol{\kappa}$ list **do**
  Assign $\boldsymbol{\kappa} := \boldsymbol{\kappa}[i]$
  Assign $\beta := \beta_0$, where $\beta_0 = 100.00$
  Initialize error convergence rate list, where all members $= k$
  while $\forall$ error convergence rate $\in$ error convergence rate list $= [k - \epsilon, k + \epsilon]$, where $\epsilon = 0.5 \times k$ **do**
    Clear error convergence rate list
    Assign $h := h_0$, where $h_0 =$
    Update $\beta := 0.99 \times \beta$ {Except the first loop}
    for $j < n_h$, where $n_h = 6$ **do**
      Calculate H_0^1 norm
      Calculate error convergence rate
      Append error convergence rate list
      Update $h := 0.5 \times h$
    end for
  end while
  return β, this β is the smallest β, which the optimal error convergence rate can be observed.
end for

Table 1: The lowest β value that provides the optimal error convergence rate solution with different k, discretization, and solver. Note that $\boldsymbol{\kappa}$ is homogeneous.

	SIPG		IIPG	
k	direct solver	iterative solver	direct solver	iterative solver
1	1.11	1.11	0.83	0.83
2	2.80	2.80	2.74	2.74
3	5.79	5.79	5.68	5.68
4	9.99	9.99	9.79	9.79
5	14.97	14.97	14.67	14.67

and it shows that the lowest β values of SIPG are higher than those of IIPG. Besides, the type of solver, direct and iterative solvers, does not influence this value, and the smallest β increases as k increases.

The choice of β, however, impacts the number of linear iterative solver significantly, as presented in Figures 1 and 2 for SIPG and IIPG results, respectively. As one observes, when β is increased, the number of iteration increases. On the other hand, when β is approaching zero, the number of iteration, again, increases dramatically; and subsequently, the solution becomes unstable. Hence, the choice of β is essential because if we select a large value of β, our simulation takes more time to converge, while a small value can also cause the higher number of iteration and more importantly unstable solution.

To predict the optimal β for the iterative solver, i.e., requires minimum linear solver iteration and provides stable solutions, we firstly identify the parameters, which impact the number of iteration (dependent variable). The chi-squared test [**?**] is

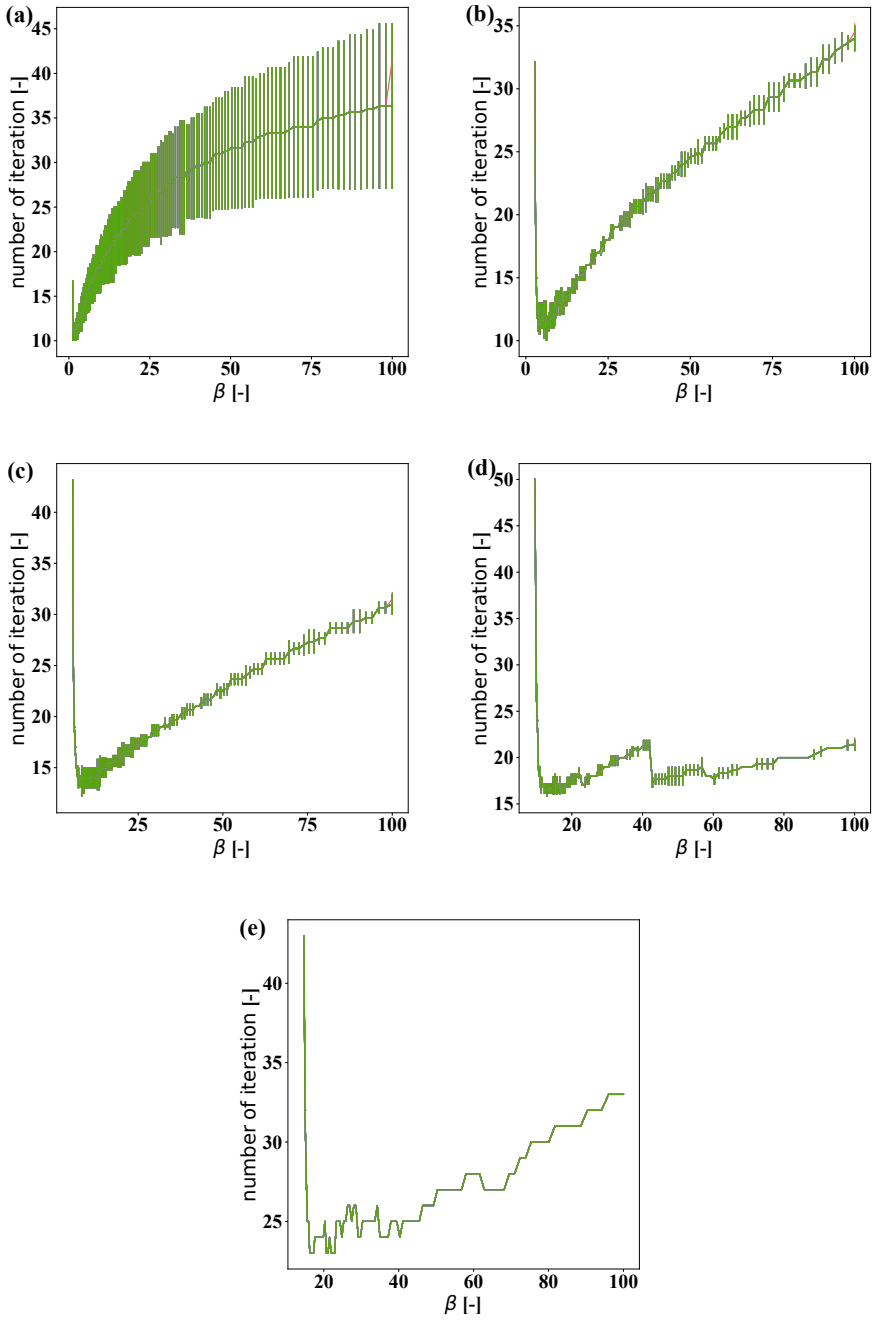

Fig. 1: Number of linear iterative solver of SIPG for (**a**) $k = 1$, (**b**) $k = 2$, (**c**) $k = 3$, (**d**) $k = 4$, and (**e**) $k = 5$. Note that each line represents different value of κ, and the error bar shows mean and standard derivation of each number of iteration.

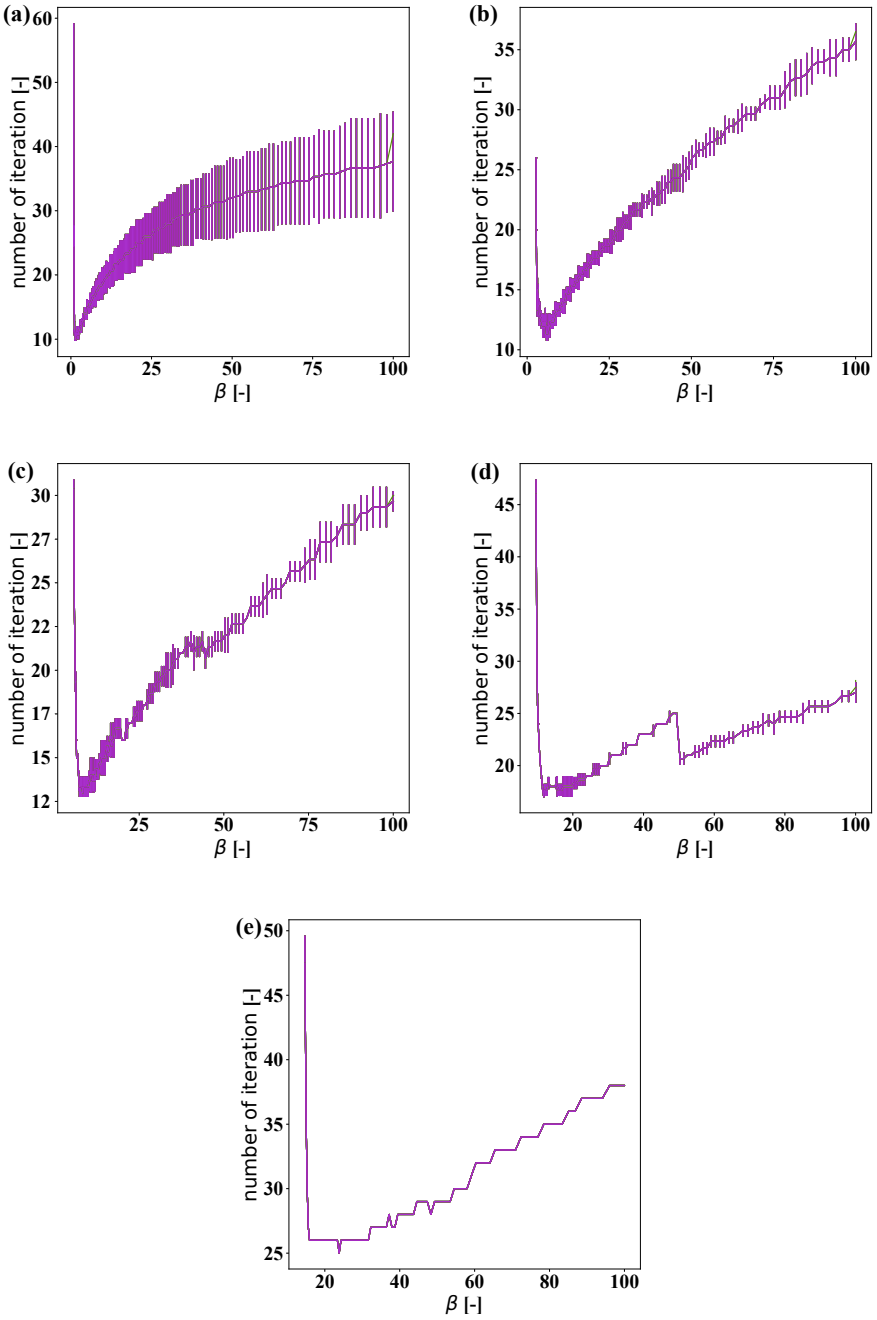

Fig. 2: Number of linear iterative solver of IIPG for (**a**) $k = 1$, (**b**) $k = 2$, (**c**) $k = 3$, (**d**) $k = 4$, and (**e**) $k = 5$. Note that each line represents different value of κ, and the error bar shows mean and standard derivation of each number of iteration.

performed, and its results are provided in Table 2. From this table, θ, β, h, and k have a p-value less than 0.025; therefore, we include these variables as independent variables for a further predictive model development.

Table 2: Elliptic equation with continuous exact solution: p-value results for each explanatory variable

Variable	p-value
θ	≈ 0.00
κ	≈ 1.00
β	≈ 0.00
h	≈ 0.00
k	≈ 0.00

In total, we have TODO data points and split training, validation, and test sets uniformly [**?**] using $[0.8, 0.1, 0.1]$ splitting ratio. We use training set to train the multi-variable regression and ANN models, validation set for hyper-parameter tuning for the ANN model, and test set for comparison performances between the multi-variable regression and ANN model. We begin with building the multi-variable regression [**?**] as follows:

Remark 5.1. The reason that κ does not effect the results: existence of κ as a coefficient of the penalty term.

$$\begin{aligned} \text{number of iteration} = \alpha &+ \gamma_1 \times \theta \\ &+ \gamma_2 \times \beta \\ &+ \gamma_3 \times h \\ &+ \gamma_4 \times k \end{aligned} \tag{5.5}$$

where $\alpha = 16.93$, $\gamma_1 = -1.11$, , $\gamma_2 = 0.21$, , $\gamma_3 = -9.85$, and $\gamma_4 = 0.02$. The r^2 and explained variance score is 0.60 and 0.60, respectively.

Subsequently, we develop ANN model using four input and one output as presented in Figure 3 and use number of hidden layer (n_{hl}) and number of neuron (n_n) for hyper-parameter tuning. For the sake of simplicity we assume each hidden layer has the same number of neuron. Rectified Linear Unit (ReLU) is used as an activation function for each neuron of the hidden layer.

Table 3 illustrates that the mean squared error of the validation set is generally decreased as N_{hl} and N_n are increased. The neural network performance, however, is not significantly improved when $N_{hl} > 2$ and $N_n > 80$. Hence, we select $N_{hl} = 2$ and $N_n = 80$ for the test set, and the r^2 and explained variance score is 0.98 and 0.98, respectively. These results illustrate the improvement of the prediction performance as the r^2 and explained variance score are improved from the multi-variable regression significantly.

5.1.2. The effect of a type of exact solution and heterogeneous coefficient. Next, we investigate the effect of a type of exact solution, continuous and discontinous solutions, on the choice of optimal β using different solvers and θ schemes. Moreover, $\boldsymbol{\kappa}$ is heterogeneous for both types of solution. For the continuous solution, we take $\Omega = [0, 1]^2$ and choose the exact solution in Ω as:

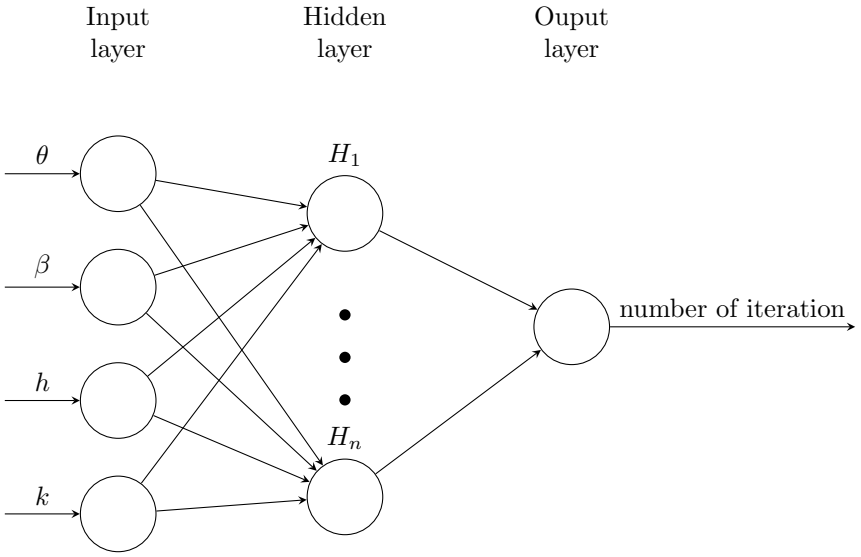

Fig. 3: Neural network architecture used for the elliptic problem with continuous exact solution. The number of hidden layers, N_{hl}, and the number of neuron for each hidden layer, N_n, are used as the sensitivity analysis parameters.

Table 3: Elliptic equation with continuous exact solution: Mean squared error of the validation set for different number of hidden layers N_{hl} and different number of neurons per layer N_n

N_{hl} \ N_n	10	20	40	80
2	4.27	1.41	1.20	0.95
4	2.21	1.43	2.13	1.58
8	3.60	1.74	1.60	0.98

$$p(x,y) := \sin(x+y), \tag{5.6}$$

which is similar to the previous section. However, $\boldsymbol{\kappa}$ is chosen as:

$$\boldsymbol{\kappa} := \kappa \sin(x+y)\boldsymbol{I}. \tag{5.7}$$

Subsequently, g is chosen as:

$$g(x,y) := 4.0\kappa \sin(x+y)\cos(x+y). \tag{5.8}$$

Next, for the discontinuous solution, we take $\Omega = [0,1]^1$ and choose the exact solution in Ω as:

$$p = \begin{cases} 2.0x\frac{\kappa_1}{\kappa_0+\kappa_1} & \text{if } x \leq 0.5, \\ \frac{(2x-1)\kappa_0+\kappa_1}{\kappa_0+\kappa_1} & \text{if } x > 0.5, \end{cases} \tag{5.9}$$

where κ_1 and κ_2 represent multiplied coefficients for the $0 \leq x \leq 0.5$ and $1.0 \geq x > 0.5$ subdomains, respectively. κ_1 and κ_2 have the same range of $[1.0, 1.0 \times 10^{-18}]$ and $\kappa_1 \neq \kappa_2$. Consequently, the $\boldsymbol{\kappa}$ for the discontinuous exact solution is read:

$$\boldsymbol{\kappa} := \begin{cases} \kappa_1 \boldsymbol{I} & \text{if } 0.0 \leq x \leq 0.5, \\ \kappa_2 \boldsymbol{I} & \text{if } 1.0 \geq x > 0.5. \end{cases} \tag{5.10}$$

Subsequently, the boundary conditions are applied as follows:

$$p = \begin{cases} 0.0 & \text{at } x = 0.0, \\ 1.0 & \text{at } x = 1.0, \end{cases} \tag{5.11}$$

We use Algorithm 5.1 where $k = 1$ for each case, i.e. each case has different combination of solver, θ, and the exact solutions. The results are shown in Table 4. The results of SIPG illustrate the similarity between the continuous and discontinuous solutions. The results of IIPG, however, show a discrepancy as the lowest β values that provides the optimal convergence rate solution are different between the continuous and discontinuous solutions. The type of solver, direct and iterative solvers, does not influence this value.

Table 4: The lowest β value that provides the optimal convergence rate solution with different type of exact solution (continuous or discontinuous), θ, and solver. Note that $\boldsymbol{\kappa}$ is heterogeneous, and $k = 1$.

	SIPG		IIPG	
exact solution	direct solver	iterative solver	direct solver	iterative solver
continuous	1.11	1.11	0.83	0.83
discontinuous	1.11	1.11	0.89	0.89

Similar to the continuous solution, the choice of β influences the number of linear iterative solver significantly, as illustrated in Figure 4. In short, when β is increased, the number of iteration increases, while the number of iteration increases sharply before the solution becomes unstable.

Following the procedure, i.e. predicting the optimal β for the iterative solver, used for the continuous solution, we firstly evaluate each independent variable using chi-squared test [**?**]. Its results are provided in Table 5, and this table shows that θ methods, β, and h have a p-value less than 0.025. Hence, we include these variables as independent variables for a further predictive model development.

We follow the same procedures, splitting technique and model development, as utilized in the previous model. The developed multi-variable regression [**?**] is read:

$$\begin{aligned} \text{number of iteration} = \alpha &+ \gamma_1 \times \theta \\ &+ \gamma_2 \times \beta \\ &+ \gamma_3 \times h \end{aligned} \tag{5.12}$$

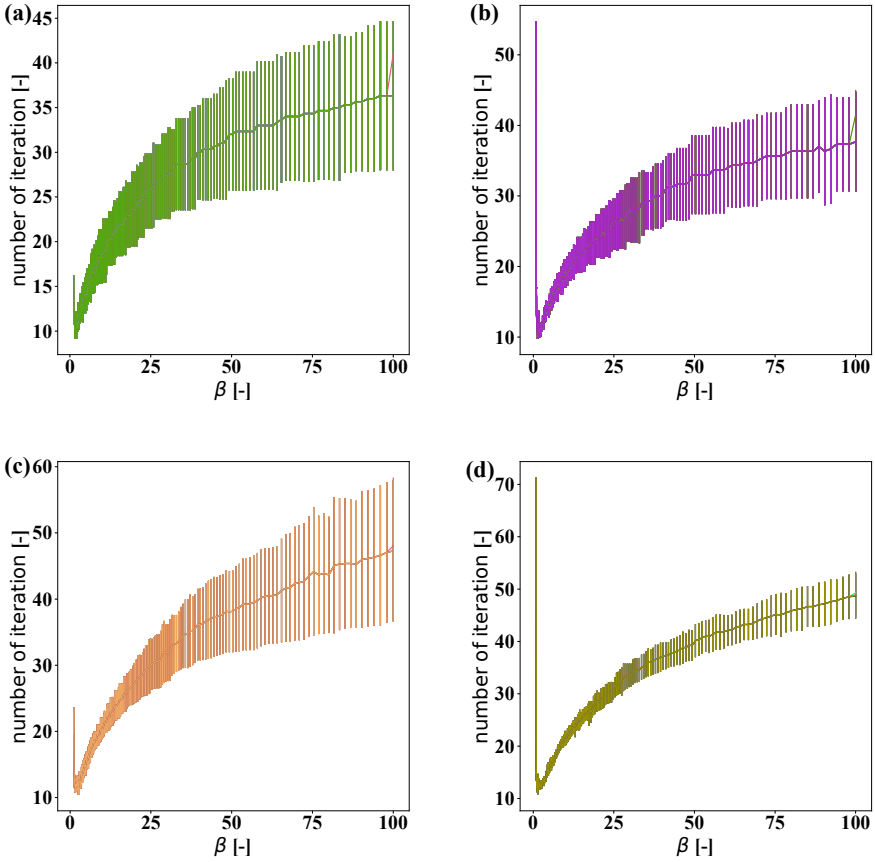

Fig. 4: Number of linear iterative solver of cases which $\boldsymbol{\kappa}$ is heterogeneous and exact solution is continuous: (**a**) SIPG, (**b**) IIPG, $\boldsymbol{\kappa}$ is heterogeneous and exact solution is discontinuous: (**c**) SIPG, and (**d**) IIPG. Note that each line represents different value of κ for the continuous solution, and κ_1 and κ_2 for the discontinuous solution. The error bar shows mean and standard derivation of each number of iteration.

Table 5: Elliptic equation with discontinuous exact solution: p-value results for each explanatory variable

Variable	p-value
θ	≈ 0.00
κ_0	≈ 1.00
κ_1	≈ 1.00
β	≈ 0.00
h	≈ 0.00

where $\alpha = 19.59$, $\gamma_1 = -0.86$, $\gamma_2 = 0.42$, and $\gamma_3 = -19.51$. The r^2 and explained variance score is 0.84 and 0.84, respectively.

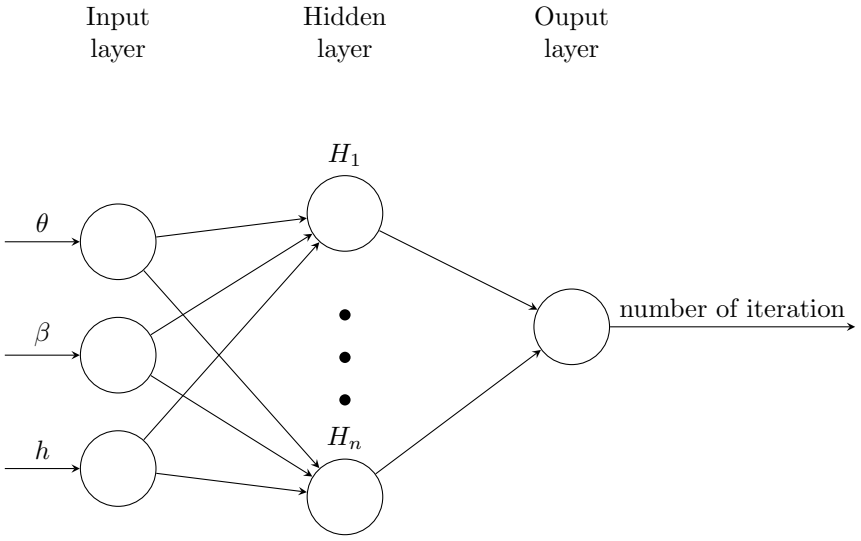

Fig. 5: Neural network architecture used for the elliptic problem with discontinuous exact solution. The number of hidden layers, N_{hl}, and the number of neuron for each hidden layer, N_n, are used as the sensitivity analysis parameters.

Next, we develop ANN model using three input and one output as shown in Figure 5 and use number of hidden layer (n_{hl}) and number of neuron (n_n) for hyper-parameter tuning. ReLU is used as an activation function for each neuron of the hidden layer.

Table 6 presents that the mean squared error of the validation set is decreased, as expected, as N_{hl} and N_n are increased. We select $N_{hl} = 2$ and $N_n = 80$ for the test set, and the r^2 and explained variance score is 0.98 and 0.98, respectively. The neural network outperforms the multi-variable regression.

Table 6: Elliptic equation with discontinuous exact solution: Mean squared error of the validation set for different number of hidden layers N_{hl} and different number of neurons per layer N_n

N_{hl} \ N_n	10	20	40	80
2	5.49	1.57	3.16	1.67
4	1.93	1.53	1.71	1.63
8	1.44	2.34	1.43	1.56

5.2. Biot's equations. TODO add oscillation examples

$$\boldsymbol{\kappa} := \begin{cases} \kappa_1 \boldsymbol{I} & \text{if } 0.0 \leq x \leq 0.5, \\ \kappa_2 \boldsymbol{I} & \text{if } 1.0 \geq x > 0.5, \end{cases} \tag{5.13}$$

and we define a ratio between κ_2 and κ_1 as:

$$\kappa_{mult} = \frac{\kappa_2}{\kappa_1} \tag{5.14}$$

Algorithm 5.2 Investigation procedure for the Biot's equations

```
Initialize: sets of each variables; κ, κ_mult, h, and β
Initialize the result vector with BOOL solution quality {solution quality; 0 - Good
and 1 - Bad}
for i < n_κ, where n_κ is the size of the specified κ list do
  Assign κ_1 := κ[i]
  for j < n_κmult, where n_κmult is the size of the specified κ_mult list do
    Assign κ_2 := κ_mult[j] × κ_1
    for k < n_h, where n_h is the size of the specified h list do
      Assign h := h[k]
      for l < n_β, where n_β is the size of the specified β list do
        Assign β := β[l]
        Perform simulation
        if Solution is stable then
          if Solution exhibits nonphysical oscillation then
            Assign 1 -> result
          else
            Assign 0 -> result
          end if
        else
          Assign 1 -> result
        end if
      end for
    end for
  end for
end for
```

Similar to the previous sections, we begin with the χ^2 test to find the statistically significant explanatory variables. The χ^2 test result is presented in Table 7, and it show that all variables, κ_1, κ_2, κ_{mult}, and h, have p-value less than 0.025. Therefore, these variables are included as independent variables to develop following predictive models.

Table 7: Biot's equations: p-value results for each explanatory variable

Variable	p-value
κ_1	≈ 0.00
κ_2	≈ 0.00
κ_{mult}	≈ 0.00
β	≈ 0.00
h	≈ 0.01

We use the same splitting technique and ratio $[0.8, 0.1, 0.1]$ employed in the previous section. The multi-variable logistic regression [**?**] is read as follows:

$$log\left(\frac{p(\mathbb{O}=1)}{1-(p(\mathbb{O}=1))}\right) = \alpha + \gamma_1 \times \boldsymbol{\kappa}_1 + \gamma_2 \times \boldsymbol{\kappa}_2 + \gamma_3 \times \kappa_{mult} + \gamma_4 \times \beta + \gamma_5 \times h \tag{5.15}$$

and

$$\mathbb{O} = \begin{cases} 1, & \text{if } p(\mathbb{O}=1) \geq 0.5 \\ 0, & \text{if } p(\mathbb{O}=1) < 0.5 \end{cases} \tag{5.16}$$

where $\alpha = -1.19$, $\gamma_1 = -0.06$, , $\gamma_2 = 0.68$, $\gamma_3 = 5.55$, $\gamma_4 = -5.49$, and $\gamma_5 = 0.32$. The accuracy of the logistic regression model is 0.80, and its confusion matrix is presented in Table 8. From this table, one can infer that the number of 'false positive' is much higher compared to that of 'false negative', which may result in the bad solution obtained from the finite element model.

Table 8: Biot's equations: Confusion matrix of the logistic regression for the test set

total test set = 1415		Test set values	
		Positive (1)	Negative (0)
Predicted values	Positive (1)	959	74
	Negative (0)	210	172

Table 9: Biot's equations: Accuracy of the validation set for different number of hidden layers N_{hl} and different number of neurons per layer N_n

N_{hl} \ N_n	10	20	40	80	120
2	0.89	0.93	0.93	0.93	0.92
4	0.89	0.93	0.93	0.93	0.93
8	0.88	0.92	0.93	0.93	0.93
16	0.73	0.73	0.73	0.27	0.27
32	0.73	0.73	0.73	0.73	0.73

The accuracy of the ANN using the test set is 0.93.

Conclusion

- For Darcy with smooth coefficients : need a minimum β but not the maximum for optimal convergence rate. However, the iterative solver is affected.
- For Darcy with non-smooth coefficients : To observe sub-optimal convergence rate, minimum and maximum bound for β is required.
- For Biot's equations : to avoid oscillations, optimal choice of β is required. Here we employ SIPG and iterative solver.

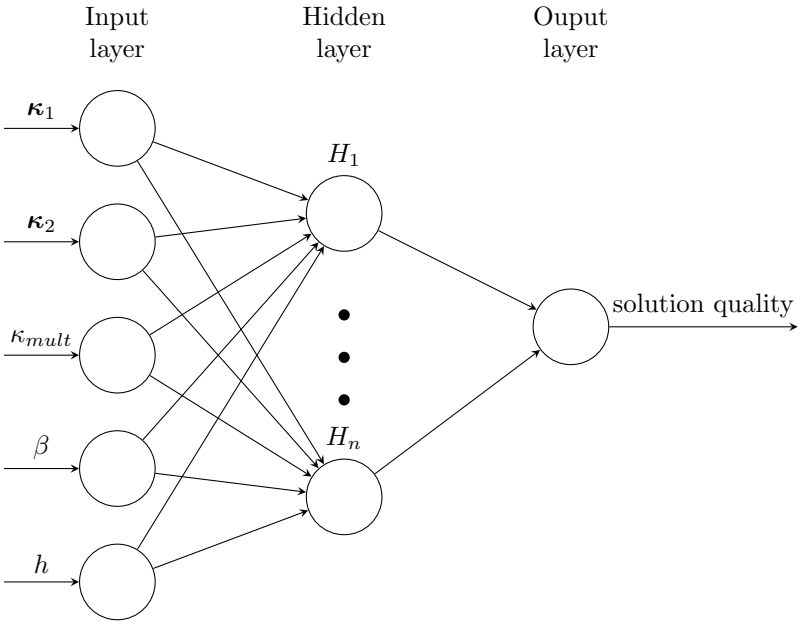

Fig. 6: Neural network architecture used for the Biot's equation. The number of hidden layers, N_{hl}, and the number of neuron for each hidden layer, N_n, are used as the sensitivity analysis parameters.

Table 10: Biot's equations: Confusion matrix of the artificial neural network (ANN) for the test set

total test set = 1415		Test set values	
		Positive (1)	Negative (0)
Predicted values	Positive (1)	961	72
	Negative (0)	15	367

6. Conclusions. Some conclusions here.

Acknowledgments. We would like to acknowledge the assistance of volunteers in putting together this example manuscript and supplement.