Meshfree-based physics-informed neural networks for the unsteady Oseen equations

Keyi Peng(彭珂依)[1,†], Jing Yue(岳靖)[2,‡], Wen Zhang(张文)[2,§], and Jian Li(李剑)[1,¶]

[1] *School of Mathematics and Data Science, Shaanxi University of Science and Technology, Xi'an 710021, China*
[2] *School of Electrical and Control Engineering, Shaanxi University of Science and Technology, Xi'an 710021, China*

(Received 6 July 2022; revised manuscript received 8 October 2022; accepted manuscript online 21 October 2022)

We propose the meshfree-based physics-informed neural networks for solving the unsteady Oseen equations. Firstly, based on the ideas of meshfree and small sample learning, we only randomly select a small number of spatiotemporal points to train the neural network instead of forming a mesh. Specifically, we optimize the neural network by minimizing the loss function to satisfy the differential operators, initial condition and boundary condition. Then, we prove the convergence of the loss function and the convergence of the neural network. In addition, the feasibility and effectiveness of the method are verified by the results of numerical experiments, and the theoretical derivation is verified by the relative error between the neural network solution and the analytical solution.

Keywords: physics-informed neural networks, the unsteady Oseen equation, convergence, small sample learning

PACS: 02.30.Jr, 02.60.Cb, 07.05.Mh **DOI:** 10.1088/1674-1056/ac9cb9

1. Introduction

Partial differential equations (PDEs) are derived mathematically or physically based on some basic principles, and they play vital roles to model many important phenomena in science and engineering, such as the Allen Cahn equation of the phase field model, the Fisher KPP equation for studying animal gene transmission, KdV equations, and Navier–Stokes equations. Among many equations, Navier–Stokes equations are a special existence, since they involve a trilinear term, which changes the essence of all the problems considered so far, from linear to nonlinear. The Oseen equation can be obtained by replacing the trilinear term with $(b\cdot\nabla)u$.[1–9] Some researchers would study the Oseen equation first, because the Oseen equations are significant for the development of robust and efficient numerical methods in solving the incompressible Navier–Stokes equations. In the past decades, traditional methods have been development greatly, e.g., researchers used FEM,[10] the parallel FEM,[11] the weak Galerkin FEM,[12] and other traditional methods to solve the Oseen equations.[13–15] Although some good results are obtained by these methods, the generation of grid often has a certain influence on the computational efficiency.

In the past few years, it is very popular to use deep learning techniques to solve a series of computational science and finance problems.[16–20] With the development of this technique, many researchers apply deep learning to solve PDEs.[21–32] In particular, the method of physics-informed neural networks (PINNs) is proposed in Ref. [33], and it is a popular method to solve PDEs using deep learning. In addition, some PINNs-based extensions are proposed.[34–39] They can be used to different types of PDEs, such as integro-differential equations,[40] heat transfer equations,[41] Schrödinger equations.[33] In order to obtain the desired neural network, researchers trained the neural network by minimizing the loss function at randomly spatiotemporal points. The loss function of PINNs can be defined by the residual differential operators, the initial condition and the boundary condition. The PINNs algorithm avoids the form of mesh and reduces computational complexity. In addition, small sample learning (SSL) can reduce the cost of data collection, so it has been frequently used in neural machine translation,[42] museum art discovery,[43] and so on. Meanwhile, researchers trained neural networks to solve PDEs by applying a small number of samples, many numerical experimental results show that it is practicable.[44]

In this paper, we use the meshfree-based PINNs to solve the unsteady Oseen equations. This method is meshfree since it can approximate the solution by neural network instead of linear combination of basis functions. Moreover, the proposed sampling method also avoids grid generation during the training process of the neural networks. According to the thought of SSL, we collect the small number of random spatiotemporal points to optimize the neural networks by minimizing the loss function, such that the neural networks satisfy the differential operators, initial condition and boundary condition. Furthermore, we prove the convergence of the loss function and the

†Corresponding author. E-mail: keyi_2000@163.com
‡Corresponding author. E-mail: yjing1995@163.com
§Corresponding author. E-mail: zw15290229577@163.com
¶Corresponding author. E-mail: jianli@sust.edu.cn

© 2023 Chinese Physical Society and IOP Publishing Ltd

convergence of the neural network. Finally, we verify the feasibility of this method by numerical experiments.

The outline of the paper is as follows. In Section 2, we give some mathematical preparations and assumptions. In Sections 3 and 4, we prove the convergence of loss function and neural network, respectively. Numerical experiments are given to verify the effectiveness of this method in Section 5.

2. Methodology

Let $\Omega \subset R^d$ $(d = 2,3,\ldots)$ be a bounded domain with Lipschitz-continuous boundary $\partial\Omega \subset R^{d-1}$ and $(0,T]$ be a time interval with $T < \infty$. We consider the following unsteady Oseen equations

$$u_t - \nu\Delta u + (b\cdot\nabla)u + \nabla p = f, \quad \text{in} \quad \Omega\times(0,T], \tag{1}$$

$$\nabla\cdot u = 0, \quad \text{in} \quad \Omega\times(0,T], \tag{2}$$

$$u = 0, \quad \text{on} \quad \partial\Omega\times(0,T], \tag{3}$$

$$u = u_0, \quad \text{on} \quad \Omega\times\{0\}, \tag{4}$$

where $u : (0,T]\times\Omega \to R^2$ represents the velocity vector, $p : (0,T]\times\Omega \to R$ the pressure, $f : (0,T]\times\Omega \to R^2$ the external body force, $\nu > 0$ the kinematic viscosity, $u_0 : \Omega \to R^2$ the initial velocity satisfying

$$\nabla\cdot u_0 = 0,$$

and $b : (0,T]\times\Omega \to R^2$ a solenoidal vector field satisfying $\nabla\cdot b = 0$.

For notational convenience, let us denote $\Omega_{\mathrm{T}} = \Omega\times(0,T]$, $\partial\Omega_{\mathrm{T}} = \partial\Omega\times(0,T]$, $\Omega_0 = \Omega\times\{0\}$, $\mathcal{G}[u,p](t,x) = u_t - \nu\Delta u + (b\cdot\nabla)u + \nabla p - f$. Meanwhile, we use the standard notations $W^{m,p}(\Omega)$ for Sobolev spaces, the norms $\|\cdot\|_{m,p}$ and the seminorms $|\cdot|_{m,p}$

$$W^{m,p}(\Omega) = \left\{v \in L^1_{\mathrm{loc}}(\Omega) : \|v\|_{m,p,\Omega} < \infty\right\},$$

$$\|v\|_{W^{m,p}}(\Omega) = \|v\|_{m,p,\Omega} = \left\{\sum_{|\alpha|\le m}\|D^\alpha v\|^p_{0,p,\Omega}\right\}^{1/p},$$

$$1 \le p \le \infty,$$

$$|v|_{m,p,\Omega} = \left\{\sum_{|\alpha|=m}\|D^\alpha v\|^p_{0,p,\Omega}\right\}^{1/p}, \quad 1 \le p \le \infty,$$

for $p = 2$, we denote $H^m(\Omega) = W^{m,2}(\Omega)$ and set

$$\|v\|_k = \sqrt{(v,v)_k} = \left\{\sum_{|\alpha|=0}^{k}\int_\Omega (D^\alpha_w v)^2\,\mathrm{d}x\right\}^{1/2},$$

$$\|q\|_0 = \left(\int_\Omega q^2\,\mathrm{d}x\right)^{1/2}, \quad X = H^1_0(\Omega)^2,$$

$$Y = L^2(\Omega)^2, \quad M = L^2_0(\Omega) = \left\{q \in L^2(\Omega) : \int_\Omega q\,\mathrm{d}x = 0\right\}.$$

We define the closed subset V of X and V_1 of Y, respectively,

$$V = \{v \in X, \quad \nabla\cdot v = 0\},$$

$$V_1 = \{v \in Y, \quad \nabla\cdot v = 0, v\cdot n \mid \partial\Omega = 0\}.$$

To prove the existence and uniqueness of the solution, we first make the following assumptions

(i) The unique solution $(v,q) \in X\times M$ of the steady Stokes problem

$$-\nu\Delta v + \nabla q = g, \quad \nabla\cdot v = 0 \quad \text{in} \quad \Omega, \quad v \mid \partial\Omega = 0,$$

for prescribed $g \in H^{k-2}$, for any $2 \le k \le 3$, satisfies

$$\|v\|_{k,\Omega} + \|q\|_{k-1,\Omega} \le c\|g\|_{k-2,\Omega}.$$

(ii) The initial velocity $u_0 \in V \cap H^2(\Omega)^2$, the solenoidal vector field $b \in L^\infty(0,T;W^{1,\infty}(\Omega)^2)$ and the prescribed force f are smooth uniformly in time, satisfying

$$\|u_0\|_{2,\Omega} + \sup_{t\in(0,T]}\left\{\|\partial_t^k f\|_{m,\Omega} + \|b\|_{1,\infty,\Omega}\right\} \le c, \quad k,m = 0,1,2.$$

(iii) There exists a constant C such that

$$\sup_{t\in(0,T]}\|\nabla u(\cdot,t)\|_{0,\Omega} \le C.$$

Lemma 1[44] Assume that assumptions (1), (2) and (3) hold. Then Eqs. (1)–(4) exist a unique solution pair $(u,p) \in (L^2(0,T;H^2(\Omega)^2)\cap C(0,T;V)\cap H^1(0,T;L^2(\Omega)^2))\times L^2(0,T;H^1(\Omega)\cap M)$. Moreover, the following bounds hold:

$$\sup_{0<t\le T}\left(\|u_t(t)\|^2_{k-2,\Omega} + \|u(t)\|^2_{k,\Omega} + \|p(t)\|^2_{k-1,\Omega}\right) \le c, \quad k = 2,3, \tag{5}$$

$$\sup_{0<t\le T}\|u_t(t)\|^2_{1,\Omega} + \int_0^T\left(\|u_{tt}(t)\|^2_{0,\Omega} + \|u_t(t)\|^2_{2,\Omega} + \|p_t(t)\|^2_{1,\Omega}\right)\mathrm{d}t \le c. \tag{6}$$

For the unsteady Oseen equations (1)–(4), we define $\mathcal{C}^n_u$ and $\mathcal{C}^m_p$ as the class of neural networks:

$$[\mathcal{C}^n_u(\psi)]^d = \left\{\Psi(t,x) : R^{d+1} \to R^d \mid \Psi_t(t,x) = \sum_{i=1}^{n}\beta_i\psi\left(\sum_{j=1}^{d}\alpha_{j,i}x_j + \alpha_{d+1,i}t_i + \gamma_i\right)\right\},$$

$$\mathcal{C}^m_p(\varphi) = \left\{\Phi`(t,x) : R^{d+1} \to R \mid \phi(t,x) = \sum_{i=1}^{m}\xi_i\varphi\left(\sum_{j=1}^{d}\sigma_{j,i}x_j + \sigma_{d+1,i}t_i + c_i\right)\right\}, \tag{7}$$

where $\iota = 1,2,\ldots,d$; φ and ψ are the shared activation functions of the hidden units in $C^2(\Omega_{\mathrm{T}})$, bounded and nonconstant; x_j is input, β_i, ξ_i, α_{ji}, and σ_{ji} are weights; γ_i and c_i are thresholds of neural network. The simplified diagram of the neural networks are displayed in Fig. 1

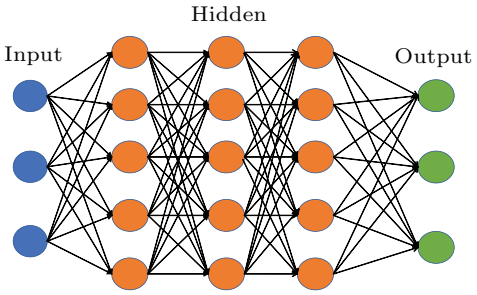


Fig. 1. The neural network architecture.

We use $[\mathcal{C}_u(\psi)]^d \times \mathcal{C}_p(\varphi)$ as the multilayer neural networks, and $[\mathcal{C}_u(\psi)]^d \times \mathcal{C}_p(\varphi) = \bigcup_{n\geq 1}[\mathcal{C}_u^n(\psi)]^d \times \bigcup_{m\geq 1}\mathcal{C}_p^m(\varphi)$. Furthermore, the parameters can be written as

$$\theta_1^l = \left(\beta_1,\ldots,\beta_n,\alpha_{1,1},\ldots,\alpha_{d,n},\alpha_{d+1,n},\gamma_1,\ldots,\gamma_n\right),$$
$$\theta_2 = \left(\xi_1,\ldots,\xi_m,\sigma_{1,1},\ldots,\sigma_{d,m},\sigma_{d+1,m},c_1,\ldots,c_m\right),$$

assuming that $(U,P) = (U(t,x;\theta_1),P(t,x;\theta_2))$ is the approximate solution to the general unsteady Oseen equations (1)–(4). Here θ_1^l and θ_2 are the stacked components of the neural network parameters θ for velocity and pressure, respectively. Define the loss function

$$\begin{aligned} J(U,P) =\ & \|\mathcal{G}[U,P](t,x;\theta)\|^2_{0,\Omega_{\mathrm{T}},\omega_1} + \|\nabla\cdot U(t,x;\theta)\|^2_{0,\Omega_{\mathrm{T}},\omega_1} \\ & + \|U(t,x;\theta)\|^2_{0,\partial\Omega_{\mathrm{T}},\omega_2} \\ & + \|U(0,x;\theta) - u(0,x)\|^2_{0,\Omega_0,\omega_3}, \end{aligned} \tag{8}$$

notice

$$\|f(y)\|^2_{0,\mathcal{Y},\omega} = \int_{\mathcal{Y}} |f(y)|^2\omega(y)\,\mathrm{d}y,$$

where $\omega(y)$ is the probability density of y in $\mathcal{Y}$. $J(U,P)$ could measure how well the function $(U(t,x;\theta_1^l),P(t,x;\theta_2))$ satisfies Eqs. (1)–(4), differential operators, boundary condition, divergence condition and initial condition.

The goal is to find parameters θ such that the approximate solution (U,P) minimizes the loss function $J(U,P)$. If $J(U,P) = 0$ then (U,P) is the solution to equations (1)–(4). However, directly minimizing $J(U,P)$ to seek θ is impracticable since the calculation is difficult. Therefore, we introduce the PINNs to update parameters of the neural networks, we collect sample points $(t_n,x_n) \in \Omega_{\mathrm{T}}$, $(\tau_n,z_n) \in \partial\Omega_{\mathrm{T}}$, and $\sigma_n \in \Omega_0$. We will follow the method stated in the following.

Algorithm 1 Overview of the method

Input: Randomly spatiotemporal points $s_n = \{(t_n,x_n),(\tau_n,z_n),\sigma_n\}$.
Output: Neural network predicted solution $u(t,x), p(t,x)$.
(1) Construct a neural network with parameters θ;
(2) Specify the training set s_n;
(3) Specify the loss function by summing the residual of PDE;
(4) Train the neural network to find the best parameters θ by minimizing the loss function $J(U,P)$ in (3);
(5) Get the predicted solution $u(t,x), p(t,x)$ on the domain;
return $u(t,x), p(t,x)$.

3. The convergence of the loss function

In this section, we ensure the existence of multilayer networks (U,P) which makes the loss function $J(U,P)$ arbitrarily small. In order to obtain the convergence of the loss function $J(U,P)$, we use the following assumption.

Lemma 2 Let the term $\Delta u(t,x)$ be locally Lipschitz in $(u(t,x),\nabla u(t,x))$ with Lipschitz constant which has at most polynomial growth on u, uniformly with respect to x. This means that

$$\begin{aligned} &|\Delta U - \Delta u| \leq (|\nabla U|^{\frac{q_1}{2}} + |\nabla u|^{\frac{q_2}{2}})|\nabla U - \nabla u|, \\ &0 \leq q_i \leq \infty, \quad i = 1,2. \end{aligned} \tag{9}$$

Theorem 1 Let $[\mathcal{C}_u(\psi)]^d$, $\mathcal{C}_p(\varphi)$ are given by Eq. (7), where ψ and φ are nonconstant and bounded. Under the assumption of Lemma 2, there exists a positive constant C and a neural network $(U,P) \in [\mathcal{C}_u(\psi)]^d \times \mathcal{C}_p(\varphi)$, such that

$$J(U,P) \leq C\varepsilon, \tag{10}$$

where $\forall\varepsilon > 0$, and C depends on $\nu, f, b, \omega_1, \omega_2, \omega_3$.

Proof From Theorem 3 of Ref. [35], we know that there is a $(U,P) \in [\mathcal{C}_u(\psi)]^d \times \mathcal{C}_p(\varphi)$ that is uniformly 2-dense on compacts of $C^2(\bar{\Omega}) \times C^1(\bar{\Omega})$. This means that for $(u,p) \in C^2(\bar{\Omega}) \times C^1(\bar{\Omega})$, $\forall\varepsilon > 0$, such that

$$\begin{aligned} &\sup_{(t,x)\in\Omega_{\mathrm{T}}} |\partial_t U(t,x;\theta_1^l) - \partial_t u(t,x)| \\ &+ \max_{|\alpha|\leq 2} \sup_{(t,x)\in\bar{\Omega}_{\mathrm{T}}} |\partial_x^{(\alpha)} U(t,x;\theta_1^l) - \partial_x^{(\alpha)} u(t,x)| < \varepsilon, \end{aligned} \tag{11}$$

$$\max_{|\alpha|\leq 1} \sup_{(t,x)\in\bar{\Omega}_{\mathrm{T}}} |P(t,x;\theta_2) - p(t,x)| < \varepsilon. \tag{12}$$

According to Lemma 2 and Hölder inequality, we can obtain

$$\begin{aligned} &\int_{\Omega_{\mathrm{T}}} |\nu(\triangle U - \triangle u)|^2\,\mathrm{d}\omega_1(t,x) \\ &\leq \nu^2 \int_{\Omega_{\mathrm{T}}} (|\nabla U|^{\frac{q_1}{2}} + |\nabla u|^{\frac{q_2}{2}})^2 |\nabla U - \nabla u|^2\,\mathrm{d}\omega_1(t,x) \\ &\leq 2\nu^2 \int_{\Omega_{\mathrm{T}}} (|\nabla U|^{q_1} + |\nabla u|^{q_2}) |\nabla U - \nabla u|^2\,\mathrm{d}\omega_1(t,x) \end{aligned}$$

$$
\begin{aligned}
&\le 2\nu^2\left[\int_{\Omega_{\mathrm{T}}}(|\nabla U|^{q_1}+|\nabla u|^{q_2})^{r_1}\,\mathrm{d}\omega_1(t,x)\right]^{\frac{1}{r_1}}\\
&\quad\times\left[\int_{\Omega_{\mathrm{T}}}|\nabla U-\nabla u|^{2r_2}\,\mathrm{d}\omega_1(t,x)\right]^{\frac{1}{r_2}}\\
&\le 2\nu^2\left[\int_{\Omega_{\mathrm{T}}}(|\nabla(U-u)|^{q_1}+|\nabla u|^{q_1\vee q_2})^{r_1}\,\mathrm{d}\omega_1(t,x)\right]^{\frac{1}{r_1}}\\
&\quad\times\left[\int_{\Omega_{\mathrm{T}}}|\nabla U-\nabla u|^{2r_2}\,\mathrm{d}\omega_1(t,x)\right]^{\frac{1}{r_2}}\\
&\le C\nu^2(\varepsilon^{q_1}+\sup|\nabla u|^{q_1\vee q_2})\varepsilon^2, \qquad (13)
\end{aligned}
$$

where $q_1\vee q_2=\max\{q_1,q_2\}$, r_1,r_2 satisfying $\frac{1}{r_1}+\frac{1}{r_2}=1$.

By applying Eqs. (11)–(13), we can reach

$$
\begin{aligned}
J(U,P) &= \|\mathcal{G}[U,P](t,x;\theta)\|^2_{0,\Omega_{\mathrm{T}},\omega_1}+\|\nabla\cdot U(t,x;\theta_1)\|^2_{0,\Omega_{\mathrm{T}},\omega_1}\\
&\quad+\|U(t,x;\theta_1)\|^2_{0,\partial\Omega_{\mathrm{T}},\omega_2}\\
&\quad+\|U(0,x;\theta_1)-u_0\|^2_{0,\Omega_0,\omega_3}\\
&= \|\mathcal{G}[U,P](t,x;\theta)-\mathcal{G}(u,p)(t,x)\|^2_{0,\Omega_{\mathrm{T}},\omega_1}\\
&\quad+\|\nabla\cdot U(t,x;\theta_1)-\nabla\cdot u(t,x)\|^2_{0,\Omega_{\mathrm{T}},\omega_1}\\
&\quad+\|U(t,x;\theta_1)-u(t,x)\|^2_{0,\partial\Omega_{\mathrm{T}},\omega_2}\\
&\quad+\|U(0,x;\theta_1)-u(0,x)\|^2_{0,\Omega_0,\omega_3}\\
&\le \int_{\Omega_{\mathrm{T}}}|U_t-u_t|^2\,\mathrm{d}\omega_1(t,x)\\
&\quad+\int_{\Omega_{\mathrm{T}}}|(b\cdot\nabla)(U-u)|^2\,\mathrm{d}\omega_1(t,x)\\
&\quad+\int_{\Omega_{\mathrm{T}}}|\nu(\triangle U-\triangle u)|^2\,\mathrm{d}\omega_1(t,x)\\
&\quad+\int_{\Omega_{\mathrm{T}}}|\nabla P-\nabla p|^2\,\mathrm{d}\omega_1(t,x)\\
&\quad+\int_{\Omega_{\mathrm{T}}}|\nabla\cdot U-\nabla\cdot u|^2\,\mathrm{d}\omega_1(t,x)\\
&\quad+\int_{\partial\Omega_{\mathrm{T}}}|U-u|^2\,\mathrm{d}\omega_2(t,x)\\
&\quad+\int_{\Omega_0}|U(0)-u(0)|^2\,\mathrm{d}\omega_3(t,x)\\
&\le C\varepsilon^2. \qquad (14)
\end{aligned}
$$

4. The convergence of the neural network

In this section, we discuss the convergence of the neural networks (U^n,P^n) to the analytical solution (u,p).

From Theorem 1, we know

$$J(U^n,P^n)\to 0.$$

Each neural network (U^n,P^n) satisfies the PDE

$$
\begin{aligned}
&\mathcal{G}[U^n,P^n](t,x)=0, \quad \text{in} \quad \Omega\times(0,T], && (15)\\
&\nabla\cdot U^n(t,x)=0, \quad \text{in} \quad \Omega\times(0,T], && (16)\\
&U^n(t,x)=0, \quad \text{on} \quad \partial\Omega\times(0,T], && (17)\\
&U^n(0,x)=u_0(x), \quad \text{on} \quad \Omega\times\{0\}. && (18)
\end{aligned}
$$

To demonstrate the results expected in this subsection, we assume:

Lemma 3 Existing a positive constant β and satisfying

$$\sup_{v\in C^2(\bar{\Omega})}\frac{\mathrm{d(v,q)}}{\|\nabla v\|_0}\ge\beta\|q\|_0,\forall q\in C^1(\bar{\Omega}).$$

Theorem 2 Under the assumptions of Lemma 1, Eqs. (15)–(18) have a unique bounded solution pair $(u,p)\in(L^2(0,T;H^2(\Omega)^2)\cap C(0,T;V)\cap H^1(0,T;L^2(\Omega)^2))\times L^2(0,T;H^1(\Omega)\cap M)$. Furthermore, the neural networks U^n converge strongly to u in $L^2(\Omega_{\mathrm{T}})$, and the P^n converges strongly to p in $L^2(\Omega_{\mathrm{T}})$. Moreover, if the sequence $\{U^n\}_{n\in N}$ and $\{P^n\}_{n\in N}$ are uniformly bounded and equicontinuous in Ω_{T}, they can uniformly converge to u and p, respectively, in Ω_{T}.

Proof By using Lemma 2, the existence and uniqueness for the solution of Eqs. (1)–(4) can be proved. We verify the boundness of (U,P) for Eqs. (1)–(4). Multiplying the above equations by $(v,q)\in[\mathcal{C}_u^n(\psi)]^d\cap[H_0^1(\Omega_{\mathrm{T}})]^d\times[\mathcal{C}_p^m(\varphi)]\cap L_0^2(\bar{\Omega}_{\mathrm{T}})$, we have

$$
\begin{aligned}
&(U_t^n,v)+\nu(\nabla U^n,\nabla v)+((b\cdot\nabla)U^n,v)\\
&+(\nabla P^n,v)=(f,v), && (19)\\
&(\nabla\cdot U^n,q)=0, && (20)
\end{aligned}
$$

since

$$
\begin{aligned}
(\nabla P^n,v) &= -(\nabla\cdot v,P^n), && (21)\\
((b\cdot\nabla)U^n,v) &= ((b\cdot\nabla)U^n,v)+\frac{1}{2}((\nabla\cdot b)U^n,v)\\
&= \frac{1}{2}((b\cdot\nabla)U^n,v)-\frac{1}{2}((b\cdot\nabla)v,U^n), && (22)
\end{aligned}
$$

noting $\nabla\cdot b=0$ and taking $(v,q)=(U^n,P^n)$, we obtain

$$(U_t^n,U^n)+\nu(\nabla U^n,\nabla U^n)=(f,U^n). \qquad (23)$$

Taking $t=0$, we obtain

$$(U_t^n(0),U^n(0))+\nu(\nabla U^n(0),\nabla U^n(0))=(f(0),U^n(0)).$$

By definition of inner product and derivative, the above equation can be changed to

$$
\begin{aligned}
&\frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}t}\|U^n(0)\|_0^2+\nu\|\nabla U^n(0)\|_0^2\\
&\le \|f(0)\|_0\|U^n(0)\|_0\\
&\le \frac{1}{2}(\|f(0)\|_0^2+\|U^n(0)\|_0^2)\\
&\le \frac{1}{2}(\|f(0)\|_0^2+C\|\nabla U^n(0)\|_0^2), \qquad (24)
\end{aligned}
$$

at the same time, integrate both sides over $t\in(0,s)$, and $s\le T$

$$
\begin{aligned}
&s\|U^n(0)\|_0^2+(2\nu-C)\int_0^s\|\nabla U^n(0)\|_0^2\,\mathrm{d}t\\
&\le C\int_0^s\|f(0)\|_0^2\,\mathrm{d}t, \qquad (25)
\end{aligned}
$$

where $(2\nu-C)>0$.

Taking the derivative of both sides with respect to $t \in (0,s)$ in Eq. (23) and taking $t=0$, we have

$$(U_{tt}^n(0), U_t^n(0)) + \nu(\nabla U_t^n(0), \nabla U_t^n(0)) = (f_t(0), U_t^n(0)).$$

By definition of inner product and derivative, the above equation can be changed to

$$\begin{aligned} &\frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}t}\|U_t^n(0)\|_0^2 + \nu\|\nabla U_t^n(0)\|_0^2 \\ \le\ & \|f_t(0)\|_0\|U_t^n(0)\|_0 \\ \le\ & \frac{1}{2}(\|f_t(0)\|_0^2 + \|U_t^n(0)\|_0^2) \\ \le\ & \frac{1}{2}(\|f_t(0)\|_0^2 + C\|\nabla\, U_t^n(0)\|_0^2), \end{aligned} \tag{26}$$

at the same time, integrate both sides over $t \in (0,s)$, and $s \le T$

$$\begin{aligned} &s\|U_t^n(0)\|_0^2 + (2\nu - C)\int_0^s \|\nabla U_t^n(0)\|_0^2 \mathrm{d}t \\ \le\ & C\int_0^s \|f_t(0)\|_0^2 \mathrm{d}t, \end{aligned} \tag{27}$$

therefore, according Eqs. (25) and (27), we know that $U^n(0)$ and $U_t^n(0)$ are bounded.

Using the similar approach, i.e., taking the derivative of both sides with respect to $t \in (0,s)$ in Eq. (23), then at the same time, integrating both sides over $t \in (0,s)$, and $s \le T$,

$$\begin{aligned} &s\|U_t^n\|_0^2 + (2\nu - C)\int_0^s \|\nabla U_t^n\|_0^2 \mathrm{d}t \\ \le\ & C\int_0^s \|f_t\|_0^2 \mathrm{d}t + s\|U_t^n(0)\|_0^2, \end{aligned} \tag{28}$$

then, U_t^n is bounded.

According Hölder inequation, Eq. (23) can be simplified to

$$\begin{aligned} \nu\|\nabla U^n\|_0^2 \ &\le\ \|f\|_0\|U^n\|_0 - \|U_t^n\|_0\|U^n\|_0 \\ &\le\ \frac{1}{2}(\|f\|_0^2 + C\|U^n\|_1^2 + \|U_t^n\|_0^2), \end{aligned} \tag{29}$$

because of the Poincaré inequation, we can reach

$$(2\nu - C)\|U^n\|_1^2 \le \|f\|_0^2 + \|U_t^n\|_0^2, \quad 2\nu - C > 0. \tag{30}$$

Up to now, we obtain U^n is uniformly bounded, so we can extract a subsequence $\{U^n\}_{n\in N}$ of U^n, which can converge weakly in $H^1(\Omega_\mathrm{T})$. Due to the compact embedding $H^1(\Omega_\mathrm{T}) \hookrightarrow L^2(\Omega_\mathrm{T})$, we have

$$\lim_{n\to\infty}\|U^n - u\|_0 = 0.$$

Thus, $\{U^n\}_{n\in N}$ can converge strongly to u in $L^2(\Omega_\mathrm{T})$.

Next, let us discuss the boundedness of pressure P^n. Firstly, we utilize Lemma 3 and Eqs. (19)–(22) to obtain

$$\begin{aligned} &\beta\|P^n\|_0 \\ \le\ & \sup_{v\in C^2(\bar{\Omega})\cap C_p^m(\varphi)} \frac{-(\nabla P^n, v)}{\|v\|_0} \\ \le\ & \sup_{v\in C^2(\bar{\Omega}_\mathrm{T})\cap C_p^m(\varphi)} \frac{(U_t^n, v) + \nu(\nabla U^n, \nabla v) - (f, v)}{\|v\|_0} \\ \le\ & \sup_{v\in C^2(\bar{\Omega}_\mathrm{T})\cap C_p^m(\varphi)} \frac{\|U_t^n\|_0\|v\|_0 + \nu\|\nabla U^n\|_0\|\nabla v\|_0 - \|f\|_0\|v\|_0}{\|v\|_0} \\ \le\ & \sup_{v\in C^2(\bar{\Omega}_\mathrm{T})\cap C_p^m(\varphi)} \frac{\|U_t^n\|_0\|v\|_0 + \nu\|\nabla U^n\|_0\|\nabla v\|_0 + \|f\|_0\|v\|_0}{\|v\|_0} \\ \le\ & \|U_t^n\|_0 + \nu\|\nabla U^n\|_0 + \|f\|_0, \end{aligned} \tag{31}$$

so we verify the boundedness of P^n, which implies that P^n can converge weakly to p. Using the same method as for U^n converge strongly to u, we obtain

$$\lim_{n\to\infty}\|P^n - p\|_0 = 0,$$

so P^n can converges strongly to p.

If U^n and P^n are equicontinuous and uniformly bounded, using the Arzelà–Ascoli theorem, we can conclude that U^n and P^n uniformly converge to u and p in Ω, respectively.

5. Numerical examples

In this section, we will verify the effectiveness of neural networks in solving unsteady Oseen equations by three examples. In the numerical experiment, the configuration of the computer is 64-bit Intel(R) Core(TM) i7 − 8700 CPU @3.20 GHz 3.19 GHz, the GPU of computer is Inter(R) UHD Graphics 630, 8GB memory.

5.1. Analytic solution

In order to estimate the performance of this method, we solve Eqs. (1)–(4) by the proposed algorithm. The analytic solutions to Eqs. (1)–(4) are as follows:

$$\begin{aligned} u_1 &= x^2(x-1)^2y(y-1)(2y-1)\,\mathrm{e}^{-t}, \\ u_2 &= -y^2(y-1)^2x(x-1)(2x-1)\,\mathrm{e}^{-t}, \\ p &= (2x-1)(2y-1)\,\mathrm{e}^{-t}, \\ b_1 &= \sin(x)\sin(y), \\ b_2 &= \cos(x)\cos(y), \end{aligned}$$

where Ω is the unit square $[0,1]\times[0,1] \subset R^2$ and $\nu = 1$. Then, the right hand f can be calculated by Eq. (1). We compute the relative L^1, L^2 errors to check the performance of the neural network:

$$\mathrm{err}L^1 = \frac{\|U-u\|_{L^1}}{\|u\|_{L^1}}, \quad \mathrm{err}L^2 = \frac{\|U-u\|_{L^2}}{\|u\|_{L^2}},$$

where U and u denote neural network solution and analytical solution, respectively. The formula of calculating pressure is similar to that of velocity, so we only give the relative error formulas of velocity.

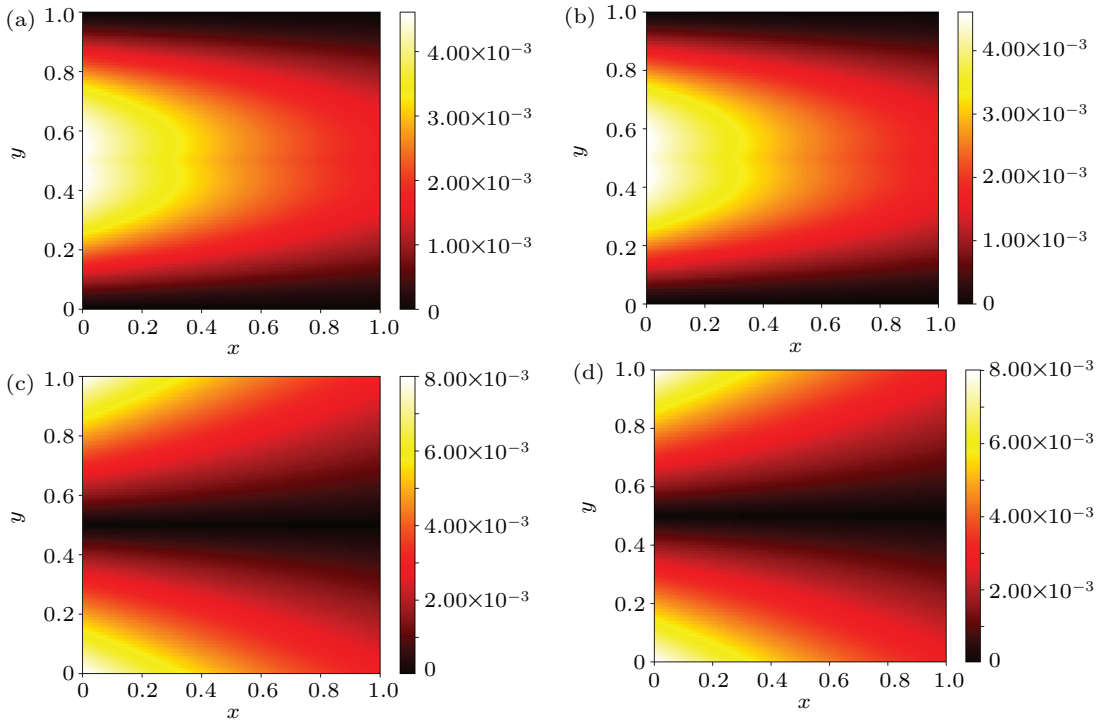


Fig. 2. Neural network solution and analytical solution of velocity and pressure: (a) the analytical solution of velocity, (b) the neural network solution of velocity, (c) the analytical solution of pressure, (d) the neural network solution of pressure.

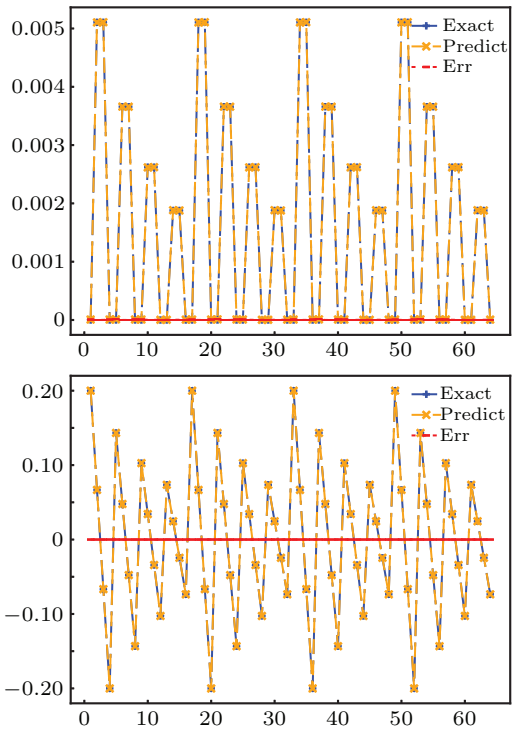


Fig. 3. Neural network solution and analytical solution of velocity and pressure: (a) pointwise error diagram of velocity, (b) pointwise error diagram of pressure.

We train the neural network by optimizing the loss function. According to the numerical experimental results, we find that the neural network satisfies the requirements of the solutions, the relative errors between the neural network solution and the analytical solution can reach 10^{-3}. The comparison between the analytical solution and neural network solutions of velocity as well as pressure are shown in Fig. 2 at $t = 0.89$, and the relative L^1, L^2 errors are 2.25×10^{-3}, 2.95×10^{-3}, respectively. Figure 3 illustrates the relative L^1 pointwise error of the velocity and pressure. Based on these results, we conclude that the neural networks solution fits to the analytical solution perfectly. Therefore, we validate the feasibility of the proposed method to solve Eqs. (1)–(4).

5.2. Influence of parameter

In this subsection, we discuss the influence of parameters on the performance of neural networks. we first study whether the accuracy of the neural network is related to the activation functions. The values of the relative errors and loss function $J(U,P)$ corresponding to each activation function are listed in Table 1.

Table 1. The neural network with different activation functions.

Activation functions	Sigmoid	tanh	Softmax	Relu
Hidden layers	2	2	2	2
Relative L^1	2.56×10^{-3}	3.08×10^{-3}	2.7×10^{-2}	2.3×10^{1}
Relative L^2	1.79×10^{-3}	3.44×10^{-3}	2.9×10^{-2}	3.7×10^{1}
$J(U,P)$	3.5×10^{-4}	1.0×10^{-5}	7.6×10^{-4}	5.7×10^{-1}
Hidden layers	3	3	3	3
Relative L^1	3.82×10^{-3}	1.05×10^{-4}	3.7×10^{-2}	1.5×10^{1}
Relative L^2	5.73×10^{-3}	8.94×10^{-4}	4.3×10^{-2}	1.9×10^{1}
$J(U,P)$	9.0×10^{-6}	8.1×10^{-6}	2.3×10^{-4}	5.5×10^{-1}
Hidden layers	4	4	4	4
Relative L^1	2.45×10^{-3}	2.35×10^{-4}	1.41×10^{-3}	7.31×10^{1}
Relative L^2	2.03×10^{-3}	2.17×10^{-4}	1.29×10^{-3}	7.32×10^{1}
$J(U,P)$	2.2×10^{-5}	1.1×10^{-6}	9.2×10^{-5}	5.1×10^{-1}

It could be seen that under certain conditions, choosing tanh as the activation function can obtain the neural network

with better performance. Therefore, we employ tanh as the activation function of the following experiments.

Next we discuss whether the accuracy of the neural networks is related to the number of the sample points, the depth and the width of neural networks. Figure 4 displays the relative error results with different numbers of sample points (3 hidden layers and 16 neurons per hidden layer).

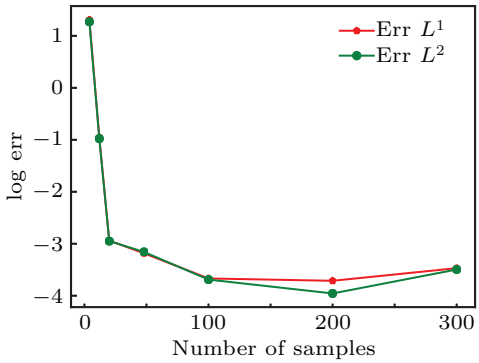


Fig. 4. The neural network with different samples.

We observe that the relative error drops slowly if the number of sample points exceeds 20. Although 200 randomly sample points show the best performance in Fig. 4, we generally take 20 random sample points in order to meet both accuracy and efficiency. Based on this conclusion, Fig. 5 shows the relative errors using different widths and depths of the neural networks. In Fig. 5(a), the results show a downward trend when the number of neurons is less than 32 (3 hidden layers). Figure 5(b) clearly shows that the neural network solution can maintain being stable when the number of hidden layers reaches a certain value (16 neurons per hidden layer).

Under certain conditions, the performance of the neural network is optimized as the number of hidden layers and neurons increase. However, we need to consider both accuracy and efficiency in practical applications. Therefore, we finally choose 3 hidden layers and 16 neurons per hidden layer as the optimal parameter for Eqs. (1)–(4).

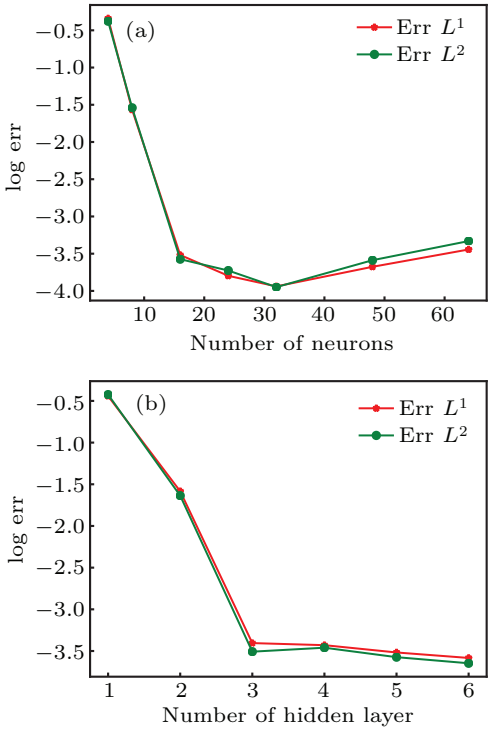


Fig. 5. Neural network solution and analytical solution of velocity and pressure: (a) the neural network with different neurons, (b) the neural network with different hidden layers.

Meanwhile, we also discuss whether the accuracy of the neural networks is related to the randomly seeds. From Table 2, we can observe that the numerical results with different randomly seeds are less distinguishable. Namely, the neural network solutions produced by different randomly seeds are in good agreement with the analytic solutions.

Table 2. The neural network with different randomly seeds.

Randomly seed	42	43	44	45	46
Relative L^1	2.37×10^{-4}	2.35×10^{-4}	5.26×10^{-4}	6.12×10^{-4}	1.02×10^{-4}
Relative L^2	5.26×10^{-4}	2.86×10^{-4}	4.43×10^{-4}	6.87×10^{-4}	1.54×10^{-4}
$J(U,P)$	4.28×10^{-6}	3.13×10^{-6}	1.27×10^{-6}	2.88×10^{-6}	6.21×10^{-6}
Randomly seed	47	48	49	50	51
Relative L^1	4.64×10^{-4}	1.79×10^{-4}	5.71×10^{-4}	3.86×10^{-4}	4.79×10^{-4}
Relative L^2	4.78×10^{-4}	2.17×10^{-4}	6.58×10^{-4}	4.36×10^{-4}	5.09×10^{-4}
$J(U,P)$	2.67×10^{-6}	2.61×10^{-6}	2.82×10^{-6}	4.27×10^{-6}	1.52×10^{-6}

Table 3. The neural network with different kinematic viscosity (ν).

$\log_{10}\nu$	–4	–3	–2	–1
Relative L^1	3.08×10^{-4}	3.88×10^{-4}	4.24×10^{-4}	5.31×10^{-4}
Relative L^2	2.86×10^{-4}	3.79×10^{-4}	5.36×10^{-4}	5.59×10^{-4}
$J(U,P)$	1.27×10^{-6}	2.32×10^{-6}	2.73×10^{-6}	3.27×10^{-6}
$\lg\nu$	1	2	3	4
Relative L^1	5.69×10^{-4}	2.47×10^{-3}	1.82×10^{-2}	1.64×10^{-1}
Relative L^2	6.97×10^{-4}	2.40×10^{-3}	1.99×10^{-2}	1.97×10^{-1}
$J(U,P)$	5.01×10^{-6}	5.27×10^{-6}	6.91×10^{-4}	9.92×10^{-3}

Further, we discuss whether the accuracy of the neural networks is related to parameters of the equations. All the results are listed in Table 3. Although the relative error is impacted on the value of ν, it can maintain a certain accuracy. Noticeably, it implies that this method is effective for solving this kind of equations.

5.3. Lid-driven cavity flow

In this subsection, we consider the 2D incompressible lid-driven cavity flow problem in a unit square, the specified ve-

locity is 1 in the x direction of the upper boundary of the region and in other directions the components of velocity are zero. Additionally, we also study the 3D incompressible lid-driven cavity flow in a unit volume. At the top surface, the tangential velocity is set 1 in the x direction, while 0 velocity is set on the remaining surface. We prescribe $\nu = 1$, $f = 0$, apply 100 randomly sample points, 3 hidden layers and 16 neurons per hidden layer. Figure 6 shows the neural network solutions at three independent time steps: $t = 0.33$, 0.66, 0.99 in the 2D case and compares the solutions of the Freefem equation. Similarly, the results in the 3D case are exhibited in Fig. 7. It shows the velocity profile solved by neural network. Tables 4 and 5 show the GPU time cost of 2D and 3D lid-driven cavity flow problems with different hidden layers, respectively.

Table 4. Computational efficiency of 2D lid-driven cavity flow with different hidden layers.

Hidden layer	1	2	3	4
GPU (s)	56.07	199.17	291.88	659.66
Iterations	20000	20000	20000	20000

Table 5. Computational efficiency of 3D Lid-driven cavity flow with different hidden layers.

Hidden layer	1	2	3	4
GPU (s)	349.95	289.88	550.35	784.15
Iterations	20000	20000	20000	20000

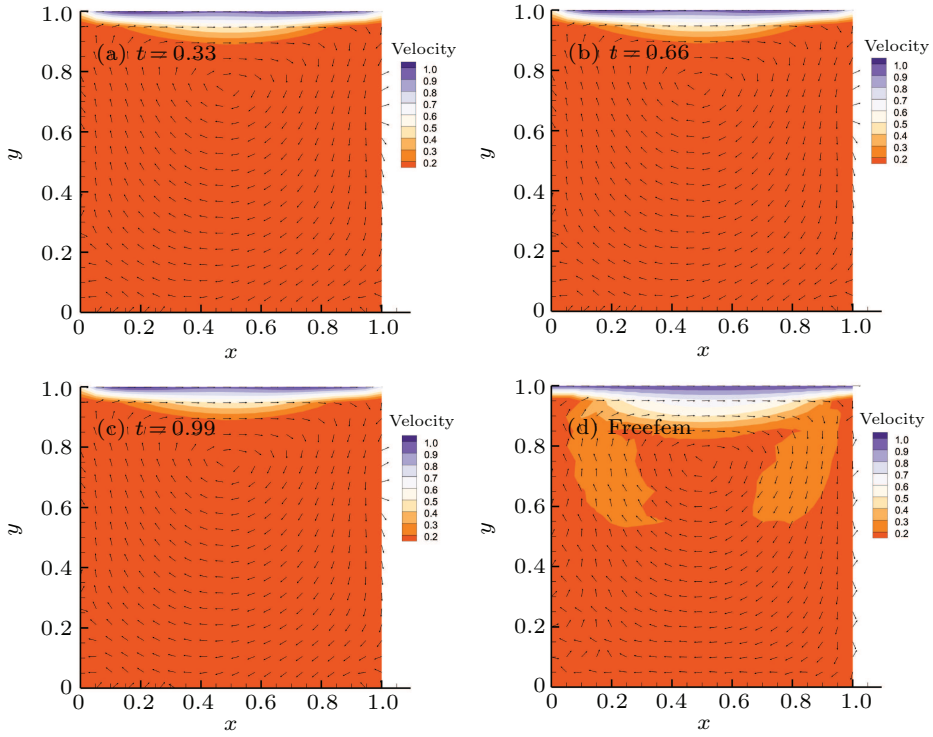


Fig. 6. The 2D lid-driven cavity flow.

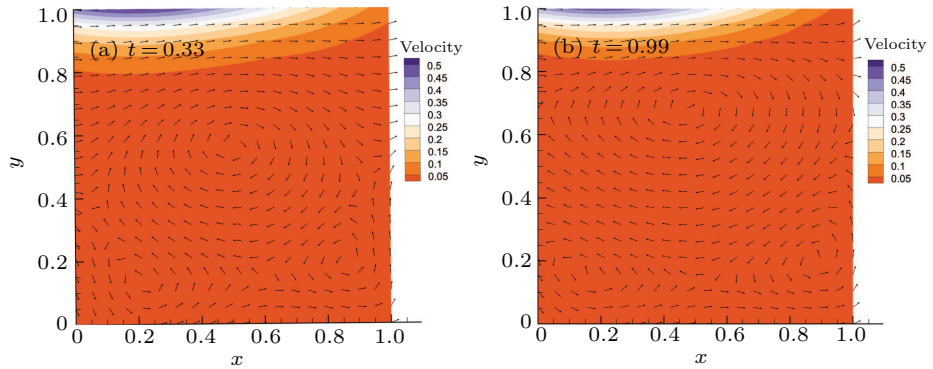


Fig. 7. The 3D lid-driven cavity flow.

Acknowledgements

Project supported in part by the National Natural Science Foundation of China (Grant No. 11771259), Shaanxi Provincial Joint Laboratory of Artificial Intelligence (Grant No. 2022JCSYS05), Innovative Team Project of Shaanxi Provincial Department of Education (Grant No. 21JP013), and Shaanxi Provincial Social Science Fund Annual Project (Grant No. 2022D332).

References

[1] Liu X, Li J and Chen Z 2018 *J. Comput. Appl. Math* **333** 442
[2] Li J, He Y and Xu H 2007 *Comput. Methods Appl. Mech. Eng.* **196** 2852
[3] Li J 2019 *Numerical methods for the incompressible Navier–Stokes equations* (Beijing: Science Press) p. 72
[4] Li J, Bai Y and Zhao X 2022 *Modern numerical methods for mathematical physics equations* (Beijing: Science Press) p. 56
[5] Li J, Lin X and Chen Z 2021 *Finite volume method for the incompressible Navier–Stokes equations* (Berlin: Springer) p. 45
[6] Li R, Gao Y, Li J and Chen Z 2018 *J. Comput. Appl. Math.* **334** 111
[7] He Y and Li J 2011 *J. Comput. Phys.* **231** 6790
[8] Li J and Chen Z 2013 *Adv. Comput. Math.* **38** 281
[9] Zhu L, Li J and Chen Z 2011 *J. Comput. Appl. Math.* **235** 2821
[10] Braack E, Burman E and John V 2007 *Comput. Methods Appl. Mech. Eng.* **196** 853
[11] Ding Q and Shang Y 2021 *Adv. Appl. Math. Mech.* **13** 1501
[12] Liu X, Li J and Chen Z 2016 *Adv. Comput. Math.* **42** 1473
[13] Winter M, Schott B and Massing A 2018 *Comput. Methods Appl. Mech. Eng.* **330** 220
[14] Xu C, Shi D and Liao X 2018 *Appl. Math. Mech.* **39** 291
[15] Massing A, Schott B and Wall W A 2018 *Comput. Methods Appl. Mech. Eng.* **328** 262
[16] Berg J and Nyström K 2018 *Neurocomputing* **317** 28
[17] Goswami S, Anitescu C, Chakraborty S and Rabczuk T 2019 arXiv:1907.02531 [stat.ML]
[18] Haghighat E, Raissi M, Moure A, Gomez H and Juanes R 2020 arXiv:2003.02751 [cs.LG]
[19] He Q, Barajas-Solano D, Tartakovsky G and Tartakovsky A M 2020 *Adv. Water. Resour.* **141** 103610)
[20] Tchelepi H A and Fuks O 2020 *Journal of Machine Learning for Modeling and Computing* **1** 19
[21] Sirignano J and Spiliopoulos K 2018 *J. Comput. Phys.* **375** 1339
[22] Raissi M, Perdikaris P and Karniadakis G E 2017 arXiv:1711.10561 [cs.AI]
[23] Raissi M, Perdikaris P and Karniadakis G E 2017 arXiv:1711.10566 [cs.AI]
[24] Berg J and Kaj N 2018 *Neurocomputing* **317** 28
[25] Lagaris I E, Likas A and Fotiadis D I 1998 *IEEE Trans. Neural Networks* **9** 987
[26] Lagaris I E, Likas A and Papageorgiou D G 2000 *IEEE Trans. Neural Networks* **11** 1041
[27] Lee H and Kang I S 1990 *J. Comput. Phys.* **91** 110
[28] Malek A and Shekari B R 2006 *Appl. Math. Comput.* **183** 260
[29] Rudd K 2013 *Solving partial differential equations using artificial neural networks* (Ph.D. Dessertation, Duke University)
[30] Ling J, Kurzawski A and Templeton J 2016 *J. Fluid. Mech.* **807** 155
[31] Tompson J, Schlachter K and Sprechmann P 2017 *International Conference on Machine Learning (PMLR)* **70** 3424
[32] Pratik C, Adam O and Stanley O 2018 *Res. Math. Sci.* **5** 30
[33] Raissi M, Perdikaris P and Karniadakis G E 2019 *J. Comput. Phys.* **378** 686
[34] Yang M and Foster J T 2022 arXiv:2202.01710 [cs.CE]
[35] Pang G, Lu L and Karniadakis G E 2019 *SIAM J. Sci. Comput.* **41** A2603
[36] Yang L, Meng X and Karniadakis G E 2021 *J. Comput. Phys.* **425** 109913
[37] Kharazmi E, Zhang Z and Karniadakis G E 2021 *Comput. Methods Appl. Mech. Eng.* **374** 113547
[38] Meng X, Li Z and Zhang D 2020 *Comput. Methods Appl. Mech. Eng.* **370** 113250
[39] Shukla K, Jagtap A D and Karniadakis G E 2021 *J. Comput. Phys.* **447** 110683
[40] Lu L, Meng X and Mao Z 2021 *SIAM Rev.* **63** 208
[41] Cai S, Wang Z and Wang S 2021 *J. Heat Transfer* **143** 1083
[42] Wang R, Utiyama M and Finch A 2018 *IEEE-ACM Trans. Audio, Speech, and Language Processing* **26** 1727
[43] Valan M, Makonyi K and Maki A 2019 *Syst. Biol.* **68** 876
[44] Li Y and Mei F 2021 *Multimedia Tools Appl.* **80** 17391