Journal of Scientific Computing (2024) 101:77
https://doi.org/10.1007/s10915-024-02711-1

Deep Adaptive Sampling for Surrogate Modeling Without Labeled Data

Xili Wang [1] · Kejun Tang [2] · Jiayu Zhai [3] · Xiaoliang Wan [4] · Chao Yang [5]

Received: 9 April 2024 / Revised: 14 October 2024 / Accepted: 21 October 2024

Abstract
Surrogate modeling is of great practical significance for parametric differential equation systems. In contrast to classical numerical methods, using physics-informed deep learning-based methods to construct simulators for such systems is a promising direction due to its potential to handle high dimensionality, which requires minimizing a loss over a training set of random samples. However, the random samples introduce statistical errors, which may become the dominant errors for the approximation of low-regularity and high-dimensional problems. In this work, we present a deep adaptive sampling method for surrogate modeling of low-regularity parametric differential equations and illustrate the necessity of adaptive sampling for constructing surrogate models. In the parametric setting, the residual loss function can be regarded as an unnormalized probability density function (PDF) of the spatial and parametric variables. In contrast to the non-parametric setting, factorized joint density models can be employed to alleviate the difficulties induced by the parametric space. The PDF is approximated by a deep generative model, from which new samples are generated and added to the training set. Since the new samples match the residual-induced distribution, the refined training set can further reduce the statistical error in the current approximate solution

✉ Kejun Tang
tangkejun@icode.pku.edu.cn

Xili Wang
xiliwang@stu.pku.edu.cn

Jiayu Zhai
zhaijy@shanghaitech.edu.cn

Xiaoliang Wan
xlwan@lsu.edu

Chao Yang
chao_yang@pku.edu.cn

1 School of Mathematical Sciences, Peking University, Beijing, China

2 PKU-Changsha Institute for Computing and Digital Economy, Changsha, China

3 Institute of Mathematical Sciences, ShanghaiTech University, Shanghai, China

4 Department of Mathematics and Center for Computation and Technology, Louisiana State University, Baton Rouge, USA

5 School of Mathematical Sciences and PKU-Changsha Institute for Computing and Digital Economy, Peking University, Beijing, China

Published online: 08 November 2024

through variance reduction. We demonstrate the effectiveness of the proposed method with a series of numerical experiments, including the physics-informed operator learning problem, the parametric optimal control problem with geometrical parametrization, and the parametric lid-driven 2D cavity flow problem with a continuous range of Reynolds numbers from 100 to 3200.

Keywords Surrogate modeling · Deep learning · Deep generative models · Deep adaptive sampling · Uncertainty quantification

Mathematics Subject Classification 65N22 · 65N75 · 65D40

1 Introduction

Solving differential equations with different parametric settings is widely found in uncertainty quantification [76–78], inverse design [26, 31], Bayesian inverse problems [11, 19, 39, 43, 58, 75], digital twins [6, 34, 65], parametric optimal control [79], and shape optimization [70], etc. The computational cost of solving such parametric differential equations with conventional numerical methods is expensive because repeated simulations (i.e., *many-query*) of differential equations are required. To handle such many-query problems, one may construct a surrogate model that can efficiently predict the parametric solution without sacrificing much accuracy, which is sufficient for many engineering applications. For instance, reduced order models (ROM) [5, 9, 18, 52] are widely used in practice, where the approximate solution is expressed as a linear combination of some bases that are computed by low-rank approximation of *snapshot matrices*. ROM becomes inefficient if the parametric solution does not lie in a low-dimensional linear subspace [4, 7, 10].

Deep learning-based methods for surrogate modeling have been proposed to give an alternative approach. One straightforward data-driven approach is to utilize deep neural networks to learn a mapping from a parameterized function space to the solution space [41, 45, 82], where simulation-based input–output pair data are used to train the deep neural networks. Constructing surrogate models without labeled data is necessary to handle the cases where simulation or experimental data are scarce or not available. To this end, numerical strategies have been developed for the neural network approximation of deterministic partial differential equations (PDEs) [17, 28, 29, 36, 47, 53, 57], based on which parametric PDEs can also be addressed. For example, physics-informed deep learning is used to construct surrogate models for efficient uncertainty quantification [21, 60, 83]. Depending on the formulation of parametric (partial or ordinary) differential equations, two types of neural network models can be considered: one is a plain neural network whose inputs include both the spatial and parametric variables, and the other one is defined by an operator learning problem, e.g., DeepONet [45, 68]. Regardless of the structure of the surrogate model, the underlying training procedure is similar, which minimizes a loss functional discretized by the random collocation points in the physical domain and the parametric space. For low-regularity deterministic problems, the collocation points significantly affect the generalization error of neural networks [62, 63]. This issue becomes worse for surrogate modeling because the parametric space may also introduce low regularity other than the additional dimensions. For example, in the Navier–Stokes equations, large Reynolds numbers cause some small-scale structures. To capture these features, the distribution of the collocation points in the training set must be consistent with the characteristics of the velocity field, and such a correspondence needs

to be maintained for all Reynolds numbers considered as the inputs of a surrogate model. Hence, we must pay particular attention to the random collocation points in the training set to obtain a sufficiently accurate surrogate model.

In this work, we develop a deep adaptive sampling approach for surrogate modeling (DAS^2) without labeled data, which incorporates the adaptive sampling strategy into the physics-informed surrogate models. We generalize DAS [63] to more complicated parametric settings and use a series of numerical experiments to demonstrate the importance of adaptive sampling in constructing surrogate models. Without losing generality, neural networks with augmented parameter inputs are used to approximate the parametric solutions. We intend to find a certain set of collocation points that results in a relatively flat residual profile. Since a flat residual profile has a small variance, the statistical error in the discretization of the loss functional can be significantly reduced for a fixed number of samples, which eventually improves the accuracy of the approximate solution. The desired training set is achieved through iterations. Assume that the surrogate model is trained with respect to a certain training set. The total residual of the parametric equations is viewed as an unnormalized probability density function (PDF). More samples will be introduced to the training set in the region of high density such that the residual over there can be reduced. To achieve this, a deep generative model is trained to approximate the residual-induced PDF, and new samples are drawn from this trained deep generative model. Once the training set is updated, the surrogate model will be further trained, after which the aforementioned procedure is repeated. The same algorithm can be applied to other types of surrogate models such as DeepONet. The main contributions of this work are summarized as follows.

1.1 Main Contributions

- We propose a deep adaptive sampling approach for surrogate modeling of parametric differential equations without labeled data.
- We demonstrate the efficiency of the proposed method with a series of numerical experiments, including the physics-informed operator learning problem, the parametric optimal control problem with geometrical parametrization, and the lid-driven 2D cavity flow problem with a continuous range of Reynolds numbers from 100 to 3200.

1.2 Related Work

The adaptive sampling-based neural network methods for solving deterministic differential equations are under active development. Nevertheless, adaptive sampling of parametric differential equations is still to be studied. We summarize the most related lines of this work: adaptive sampling methods for deterministic problems and neural network methods for parametric differential equations.

1.2.1 Adaptive Sampling Methods

Solving (partial) differential equations with deep learning methods usually needs a large set of collocation points, particularly when the solution has subtle structures such as high frequency, high-density concentration, multiscale structure, or discontinuity [20, 54, 69, 81]. Adaptive collocation points may significantly reduce the computational cost, where the essence is to define a proper error indicator and generate training collocation points accordingly.

The residual-based adaptive refinement (RAR) method [46, 74] is proposed to enhance the performance of physics-informed machine learning. In RAR, one needs to construct a set of uniform samples as a candidate set, within which the samples associated with large residuals are selected and added to the current training set. However, such a strategy is not effective for high-dimensional problems since most of the volume of the computational domain concentrates around its surface [73]. To obtain true samples from the residual-induced distribution, classical sampling methods such as MCMC can be employed [22, 71, 80], which, however, are also affected by the curse of dimensionality. To handle high-dimensional problems, we need to introduce other techniques. In [63], the deep adaptive sampling (DAS) method is proposed, where a normalizing flow model is used to approximate the residual-induced distribution, based on which new collocation points are generated to further improve the accuracy of the current approximate solution. DAS uses the current residual as an explicit guidance for the selection of new collocation points, which is similar to the procedure of classical adaptive methods such as the adaptive finite element method. Another track is to implicitly search for a distribution that generates collocation points that result in a smooth residual profile. In [64], an adversarial adaptive sampling framework (AAS) is proposed to seek an optimal model for the solution and an optimal distribution for the training set at the same time through a min-max formulation, which can be regarded as a generalization of the strategies that aim to find a better weight for each fixed sample [2, 27]. To reduce the training cost from the deep generative model, one can replace the deep generative model in DAS or AAS with other density models, such as Gaussian mixture models [32]. In [12, 23, 24], the authors reformulate the adaptive sampling procedure as a failure event subject to a threshold that helps determine where new collocation points are needed. Other related works include [8, 30, 51, 59].

1.2.2 Neural Network Methods for Parametric Differential Equations

The study of parametric PDEs with neural network methods started from the very beginning of PINNs in [53], where neural network provides a general model for both forward and inverse problems. We focus on parametric forward problems in this work, which can also be regarded as an operator learning problem. In [45], DeepONet is proposed, which formulates an operator that maps infinite-dimensional data, e.g., boundary and initial conditions, to the solution functions of parametric differential equations. Another typical operator model is the Fourier Neural Operaotr (FNO) [40, 41]. Physics-informed operator learning has been developed in [42, 68] to reduce the dependence of DeepONet and FNO on data in the training process.

2 Problem Setting and Statistical Errors in Physics-Informed Surrogate Modeling

Let Ω_s be a spatial domain (in $\mathbb{R}^n$) which is bounded, connected and with a smooth boundary $\partial\Omega_s$, and $\boldsymbol{x} \in \Omega_s$ is a spatial variable. Let $\boldsymbol{\xi}$ be a vector that collects a finite number of parameters. The dimension of $\boldsymbol{\xi}$ is denoted by d, i.e., we write $\boldsymbol{\xi} = [\xi_1, \ldots, \xi_d]^\mathsf{T}$. We restrict our attention to the situation that $\boldsymbol{\xi}$ has a bounded and connected support. Without loss of generality, we next assume the support of $\boldsymbol{\xi}$ to be Ω_p where $\Omega_p := [C_1, C_2]^d$ and C_1, C_2 are two constants. The physics of problems considered here are governed by differential equations over the spatial domain Ω_s and boundary conditions on the boundary $\partial\Omega_s$. Consider the following parametric differential equations: find $u : \Omega_s \times \Omega_p \mapsto \mathbb{R}$ such that

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\xi}; u(\boldsymbol{x}, \boldsymbol{\xi})) = s(\boldsymbol{x}, \boldsymbol{\xi}) \qquad \forall (\boldsymbol{x}, \boldsymbol{\xi}) \in \Omega_s \times \Omega_p, \tag{1}$$

$$\mathcal{B}(\boldsymbol{x}, \boldsymbol{\xi}; u(\boldsymbol{x}, \boldsymbol{\xi})) = g(\boldsymbol{x}, \boldsymbol{\xi}) \qquad \forall (\boldsymbol{x}, \boldsymbol{\xi}) \in \partial\Omega_s \times \Omega_p, \tag{2}$$

where $\mathcal{L}$ is a differential operator and $\mathcal{B}$ is a boundary operator, both of which can involve parameters. s is the source function and g specifies the boundary conditions. To simplify the notation, we denote $\Omega = \Omega_s \times \Omega_p$ and $\partial\Omega = \partial\Omega_s \times \Omega_p$. The goal of this study is to construct a surrogate model, which is the parametric solution to the parametric differential equation. Once this surrogate is constructed, the solution $u(\boldsymbol{x}, \boldsymbol{\xi})$ can be efficiently predicted for any $\boldsymbol{\xi}$ without solving the (partial or ordinary) differential equation repeatedly.

The framework of physics-informed surrogate modeling for parametric differential equations is as follows. Let $u_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{\xi})$ be a neural network parameterized with $\boldsymbol{\theta}$, where the input of the neural network is the tuple $(\boldsymbol{x}, \boldsymbol{\xi})$. One can use $u_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{\xi})$ to approximate $u(\boldsymbol{x}, \boldsymbol{\xi})$ through minimizing the following loss functional

$$\begin{aligned} &J(u_{\boldsymbol{\theta}}) = J_r(u_{\boldsymbol{\theta}}) + \gamma J_b(u_{\boldsymbol{\theta}}) \quad \text{with} \\ &J_r(u_{\boldsymbol{\theta}}) = \int_{\Omega} |r(\boldsymbol{x}, \boldsymbol{\xi}; \boldsymbol{\theta})|^2 d\boldsymbol{x} d\boldsymbol{\xi} \;\text{ and }\; J_b(u_{\boldsymbol{\theta}}) = \int_{\partial\Omega} |b(\boldsymbol{x}, \boldsymbol{\xi}; \boldsymbol{\theta})|^2 d\boldsymbol{x} d\boldsymbol{\xi}, \end{aligned} \tag{3}$$

where $r(\boldsymbol{x}, \boldsymbol{\xi}; \boldsymbol{\theta}) = \mathcal{L}u_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{\xi}) - s(\boldsymbol{x}, \boldsymbol{\xi})$, and $b(\boldsymbol{x}, \boldsymbol{\xi}; \boldsymbol{\theta}) = \mathcal{B}u_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{\xi}) - g(\boldsymbol{x}, \boldsymbol{\xi})$ are the residuals that measure how well $u_{\boldsymbol{\theta}}$ satisfies the parametric differential equations and the boundary conditions, respectively, and $\gamma > 0$ is a penalty parameter. Before optimizing this loss functional with respect to $\boldsymbol{\theta}$, we need to discretize numerically the integrals defined in (3), which is often achieved with uniformly distributed collocation points. Let $\mathsf{S}_{\Omega} = \{\boldsymbol{x}_{\Omega}^{(i)}, \boldsymbol{\xi}^{(i)}\}_{i=1}^{N_r}$ and $\mathsf{S}_{\partial\Omega} = \{\boldsymbol{x}_{\partial\Omega}^{(i)}, \boldsymbol{\xi}^{(i)}\}_{i=1}^{N_b}$ be two sets of uniformly distributed collocation points in Ω and $\partial\Omega$ respectively. We minimize the following empirical loss in practice

$$J_N(u_{\boldsymbol{\theta}}) = J_{r,N} + \gamma J_{b,N} = \frac{1}{N_r} \sum_{i=1}^{N_r} r^2(\boldsymbol{x}_{\Omega}^{(i)}, \boldsymbol{\xi}^{(i)}; \boldsymbol{\theta}) + \gamma \frac{1}{N_b} \sum_{i=1}^{N_b} b^2(\boldsymbol{x}_{\partial\Omega}^{(i)}, \boldsymbol{\xi}^{(i)}; \boldsymbol{\theta}), \tag{4}$$

where $J_{r,N}$ and $J_{b,N}$ can be regarded as the Monte Carlo (MC) approximations of $J_r(u_{\boldsymbol{\theta}})$ and $J_b(u_{\boldsymbol{\theta}})$ subject to statistical errors of $O(N_r^{-1/2})$ and $O(N_b^{-1/2})$ respectively. Let $u_{\boldsymbol{\theta}_N^*}$ be the minimizer of the empirical loss $J_N(u_{\boldsymbol{\theta}})$

$$u_{\boldsymbol{\theta}_N^*} = \arg\min_{\boldsymbol{\theta}} J_N(u_{\boldsymbol{\theta}}) \tag{5}$$

and $u_{\boldsymbol{\theta}^*}$ be the minimizer of the original loss functional $J(u_{\boldsymbol{\theta}})$

$$u_{\boldsymbol{\theta}^*} = \arg\min_{\boldsymbol{\theta}} J(u_{\boldsymbol{\theta}}). \tag{6}$$

We can decompose the error of $u_{\boldsymbol{\theta}_N^*}$ into two parts as follows

$$\mathbb{E}\left(\left\|u_{\boldsymbol{\theta}_N^*} - u\right\|_{\Omega}\right) \leq \mathbb{E}\left(\left\|u_{\boldsymbol{\theta}_N^*} - u_{\boldsymbol{\theta}^*}\right\|_{\Omega}\right) + \|u_{\boldsymbol{\theta}^*} - u\|_{\Omega},$$

where $\mathbb{E}$ denotes the expectation with respect to the random samples and the norm $\|\cdot\|_{\Omega}$ corresponds to a proper function space for u. Without taking into account the optimization error, one can see that the total error of neural network approximation for parametric differential equations mainly consists of two parts: the approximation error and the statistical error. The approximation error is dependent on the model capability of neural networks, while the statistical error originates from the random collocation points in the training set.

It has been noticed that uniformly distributed collocation points are not effective for training neural network models if the solution has low regularity [62, 63, 74] since the effective

sample size of the Monte Carlo approximation of $J(u_{\boldsymbol{\theta}})$ is significantly reduced by the large variance induced by the low regularity. For high-dimensional problems, random samples become more localized due to the curse of dimensionality [73], which shares some similarities with the low-dimensional problems of low regularity. Therefore, adaptive sampling is needed. In this work, we propose a deep adaptive sampling approach for surrogate modeling of parametric differential equations without labeled data, which generalizes the DAS method [63] to more complicated parametric settings. For simplicity and clarity, we only consider $J_r(u_{\boldsymbol{\theta}})$ and remove the boundary term $J_b(u_{\boldsymbol{\theta}})$. This is because one can employ some penalty-free techniques [3, 56] to remove $J_b(u_{\boldsymbol{\theta}})$ from the loss.

3 Deep Adaptive Sampling for Surrogate Modeling

The statistical error comes from $J_N(u_{\boldsymbol{\theta}})$ as the discretization of $J(u_{\boldsymbol{\theta}})$. One straightforward way to reduce the error of $J_N(u_{\boldsymbol{\theta}})$ is to increase the number of uniformly distributed collocation points in the training set. However, if the solution is of low regularity, the large variance of the residual will significantly reduce the number of effective samples such that the final approximate solution may gain barely any improvement. To alleviate this issue, the selection of collocation points must be consistent with the problem properties, in other words, adaptive sampling needs to be considered.

A deep adaptive sampling (DAS) method has been developed in [63] for deterministic PDEs. We in this work intend to generalize DAS to deal with more complicated parametric differential equations and call this generalization DAS^2 for short, i.e., *deep adaptive sampling* for *surrogates*. The main difficulties come from the additional dimensions from $\boldsymbol{\xi}$. First, samples are needed from both the spatial domain and the parametric space for the discretization of the loss functional. Second, low regularity may come from the spatial domain, the parametric space, or both. Without assuming any prior knowledge of the residual profile, we need to efficiently generate random samples that are consistent with an arbitrary high-dimensional distribution. To handle such a situation, we employ a deep generative model, called KRnet, to approximate the residual-induced distribution and then generate random collocation points accordingly. The PDF defined by KRnet is

$$p_{\mathsf{KRnet}}(\boldsymbol{x},\boldsymbol{\xi};\boldsymbol{\theta}_f) = p_{\boldsymbol{Z}}(f_{\mathsf{KRnet}}(\boldsymbol{x},\boldsymbol{\xi};\boldsymbol{\theta}_f))\left|\det \nabla_{\boldsymbol{x},\boldsymbol{\xi}} f_{\mathsf{KRnet}}\right|, \tag{7}$$

where f_{KRnet} denotes an *invertible* mapping defined by KRnet parameterized with $\boldsymbol{\theta}_f$, and the prior distribution $p_{\boldsymbol{Z}}$ for the random vector $\boldsymbol{z}$ is usually chosen as the standard normal distribution. The overall structure of KRnet is specified as follows

$$\boldsymbol{z} = f_{\mathsf{KRnet}}(\boldsymbol{x},\boldsymbol{\xi}) = L_N \circ f^{\mathsf{outer}}_{[K-1]} \circ \cdots \circ f^{\mathsf{outer}}_{[1]}(\boldsymbol{x},\boldsymbol{\xi}),$$

where $f^{\mathsf{outer}}_{[i]}$ is defined as

$$f^{\mathsf{outer}}_{[k]} = L_S \circ f^{\mathsf{inner}}_{[k,L]} \circ \cdots \circ f^{\mathsf{inner}}_{[k,1]} \circ L_R.$$

Here, $f^{\mathsf{inner}}_{[k,i]}$ is a combination of L affine coupling layers [16, 38] and one scale and bias layer, and L_N, L_S and L_R represent the nonlinear layer, the squeezing layer and the rotation layer respectively, where details can be found in the literature [61–63, 67].

3.1 Sample from a Joint PDF

When the low regularity is related to both $\boldsymbol{x}$ and $\boldsymbol{\xi}$, the adaptive sampling for both $\boldsymbol{x}$ and $\boldsymbol{\xi}$ is needed. We need to generate samples from a joint PDF $\hat{r}(\boldsymbol{x}, \boldsymbol{\xi})$ induced by the residual $r(\boldsymbol{x}, \boldsymbol{\xi}; \boldsymbol{\theta})$ for a certain $\boldsymbol{\theta}$. Following [63], $\hat{r}(\boldsymbol{x}, \boldsymbol{\xi})$ is defined as

$$\hat{r}(\boldsymbol{x}, \boldsymbol{\xi}) \propto r^2(\boldsymbol{x}, \boldsymbol{\xi}; \boldsymbol{\theta}) h(\boldsymbol{x}, \boldsymbol{\xi}),$$

where $h(\boldsymbol{x}, \boldsymbol{\xi})$ is a cutoff function as defined in [63]. The cutoff function $h(\boldsymbol{x}, \boldsymbol{\xi})$ is defined on a compact support $B \supset \Omega$, where $h(\boldsymbol{x}, \boldsymbol{\xi}) = 1$ if $(\boldsymbol{x}, \boldsymbol{\xi}) \in \Omega$ and then decays linearly to 0 towards ∂B. B is chosen to be slightly larger than Ω [63]. We then employ the PDF model induced by KRnet to approximate $\hat{r}(\boldsymbol{x}, \boldsymbol{\xi})$ on B. Mathematically, we need to solve the following optimization problem

$$\boldsymbol{\theta}_f^* = \arg\min_{\boldsymbol{\theta}_f} D_{\mathrm{KL}}(\hat{r}(\boldsymbol{x}, \boldsymbol{\xi}) || p_{\mathrm{KRnet}}(\boldsymbol{x}, \boldsymbol{\xi}; \boldsymbol{\theta}_f)), \tag{8}$$

where $D_{\mathrm{KL}}(\cdot||\cdot)$ denotes the Kullback-Leibler (KL) divergence between two distributions. Let $\boldsymbol{\theta}_f^*$ be the optimal parameter. Since B is slightly larger than Ω, we may generate random samples as

$$(\boldsymbol{x}, \boldsymbol{\xi}) = f_{\mathrm{KRnet}}^{-1}(z; \boldsymbol{\theta}_f^*),$$

and only keep those that belong to Ω. The KL divergence in (8) is

$$\begin{aligned} &D_{\mathrm{KL}}(\hat{r}(\boldsymbol{x}, \boldsymbol{\xi}) || p_{\mathrm{KRnet}}(\boldsymbol{x}, \boldsymbol{\xi}; \boldsymbol{\theta}_f)) \\ &= \int_B \hat{r}(\boldsymbol{x}, \boldsymbol{\xi}) \log \hat{r}(\boldsymbol{x}, \boldsymbol{\xi}) d\boldsymbol{x} d\boldsymbol{\xi} - \int_B \hat{r}(\boldsymbol{x}, \boldsymbol{\xi}) \log p_{\mathrm{KRnet}}(\boldsymbol{x}, \boldsymbol{\xi}) d\boldsymbol{x} d\boldsymbol{\xi}. \end{aligned}$$

The first term is independent on $\boldsymbol{\theta}_f$, which does not affect the optimization step for p_{KRnet} defined in equation (7). So, the PDF approximation step is equivalent to minimizing the cross entropy between $\hat{r}$ and p_{KRnet} [13, 55]:

$$H(\hat{r}, p_{\mathrm{KRnet}}) = -\int_B \hat{r} \log p_{\mathrm{KRnet}} d\boldsymbol{x} d\boldsymbol{\xi}.$$

To compute this cross entropy numerically, we need to use the importance sampling technique since the samples from $\hat{r}$ are not available. Here, we use a PDF model with known parameters $\hat{\boldsymbol{\theta}}_f$ for importance sampling:

$$H(\hat{r}, p_{\mathrm{KRnet}}) \approx -\frac{1}{m} \sum_{i=1}^{m} \frac{\hat{r}(\boldsymbol{x}^{(i)}, \boldsymbol{\xi}^{(i)}) \log p_{\mathrm{KRnet}}(\boldsymbol{x}^{(i)}, \boldsymbol{\xi}^{(i)}; \boldsymbol{\theta}_f)}{p_{\mathrm{KRnet}}(\boldsymbol{x}^{(i)}, \boldsymbol{\xi}^{(i)}; \hat{\boldsymbol{\theta}}_f)}, \tag{9}$$

where m is the number of collocation points for estimating the cross entropy and the choice of $\hat{\boldsymbol{\theta}}_f$ is specified in Algorithm 1.

3.1.1 Factorize a Joint PDF

In the above formulation, we use one KRnet to model the joint distribution of $\boldsymbol{x}$ and $\boldsymbol{\xi}$ without differentiating the dependence of the problem on $\boldsymbol{x}$ and $\boldsymbol{\xi}$. To make the density model more consistent with the problem, we may consider factorizing the joint PDF with conditional distribution. Two types of conditional PDF models can be considered:

$$p_{\boldsymbol{x}|\boldsymbol{\xi}}(\boldsymbol{x}|\boldsymbol{\xi}; \boldsymbol{\theta}_f) = p_{\boldsymbol{Z}|\boldsymbol{\xi}}(f_{\mathrm{KRnet}}(\boldsymbol{x}; \boldsymbol{\xi}, \boldsymbol{\theta}_f)) \left|\det \nabla_{\boldsymbol{x}} f_{\mathrm{KRnet}}\right|,$$

and

$$p_{\boldsymbol{\xi}|\boldsymbol{x}}(\boldsymbol{\xi}|\boldsymbol{x};\boldsymbol{\theta}_f) = p_{\boldsymbol{Z}|\boldsymbol{x}}(f_{\mathsf{KRnet}}(\boldsymbol{\xi};\boldsymbol{x},\boldsymbol{\theta}_f))\left|\det \nabla_{\boldsymbol{\xi}} f_{\mathsf{KRnet}}\right|,$$

both of which are modeled by a conditional KRnet. Note that the generalization of KRnet for conditional distribution is simple, where we simply include the conditional information in the inputs of the affine coupling layers and set $p_{\boldsymbol{Z}|\boldsymbol{x}}$ or $p_{\boldsymbol{Z}|\boldsymbol{\xi}}$ a prior conditional distribution [72]. Then the joint PDF can be written as

$$p_{\boldsymbol{x},\boldsymbol{\xi}}(\boldsymbol{x},\boldsymbol{\xi};\boldsymbol{\theta}_f) = p(\boldsymbol{\xi};\boldsymbol{\theta}_{f_1})p(\boldsymbol{x}|\boldsymbol{\xi};\boldsymbol{\theta}_{f_2}) \tag{10}$$

or

$$p_{\boldsymbol{x},\boldsymbol{\xi}}(\boldsymbol{x},\boldsymbol{\xi};\boldsymbol{\theta}_f) = p(\boldsymbol{x};\boldsymbol{\theta}_{f_1})p(\boldsymbol{\xi}|\boldsymbol{x};\boldsymbol{\theta}_{f_2}), \tag{11}$$

where $p(\boldsymbol{\xi};\boldsymbol{\theta}_{f_1})$ in (10) and $p(\boldsymbol{x};\boldsymbol{\theta}_{f_1})$ in (11) can be modeled by another KRnet. Factorizing the joint PDF, although one more KRnet is needed, improves the modeling capability, especially for random events of small probability. According to the problem properties, we may choose a proper factorization. For example, if it is known that low regularity occurs in the parametric space, we may opt for equation (10). On the other hand, the prior knowledge can also be used to simplify the sampling strategy. For example, if for each $\boldsymbol{\xi}$ the residual profile is smooth in terms of $\boldsymbol{x}$, rather than the joint distribution, we can draw samples from a marginal one as described in the next section.

3.2 Sample from a Marginal PDF

We let

$$\tilde{r}^2(\boldsymbol{\xi};\boldsymbol{\theta}) = \int_{\Omega_s} r^2(\boldsymbol{x},\boldsymbol{\xi};\boldsymbol{\theta})d\boldsymbol{x}.$$

We assume that the low regularity originates only from the parametric space and for any $\boldsymbol{\xi}$ and $\boldsymbol{\theta}$, $\tilde{r}^2(\boldsymbol{\xi};\boldsymbol{\theta})$ can be well approximated by a fixed set of uniform samples $\{\boldsymbol{x}^{(i)}\}_{i=1}^{m_x}$ in the spatial domain, i.e.,

$$\tilde{r}^2(\boldsymbol{\xi};\boldsymbol{\theta}) \approx \frac{1}{m_{\boldsymbol{x}}}\sum_{i=1}^{m_{\boldsymbol{x}}} r^2(\boldsymbol{x}^{(i)},\boldsymbol{\xi};\boldsymbol{\theta}). \tag{12}$$

In this way, the empirical loss in equation (4) could be rewritten as

$$J_N(u_{\boldsymbol{\theta}}) = \frac{1}{N_{\tilde{r}}}\sum_{i=1}^{N_{\tilde{r}}}\tilde{r}^2(\boldsymbol{\xi}^{(i)};\boldsymbol{\theta}) + \gamma\frac{1}{N_b}\sum_{i=1}^{N_b} b^2(\boldsymbol{x}_{\partial\Omega}^{(i)},\boldsymbol{\xi}^{(i)};\boldsymbol{\theta}). \tag{13}$$

Similar to sampling a joint PDF, we can approximate the residual-induced distribution $\hat{r}(\boldsymbol{\xi}) \propto \tilde{r}^2(\boldsymbol{\xi};\boldsymbol{\theta})h(\boldsymbol{\xi})$ by the following optimization problem

$$\boldsymbol{\theta}_f^* = \arg\min_{\boldsymbol{\theta}_f} D_{\mathsf{KL}}(\hat{r}(\boldsymbol{\xi})\|p_{\mathsf{KRnet}}(\boldsymbol{\xi};\boldsymbol{\theta}_f)),$$

where $h(\boldsymbol{\xi})$ is defined the same way as in the previous section on a compact support B_p that is slightly larger than Ω_p. Again, minimizing the KL divergence is equivalent to minimizing the cross entropy between $\hat{r}$ and p_{KRnet}:

$$H(\hat{r},\, p_{\mathsf{KRnet}}) = -\int_{B_p}\hat{r}(\boldsymbol{\xi})\log p_{\mathsf{KRnet}}(\boldsymbol{\xi};\boldsymbol{\theta}_f)d\boldsymbol{\xi}.$$

and we approximate the cross entropy using the importance sampling technique:

$$H(\hat{r},\, p_{\mathsf{KRnet}}) \approx -\frac{1}{m}\sum_{i=1}^{m}\frac{\hat{r}(\boldsymbol{\xi}^{(i)})}{p_{\mathsf{KRnet}}(\boldsymbol{\xi}^{(i)};\hat{\boldsymbol{\theta}}_f)}\log p_{\mathsf{KRnet}}(\boldsymbol{\xi}^{(i)};\boldsymbol{\theta}_f), \tag{14}$$

where $\hat{\boldsymbol{\theta}}_f$ is specified in Algorithm 2.

3.3 Algorithm

Given an initial set of collocation points $\mathsf{S}_{\Omega,0}$, the empirical loss defined in (4) is minimized to yield $u_{\boldsymbol{\theta}_N^{*,(1)}}$. For $\boldsymbol{\theta}_N^{*,(1)}$, one can seek $p_{\mathsf{KRnet}}(\boldsymbol{x},\boldsymbol{\xi};\boldsymbol{\theta}_f^{*,(1)})$ by minimizing the cross entropy (see (9)). In this step, uniform samples are used to compute the cross entropy. After the PDF approximation step is finished, a new set of collocation points $\mathsf{S}^g_{\Omega,1}$ is generated by $p_{\mathsf{KRnet}}(\boldsymbol{x},\boldsymbol{\xi};\boldsymbol{\theta}_f^{*,(1)})$. The training set is refined as $\mathsf{S}_{\Omega,1} = \mathsf{S}_{\Omega,0}\cup\mathsf{S}^g_{\Omega,1}$. We then continue to update $u_{\boldsymbol{\theta}}$ using $\boldsymbol{\theta}_N^{*,(1)}$ as the initial parameters and $\mathsf{S}_{\Omega,1}$ as the training set, resulting in a refined model. In general, at the k-th stage, we minimize the empirical loss on $\mathsf{S}_{\Omega,k-1}$ to get the approximate solution $u_{\boldsymbol{\theta}_N^{*,(k)}}$. For PDF approximation, we let $p_{\mathsf{KRnet}}(\boldsymbol{x},\boldsymbol{\xi};\hat{\boldsymbol{\theta}}_f) = p_{\mathsf{KRnet}}(\boldsymbol{x},\boldsymbol{\xi};\boldsymbol{\theta}_f^{*,(k-1)})$ for importance sampling in equation (9). Once the PDF model is trained, the training set is refined as $\mathsf{S}_{\Omega,k+1} = \mathsf{S}_{\Omega,k}\cup\mathsf{S}^g_{\Omega,k+1}$. We repeat the procedure to obtain an adaptive algorithm for the refinement of the training set by sampling a joint PDF.

Algorithm 1 DAS^2 based on the joint PDF

Input: Initial $p_{\mathsf{KRnet}}(\boldsymbol{x},\boldsymbol{\xi};\boldsymbol{\theta}_f^{(0)})$, $u_{\boldsymbol{\theta}_N^{(0)}}(\boldsymbol{x},\boldsymbol{\xi})$, maximum epoch number N_e, batch size m, initial training set $\mathsf{S}_{\Omega,0} = \{\boldsymbol{x}_0^{(i)},\boldsymbol{\xi}_0^{(i)}\}_{i=1}^{n_r}$.

for $k = 0 : N_{\text{adaptive}} - 1$ **do**
  // Train surrogate models
  for $i = 1 : N_e$ **do**
    for j steps **do**
      Sample m samples from $\mathsf{S}_{\Omega,k}$.
      Update $u_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{\xi})$ by descending the stochastic gradient of $J_N(u_{\boldsymbol{\theta}})$ (see equation (4)).
    end for
  end for
  // Update KRnet
  for $i = 1 : N_e$ **do**
    for j steps **do**
      Sample m samples from $p_{\mathsf{KRnet}}(\boldsymbol{x},\boldsymbol{\xi};\boldsymbol{\theta}_f^{*,(k-1)})$.
      Update $p_{\mathsf{KRnet}}(\boldsymbol{x},\boldsymbol{\xi};\boldsymbol{\theta}_f)$ by descending the stochastic gradient of $H(\hat{r},\, p_{\mathsf{KRnet}})$ (see equation (9)).
    end for
  end for
  // Refine training set
  Generate $\mathsf{S}^g_{\Omega,k+1}\subset\Omega$ with size n_r through $p_{\mathsf{KRnet}}(\boldsymbol{x},\boldsymbol{\xi};\boldsymbol{\theta}_f^{*,(k+1)})$.
  $\mathsf{S}_{\Omega,k+1} = \mathsf{S}_{\Omega,k}\cup\mathsf{S}^g_{\Omega,k+1}$.
end for

Output: $u_{\boldsymbol{\theta}_N^*}(\boldsymbol{x},\boldsymbol{\xi})$

Algorithm 2 DAS^2 based on the marginal PDF

Input: Initial $p_{\mathsf{KRnet}}(\boldsymbol{\xi};\boldsymbol{\theta}_f^{(0)})$, $u_{\boldsymbol{\theta}_N^{(0)}}(\boldsymbol{x},\boldsymbol{\xi})$, maximum epoch number N_e, batch size m, initial training set $\mathsf{S}_{\Omega_p,0}=\{\boldsymbol{\xi}_0^{(i)}\}_{i=1}^{n_r}$, $m_{\boldsymbol{x}}$ samples from Ω_s.

for $k=0:N_{\text{adaptive}}-1$ **do**
  // Train surrogate models
  for $i=1:N_e$ **do**
    for j steps **do**
      Sample m samples from $\mathsf{S}_{\Omega_p,k}$.
      Update $u_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{\xi})$ by descending the stochastic gradient of $J_N(u_{\boldsymbol{\theta}})$ (see equation (13)).
    end for
  end for
  // Update KRnet
  for $i=1:N_e$ **do**
    for j steps **do**
      Sample m samples from $p_{\mathsf{KRnet}}(\boldsymbol{\xi};\boldsymbol{\theta}_f^{*,(k-1)})$.
      Update $p_{\mathsf{KRnet}}(\boldsymbol{\xi};\boldsymbol{\theta}_f)$ by descending the stochastic gradient of $H(\hat{r},\, p_{\mathsf{KRnet}})$ (see equation (14)).
    end for
  end for
  // Refine training set
  Generate $\mathsf{S}^g_{\Omega_p,k+1}\subset\Omega_p$ with size n_r through $p_{\mathsf{KRnet}}(\boldsymbol{\xi};\boldsymbol{\theta}_f^{*,(k+1)})$.
  $\mathsf{S}_{\Omega_p,k+1}=\mathsf{S}_{\Omega_p,k}\cup\mathsf{S}^g_{\Omega_p,k+1}$.
end for

Output: $u_{\boldsymbol{\theta}_N^*}(\boldsymbol{x},\boldsymbol{\xi})$

For simplicity and clarity, we focus on the adaptivity of S_Ω and the treatment of the boundary points can be found in [63] (section 4.3). The deep adaptive sampling algorithm for surrogate modeling is summarized in Algorithm 1, where N_{adaptive} is a given number of maximum adaptivity iterations, m is the batch size for stochastic gradient, and N_e is the number of epochs for training $u_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{\xi})$ and $p_{\mathsf{KRnet}}(\boldsymbol{x},\boldsymbol{\xi};\boldsymbol{\theta}_f)$. The algorithms consist of three steps in one loop: training surrogate models, updating KRnet and refining the training set. The same procedure can be applied to the marginal PDF, which results in Algorithm 2.

4 Analysis

Inspired by the literature [14, 15], we include some preliminary analysis of DAS^2. We first establish the relationship between the loss functional and its discretization at the optimal model parameters for a certain training set. For the ideal case, we show that the expectation of the discretized loss functional does not increase at the optimal model parameters given by two adjacent adaptivity iterations. Before presenting the analysis, the following assumptions are introduced.

Assumption 1 ([14]) Let $\boldsymbol{\theta}\in\Theta=[-a,a]^D$ be the trainable parameters of $u_{\boldsymbol{\theta}}$ where $a>0$ is a constant. Assume that two operators $\mathscr{M}_1:\boldsymbol{\theta}\mapsto J_{r,N}$ and $\mathscr{M}_2:\boldsymbol{\theta}\mapsto J_r$ are Lipschitz continuous in the ℓ_∞ sense with Lipschitz constant $\mathfrak{L}$ for $\boldsymbol{\theta}\in\Theta$.

Assumption 2 ([14]) Let $c>0$ be a constant that is independent of Θ. Assume that $J_{r,N}\in[0,c]$ for all $\boldsymbol{\theta}\in\Theta$.

Assumption 3 ([63]) Assume that $p_{\text{KRnet}}(\boldsymbol{x}, \boldsymbol{\xi}; \boldsymbol{\theta}_f^{*,(k)})$ is the optimal candidate for the change of measure for problem (5) at the k-th stage

$$p_{\text{KRnet}}(\boldsymbol{x}, \boldsymbol{\xi}; \boldsymbol{\theta}_f^{*,(k)}) = c_k r^2(\boldsymbol{x}, \boldsymbol{\xi}; \boldsymbol{\theta}_N^{*,(k)}),$$

where $\boldsymbol{\theta}_N^{*,(k)}$ is the minimizer in (5) and $\boldsymbol{\theta}_f^{*,(k)}$ is the minimizer in (8) given $\boldsymbol{\theta}_N^{*,(k)}$, and

$$c_k = 1/\int_{\Omega} r^2(\boldsymbol{x}, \boldsymbol{\xi}; \boldsymbol{\theta}_N^{*,(k)}) d\boldsymbol{x} d\boldsymbol{\xi}$$

is the normalization constant.

If the collocation points are independently and identically distributed according to a given probability distribution, then $J_r(u_{\boldsymbol{\theta}_N^*})$ can be bounded by the discretized loss with high probability, which is stated as follows.

Theorem 1 *Suppose that Assumption 1 and Assumption 2 are satisfied and the boundary loss is zero. Let $\boldsymbol{\theta}_N^*$ be a minimizer of $J_{r,N}$ where the collocation points are independently drawn from a given probability distribution. Given $\varepsilon \in (0, 1)$, the following inequality holds*

$$J_r(u_{\boldsymbol{\theta}_N^*}) \leq \varepsilon^2 + J_{r,N}(u_{\boldsymbol{\theta}_N^*})$$

with probability at least $1 - (4a\mathfrak{L}/\varepsilon^2)^D \exp(-N_r\varepsilon^4/2c^2)$.

Proof See Appendix A. □

The expectation of the discretized loss at two adjacent adaptivity stages satisfies the following property.

Theorem 2 *Under the same conditions of Theorem 1, suppose that Assumption 3 is satisfied. Assume that*

$$J_{r,N}(u_{\boldsymbol{\theta}_N^{*,(k)}}) = \frac{1}{N_r} \sum_{i=}^{N_r} \frac{r^2(\boldsymbol{x}^{(i)}, \boldsymbol{\xi}^{(i)}; \boldsymbol{\theta}_N^{*,(k)})}{p_{\text{KRnet}}(\boldsymbol{x}^{(i)}, \boldsymbol{\xi}^{(i)}; \boldsymbol{\theta}_f^{*,(k-1)})},$$

where each $(\boldsymbol{x}^{(i)}, \boldsymbol{\xi}^{(i)})$ is drawn from $p_{\text{KRnet}}(\boldsymbol{x}^{(i)}, \boldsymbol{\xi}^{(i)}; \boldsymbol{\theta}_f^{*,(k-1)})$, then the following inequality holds

$$\mathbb{E}(J_{r,N}(u_{\boldsymbol{\theta}_N^{*,(k+1)}})) \leq \mathbb{E}(J_{r,N}(u_{\boldsymbol{\theta}_N^{*,(k)}})).$$

Proof See Appendix B. □

Theorem 1 provides the relationship between the loss and the discretized loss, which is similar to the results in [14, 15]. The analysis in Theorem 1 is not restricted to the linear differential equations, while the results in [63] only involve non-parametric linear differential equations. From the above analysis, if the number of parameters in $u_{\boldsymbol{\theta}}$ and the number of samples are properly chosen, then the loss is bounded by the discretized loss with high probability. In Theorem 2, for simplicity, we consider replacing all collocation points with new samples from the trained deep generative model [63]. Although the analysis of the loss behavior during the adaptive procedure is restricted to such a strategy, it can provide a perspective to understand the mechanism of DAS^2. We note that quantifying the decay of the error is not straightforward since it depends on the optimization procedure of deep neural networks. However, obtaining the convergence rate of such an optimization problem is still an open question.

5 Numerical Study

In this section, we conduct four numerical experiments, including the physics-informed operator learning problem, the parametric optimal control problem with geometrical parametrization, and the parametric lid-driven 2D cavity flow problem with a continuous range of Reynolds numbers from 100 to 3200, to demonstrate the effectiveness of the proposed method. All these test problems involve low regularity. Two types of neural network structures are considered for the surrogate model $u_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{\xi})$: one is the feedforward neural network with inputs $\boldsymbol{x}$ and $\boldsymbol{\xi}$, and the other one is the structure given by DeepONet [45] where the problem is treated as an physics-informed operator learning problem. The choice of the sampling strategy depends on the problem property instead of the model structure. For comparison, we also test the performance of some baseline sampling strategies, such as the residual-based adaptive refinement (RAR) method [46, 74], the quasi-random sampling (QRS) method implemented in the SciPy module [66], and an adaptive collocation point movement approach based on interacting particle methods in a recent literature [30]. The code of this study will be released on https://github.com/MJfadeaway/DAS-2.

5.1 A One-Dimensional Parametric Ordinary Differential Equation

We start with the following one-dimensional parametric ordinary differential equation (ODE)

$$\frac{\mathrm{d}u}{\mathrm{d}x} = \xi u, \quad u(0, \xi) = u_0, \quad x \in [0, 1],$$

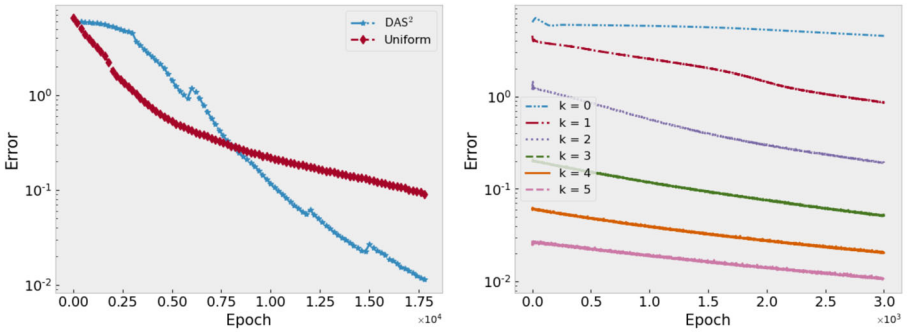
where $\xi \in \Omega_p = [-3, 3]$, i.e., $C_1 = -3$ and $C_2 = 3$ (see section 2), and the initial condition is set to $u_0 = 1$. The exact solution is

$$u(x, \xi) = u_0 e^{\xi x}. \tag{15}$$

This is a widely used test problem for polynomial chaos methods in uncertainty quantification [78].

We use a six-layer fully connected neural network to construct a surrogate model $u_{\boldsymbol{\theta}}(x, \xi)$ as the approximation solution of the parametric ODE, where each hidden layer has 32 neurons. For KRnet, we set $K = 2$ and take $L = 6$ affine coupling layers. For each affine coupling layer, a two-layer fully connected neural network is used, where each hidden layer has 24 neurons. The maximum epoch number for training both $u_{\boldsymbol{\theta}}(x, \xi)$ and $p_{\mathsf{KRnet}}(x, \xi; \boldsymbol{\theta}_f)$ is set to $N_e = 3000$. In this test problem, the ADAM optimizer [37] is employed for all training processes. The learning rate for the ADAM optimizer is set to 0.0001, and the batch size is set to $m = 1000$. For DAS^2, we use the joint PDF for sampling. The collocation points in the initial training set are uniform samples, $n_r = 1000$ is set during the adaptive sampling procedure, and the number of adaptivity iterations is set to $N_{\text{adaptive}} = 6$. For the uniform sampling strategy, the maximum epoch number is set to be the same as the total number of epochs of DAS^2, and the number of samples is set to $|\mathsf{S}_\Omega| = 6000$ (the same as the 6-th adaptivity iteration of DAS^2). To assess the effectiveness of our DAS^2 method, we generate a uniform meshgrid with size 256×256 in the spatial-parametric space $[0, 1] \times [-3, 3]$ and compute the mean square error on these grid points.

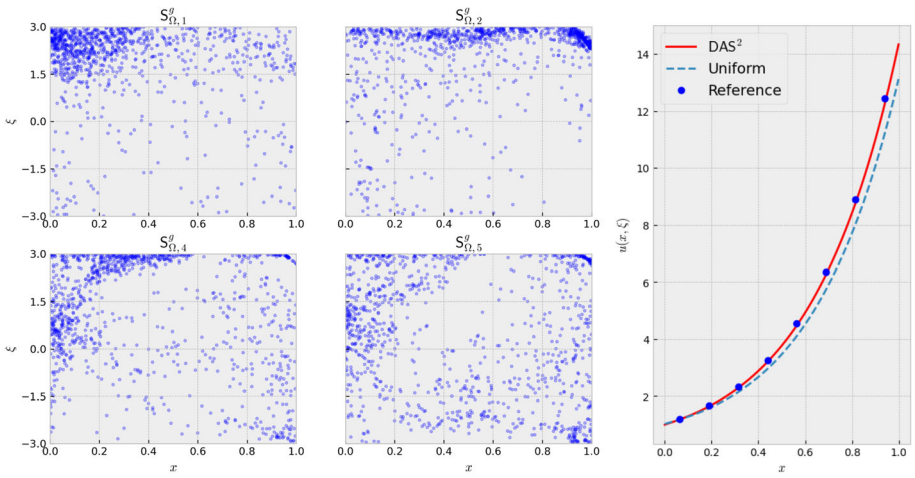
In Fig. 1, we plot the approximation error given by different sampling strategies with respect to epoch in the left plot and the error evolution of DAS^2 at different adaptivity iteration steps in the right plot. In terms of the number of epochs, the error of DAS^2 decays more quickly than the uniform sampling method. The approximation error of DAS^2 drops as



(a) The error with respect to epoch.

(b) The error of DAS^2 at certain adaptivity iteration steps.

Fig. 1 The errors for the parametric ODE test problem



(a) The evolution of $\mathsf{S}^g_{\Omega,k}$ in DAS^2.

(b) $u(x,\xi)$.

Fig. 2 The results of the parametric ODE test problem. Left: The evolution of $\mathsf{S}^g_{\Omega,k}$ in DAS^2; Right: The exact solution and the approximate solutions with different sampling strategies for $\xi = 2.725$

the adaptivity iteration step k increases. It can be found from equation (15) that $u(x,\xi)$ grows exponentially with respect to x and ξ. When ξ is near 3, the solution increases dramatically. To capture this information, more samples are located in the area that ξ is near 3. Figure 2(a) shows the evolution of S^g_Ω of DAS^2 with respect to adaptivity iterations $k = 1, 2, 4, 5$ ($|\mathsf{S}^g_{\Omega,k}| = 1000$), where $\mathsf{S}^g_{\Omega,1}$ indicates that large point-wise residuals are located in the upper left corner of the x-ξ plane. After the set of collocation points is augmented by $\mathsf{S}^g_{\Omega,1}$, the residual profile changes as shown in $\mathsf{S}^g_{\Omega,2}$. Such a pattern is repeated until $\mathsf{S}^g_{\Omega,k}$ is near a set of uniform samples. Figure 2(b) shows the exact solution, the solution obtained by DAS^2 and the solution obtained by the uniform sampligng strategy for $\xi = 2.725$. It is seen that DAS^2 yields a more accurate approximation.

5.2 Physics-Informed Operator Learning for a Dynamical System with High-Dimensional Parameters

Next we consider the following dynamical system

$$\begin{cases} \dfrac{\mathrm{d}u(x,\boldsymbol{\xi})}{\mathrm{d}x} = e^{-D\|\boldsymbol{\xi}-\mathbf{0.5}\|^2} f(x,\boldsymbol{\xi}),\ x \in [0,1], \\ u(0,\boldsymbol{\xi}) = 0, \end{cases} \tag{16}$$

where D is a fixed parameter, and $\boldsymbol{\xi} \in \Omega_p = [-M, M]^d$, i.e., $C_1 = -M$ and $C_2 = M$. The goal is to learn the solution operator from f to the solution u without any paired input–output data when f is sampled from a given function space. This example without $\exp(-D\|\boldsymbol{\xi} - \mathbf{0.5}\|^2)$ is used to test the performance of DeepONet [45, 68]. Here, we add a term $\exp(-D\|\boldsymbol{\xi} - \mathbf{0.5}\|^2)$ to the right-hand side to make this problem more challenging. We assume that f is drawn from the space spanned by orthogonal (e.g. Chebyshev) polynomials as studied in [45]. Let T_i be Chebyshev polynomials of the first kind. We define the orthogonal polynomials of degree d as:

$$V_{\text{poly}} = \left\{ \sum_{i=0}^{d-1} \xi_i T_i(x) : |\xi_i| \le M \right\}.$$

This function space is parameterized with $\boldsymbol{\xi} = [\xi_0, \xi_1, ..., \xi_{d-1}]^{\mathsf{T}}$. Given a realization of $\boldsymbol{\xi}$, we can generate a continuous function f as the following form

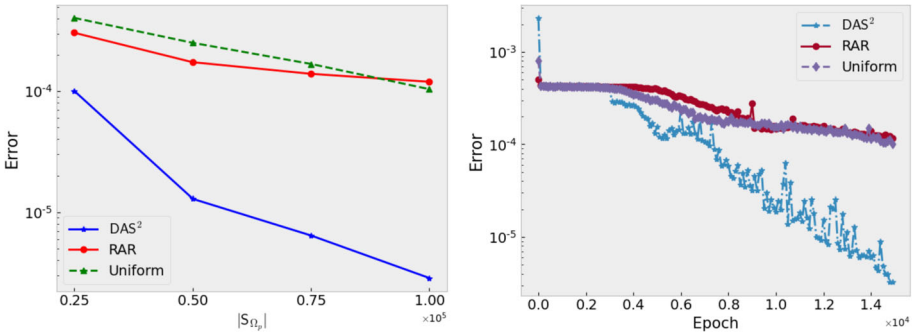
$$f(x,\boldsymbol{\xi}) = \sum_{i=0}^{d-1} \xi_i T_i(x).$$

In this example, the parametric solution $u(x,\boldsymbol{\xi})$ is approximated by

$$u_{\boldsymbol{\theta}}(x,\boldsymbol{\xi}) \approx \sum_{i=1}^{l} q_{\boldsymbol{\theta}_1}^{(i)}(x) t_{\boldsymbol{\theta}_2}^{(i)}(\boldsymbol{\xi}) + b_0, \tag{17}$$

where $q_{\boldsymbol{\theta}_1}^{(i)}$ and $t_{\boldsymbol{\theta}_2}^{(i)}$ are i-th outputs of two neural networks q (parameterized with $\boldsymbol{\theta}_1$) and t (parameterized with $\boldsymbol{\theta}_2$) respectively, both of which have l outputs, and $b_0 \in \mathbb{R}$ is a bias to be trained. Denoting the whole parameters in (17) by $\boldsymbol{\theta} = \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, b_0\}$ for short.

The experimental setup is as follows. We set $M = 1, d = 8, D = 10$. $q_{\boldsymbol{\theta}_1}(x)$ and $t_{\boldsymbol{\theta}_2}(\boldsymbol{\xi})$ are both five-layer fully connected neural networks and each hidden layer has 50 neurons. For DAS^2, we use a marginal PDF for adaptive sampling because the singularity is mainly in the parametric space. To compute the marginal PDF and the loss functional, we use $m_{\boldsymbol{x}} = 100$ uniform grid points in $[0, 1]$ to discretize the integral in equation (12) and (13), in other words, adaptive sampling is not considered in the physical space since the low regularity is from the parametric space. We set $K = 4$ and the configuration for the affine coupling layer is the same as the previous example. The number of epochs for training both $u_{\boldsymbol{\theta}}(x,\boldsymbol{\xi})$ and $p_{\mathsf{KRnet}}(\boldsymbol{\xi};\boldsymbol{\theta}_f)$ is set to $N_e = 3000$. The learning rate for the ADAM optimizer is set to 0.0001, and the batch size is set to $m = 5000$. The numbers of adaptivity iterations is set to $N_{\text{adaptive}} = 5$. For the uniform sampling strategy, we generate $f(x,\boldsymbol{\xi})$ with each $\xi_i \sim \mathsf{Uni}(-M, M)$ where $\mathsf{Uni}(-M, M)$ is the uniform distribution on $[-M, M]$. To measure the quality of approximation, we generate a validation set, which contains 10000 uniformly distributed points in $[-M, M]^d$ and 10000 points in the d-dimensional ball centered at $\mathbf{0.5}$



(a) The error w.r.t. sample size $|S_{\Omega_p}|$. (b) The error evolution with $|S_{\Omega_p}| = 1 \times 10^5$.

Fig. 3 Approximation errors for the physics-informed operator learning problem

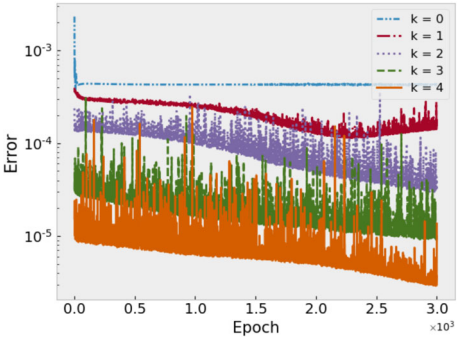


Fig. 4 The error evolution of DAS^2 at different adaptivity iteration steps for the physics-informed operator learning problem. $|S_{\Omega_p}| = 1 \times 10^5$

Table 1 The physics-informed operator learning problem: inference time and error for different $|S_{\Omega_p}|$ and sampling strategies

Sampling strategy	$\lvert S_{\Omega_p}\rvert$			
	2.5×10^4	5×10^4	7.5×10^4	1×10^5
Uniform (0.006s)	4.0×10^{-4}	2.5×10^{-4}	1.7×10^{-4}	1.0×10^{-4}
RAR (0.006s)	3.0×10^{-4}	1.7×10^{-4}	1.4×10^{-4}	1.2×10^{-4}
DAS^2 (0.03s)	1.0×10^{-4}	1.3×10^{-5}	6.4×10^{-6}	2.9×10^{-6}

The computing time of RK45 is about 105 s

with radius 0.5. To compute the reference solution, we employ the classical Runge-Kutta45 (RK45) method to solve the ODE for each function f with a certain $\boldsymbol{\xi}$.

In Fig. 3, we plot the mean square error of different sampling strategies with respect to the sample size $|S_{\Omega_p}|$ in the left plot and with respect to the number of epochs in the right plot. For DAS^2, the numbers of collocation points in $S^g_{\Omega_p,k}$ $(k = 1, 2, 3, 4)$ are set to $n_r = 5 \times 10^3, 1 \times 10^4, 1.5 \times 10^4, 2 \times 10^4$ for $|S_{\Omega_p}| = 2.5 \times 10^4, 5 \times 10^4, 7.5 \times 10^4, 1 \times 10^5$ respectively. For the uniform sampling strategy, the model is trained with 1.5×10^4 epochs to match the total number of epochs of DAS^2. For the heuristic method RAR, the numbers of collocation points in $S^g_{\Omega_p,k}$ $(k = 1, 2, 3, 4)$ are set to $n_r = 2.5 \times 10^3, 5 \times 10^3, 7.5 \times 10^3, 1 \times 10^4$

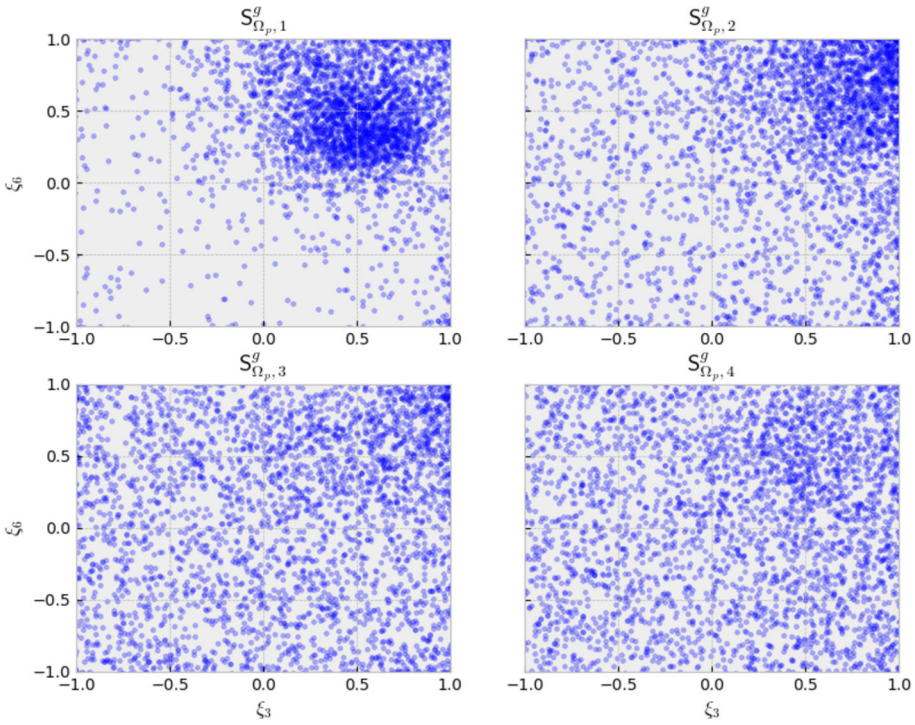


Fig. 5 The evolution of $S^g_{\Omega_p,k}$ in DAS^2 for the physics-informed operator learning problem, $|S_{\Omega_p}| = 1 \times 10^5$

for $|S_{\Omega_p}| = 2.5 \times 10^4, 5 \times 10^4, 7.5 \times 10^4, 1 \times 10^5$ respectively. From the left plot of Fig. 3, it can be seen that DAS^2 improves the accuracy significantly compared to the uniform sampling strategy and RAR. The right plot of Fig. 3 shows that as the number of epochs increases, especially from the start of the third adaptivity iteration, the error of DAS^2 decreases much faster than those of uniform sampling and RAR. Figure 4 shows the errors of DAS^2 at each adaptivity iteration step k. It is seen that the error drops dramatically after we refine the solution using $S^g_{\Omega_p,1}$ and $S^g_{\Omega_p,2}$. Table 1 shows the inference time and the errors for the uniform sampling strategy, RAR and DAS^2. As a surrogate model, the inference time of DAS^2 is much less than that of RK45, which is desired. It can be seen that the inference time of DAS^2 is more than that of RAR. However, the errors of the uniform sampling strategy and RAR are much larger than that of DAS^2 since the uniform sampling strategy and RAR are not able to accurately discretize the loss functional for this low-regularity high-dimensional problem [74]. From Table 1, it is clear that DAS^2 is one order of magnitude more accurate than RAR and the uniform sampling strategy.

Figure 5 shows 3000 samples from DAS^2 for the four adaptivity iterations, where the components ξ_3 and ξ_6 are used for visualization. We have also checked the other components, and no significantly different results were found. $S^g_{\Omega_p,1}$ shows that the error profile has a peak around $\hat{\boldsymbol{\xi}} = \mathbf{0.5}$ which matches the fact that there is a decay term with respect to $\boldsymbol{\xi}$ in equation (16). After the training set is augmented with $S^g_{\Omega_p,1}$, the error profile becomes more flat as shown by the distribution of $S^g_{\Omega_p,2}$. This is expected since more training samples are

added to the neighborhood of $\mathbf{0.5}$ where the error should be reduced. Figure 6 shows $u(x, \boldsymbol{\xi})$ corresponding to different $\boldsymbol{\xi}$ obtained by DAS^2, RAR and the uniform sampling method. The realizations of $\boldsymbol{\xi}$ we choose for visualization are randomly drawn from the d-dimensional ball centered at $\mathbf{0.5}$ with radius 0.5, since $u(x, \boldsymbol{\xi})$ is close to zero when $\boldsymbol{\xi}$ is far away from $\mathbf{0.5}$ due to the decay term in problem (16). As shown in Fig. 6, for different $\boldsymbol{\xi}$ the solutions $u(x, \boldsymbol{\xi})$ obtained by DAS^2 are much more accurate than those given by RAR and uniform sampling.

5.3 Surrogate Modeling for an Optimal Control Problem with Geometrical Parametrization

In this test case, we are going to build a surrogate model for the following parametric optimal control problem:

$$\begin{cases} \min\limits_{y(\boldsymbol{x},\boldsymbol{\xi}),u(\boldsymbol{x},\boldsymbol{\xi})} J\left(y\left(\boldsymbol{x},\boldsymbol{\xi}\right),u\left(\boldsymbol{x},\boldsymbol{\xi}\right)\right)=\dfrac{1}{2}\left\|y(\boldsymbol{x},\boldsymbol{\xi})-y_d(\boldsymbol{x},\boldsymbol{\xi})\right\|_{2,\Omega}^2+\dfrac{\alpha}{2}\left\|u(\boldsymbol{x},\boldsymbol{\xi})\right\|_{2,\Omega}^2, \\ \text{subject to} \begin{cases} -\Delta y(\boldsymbol{x},\boldsymbol{\xi})=u(\boldsymbol{x},\boldsymbol{\xi}) & \text{in } \Omega, \\ y(\boldsymbol{x},\boldsymbol{\xi})=1 & \text{on } \partial\Omega, \end{cases} \\ \text{and} \quad u_a \le u(\boldsymbol{x},\boldsymbol{\xi}) \le u_b \quad \text{a.e. in } \Omega, \end{cases} \tag{18}$$

where $\boldsymbol{\xi}=(\xi_1,\xi_2)$ represents the geometrical and desired state parameters. The parametric computational domain (also depending on $\boldsymbol{\xi}$) is $\Omega=([0,2]\times[0,1])\backslash\mathbb{B}((1.5,0.5),\xi_1)$ which is illustrated in Fig. 7 and the desired state is given by

$$y_d(\boldsymbol{x},\boldsymbol{\xi})=\begin{cases} 1 & \text{in } \Omega_1=[0,1]\times[0,1], \\ \xi_2 & \text{in } \Omega_2=([1,2]\times[0,1])\backslash\mathbb{B}((1.5,0.5),\xi_1), \end{cases}$$

where $\mathbb{B}((1.5,0.5),\xi_1)$ is a ball of radius ξ_1 with center $(1.5,0.5)$. We set $\alpha=0.001$, $u_a=0$, $u_b=10$, and the domain for the parameter to be $\boldsymbol{\xi}\in\Omega_p=[0.05,0.45]\times[0.5,2.5]$.This test problem is related to the application of local hyperthermia treatment of cancer, which is inspired by the literature [35, 49]. The background of this test problem is that we expect to accomplish a specific temperature field in the tumor area and another temperature field in the non-lesion area by heat source control. The circle represents a certain body organ where the tumor area is. We intend to seek an effective surrogate model of the optimal heat source control for different expected temperature fields and organ shapes (i.e. different $\boldsymbol{\xi}$).

As studied in [79], one can use the necessary conditions for the minimizer of (18) to find the optimal solution to the parametric optimal control problem. That is, we solve the KKT system of (18) to find its minimizer, which is a parametric PDE system as follows

$$\begin{cases} -\Delta y(\boldsymbol{x},\boldsymbol{\xi})=u(\boldsymbol{x},\boldsymbol{\xi}) & \text{in } \Omega, \\ y(\boldsymbol{x},\boldsymbol{\xi})=1 & \text{on } \partial\Omega, \\ -\Delta p(\boldsymbol{x},\boldsymbol{\xi})=y(\boldsymbol{x},\boldsymbol{\xi})-y_d(\boldsymbol{x},\boldsymbol{\xi}) & \text{in } \Omega, \\ p(\boldsymbol{x},\boldsymbol{\xi})=0 & \text{on } \partial\Omega, \\ u(\boldsymbol{x},\boldsymbol{\xi})=-\dfrac{1}{\alpha}\mathrm{P}_{[u_a,u_b]}\left(p\left(\boldsymbol{x},\boldsymbol{\xi}\right)\right) & \text{in } \Omega, \end{cases} \tag{19}$$

where $p(\boldsymbol{x},\boldsymbol{\xi})$ is the adjoint variable and

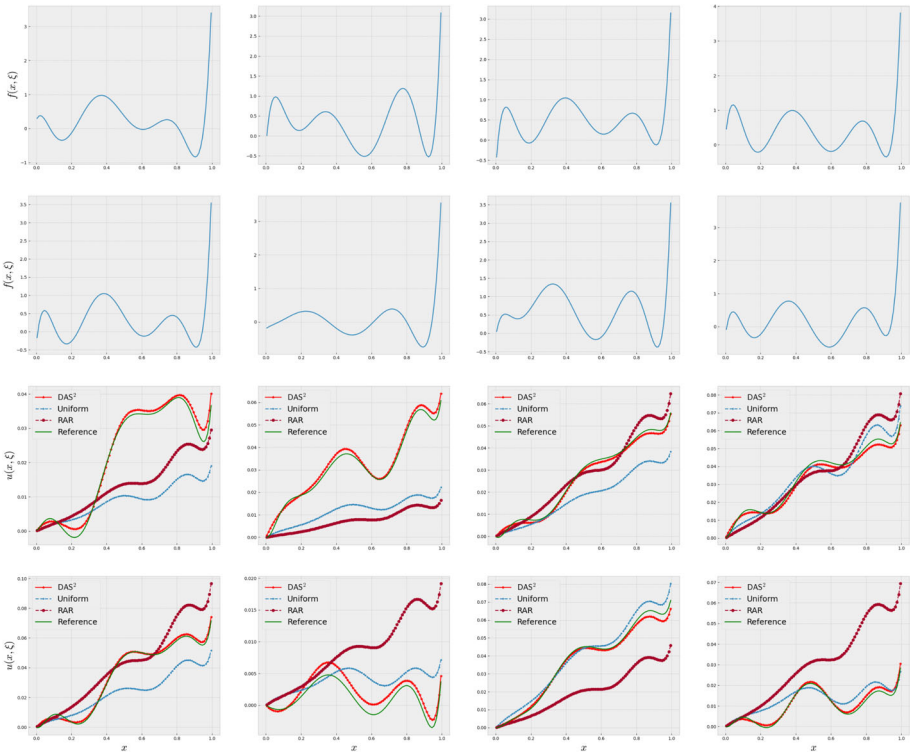


Fig. 6 Solutions of the physics-informed operator learning problem: the first two rows above show $f(x, \boldsymbol{\xi})$ with different realizations of $\boldsymbol{\xi}$, the two rows below show the corresponding solutions $u(x, \boldsymbol{\xi})$

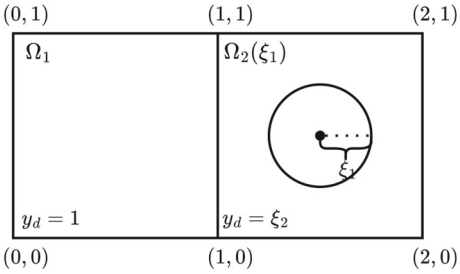
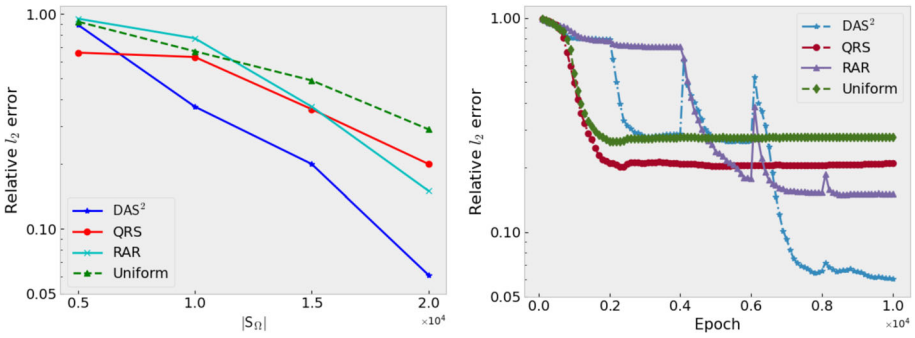


Fig. 7 The parametric computational domain Ω

$$\mathrm{P}_{[u_a,u_b]}\left(p\left(\boldsymbol{x},\boldsymbol{\xi}\right)\right)=\begin{cases} u_b, & \text{if } u_b < p(\boldsymbol{x},\boldsymbol{\xi}),\\ p(\boldsymbol{x},\boldsymbol{\xi}), & \text{if } u_a \le p(\boldsymbol{x},\boldsymbol{\xi}) \le u_b,\\ u_a, & \text{if } p(\boldsymbol{x},\boldsymbol{\xi}) < u_a. \end{cases}$$

Define a length factor function as [79]

$$l(\boldsymbol{x},\boldsymbol{\xi}) = x_1(2-x_1)x_2(1-x_2)(\xi_1^2-(x_1-1.5)^2-(x_2-0.5)^2).$$



(a) The error w.r.t. sample size $|S_\Omega|$. (b) The error evolution with $|S_\Omega| = 2 \times 10^4$.

Fig. 8 Approximation errors for the parametric optimal control problem

We choose three six-layer fully connected neural networks $u_{\boldsymbol{\theta}_u}(\boldsymbol{x}, \boldsymbol{\xi})$, $y_{\boldsymbol{\theta}_y}(\boldsymbol{x}, \boldsymbol{\xi})$ and $p_{\boldsymbol{\theta}_p}(\boldsymbol{x}, \boldsymbol{\xi})$, where each hidden layer has 25 neurons. We let

$$u(\boldsymbol{x}, \boldsymbol{\xi}) \approx u_{\boldsymbol{\theta}_u}(\boldsymbol{x}, \boldsymbol{\xi}),\ y(\boldsymbol{x}, \boldsymbol{\xi}) \approx l(\boldsymbol{x}, \boldsymbol{\xi}) y_{\boldsymbol{\theta}_y}(\boldsymbol{x}, \boldsymbol{\xi}) + 1,\ p(\boldsymbol{x}, \boldsymbol{\xi}) \approx l(\boldsymbol{x}, \boldsymbol{\xi}) p_{\boldsymbol{\theta}_p}(\boldsymbol{x}, \boldsymbol{\xi})$$

The Dirichlet boundary conditions of $y(\boldsymbol{x}, \boldsymbol{\xi})$ and $p(\boldsymbol{x}, \boldsymbol{\xi})$ are naturally satisfied. We then substitute the defined approximators into equation (19) to minimize the residual. More details about the discretization of problem (18) can be found in [79]. We here focus on the importance of adaptive sampling for surrogate modeling.

In this example, we use the joint PDF for sampling and the spatial-parametric space is defined as:

$$\begin{aligned}\Omega := \{(\boldsymbol{x}, \boldsymbol{\xi}) | & 0 \le x_1 \le 2, 0 \le x_2 \le 1, 0.05 \le \xi_1 \le 0.45, 0.5 \le \xi_2 \le 2.5, \\ & (x_1 - 1.5)^2 + (x_2 - 0.5)^2 \ge \xi_1^2\}.\end{aligned}$$

To obtain an accurate approximation, the optimizer for training $u_{\boldsymbol{\theta}_u}$, $y_{\boldsymbol{\theta}_y}$ and $p_{\boldsymbol{\theta}_p}$ is set to be the BFGS method [33], followed by the setting in [79]. The number of epochs for solving PDEs is set to $N_e = 2000$. For KRnet, we set $K = 2$ and the configuration for the affine coupling layers remains the same as the previous experiment. KRnet is trained by the ADAM optimizer with a learning rate 0.0001, where the number of epochs is set to $N_e = 2000$. For DAS^2, the number of adaptivity iterations is set to $N_{\text{adaptive}} = 5$. To demonstrate the effectiveness of the proposed method, we adopt dolfin-adjoint [48] to solve the optimal control problem with some fixed parameters. The dolfin-adjoint solutions, which are regarded as the ground truth, are evaluated on a 200×100 grid for the physical domain and for $\boldsymbol{\xi}$ located on an 11×11 grid for the parametric domain.

In Fig. 8, we plot the relative l_2 errors given by different sampling strategies with respect to the sample size in the left plot and with respect to the number of epochs in the right plot. For each sample size, we take three runs with different initialization and compute the mean relative error of the three runs as the final error. For DAS^2, the size of the initial training set $|S_{\Omega,0}| = n_r$ is set to $1 \times 10^3, 2 \times 10^3, 3 \times 10^3, 4 \times 10^3$ for $|S_\Omega| = 0.5 \times 10^4, 1 \times 10^4, 1.5 \times 10^4, 2 \times 10^4$ respectively. For the uniform sampling strategy and the qusi-random sampling (QRS) strategy, the number of epochs is set to be the same as the total number of epochs of DAS^2, and the number of points in S_Ω is also set to be the same as DAS^2. For the heuristic method RAR, the numbers of collocation points in $S_{\Omega,k}^g$ $(k = 1, 2, 3, 4)$ are set to $n_r = 5 \times 10^2, 1 \times 10^3, 1.5 \times 10^3, 2 \times 10^3$ for $|S_\Omega| = 5 \times 10^3, 1 \times 10^4, 1.5 \times 10^4, 2 \times 10^4$

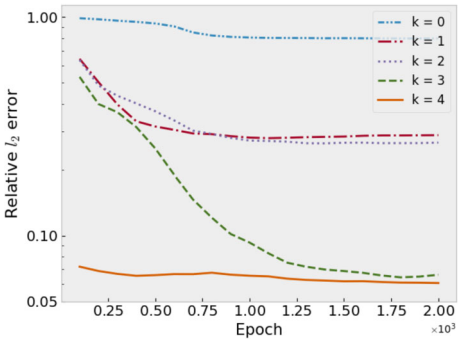
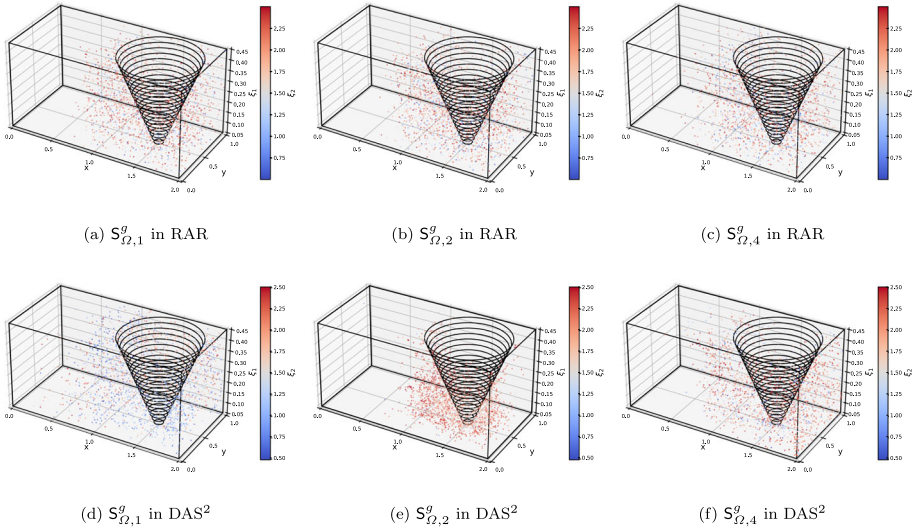


Fig. 9 The errors for the parametric optimal control problem of DAS^2 at different adaptivity iteration steps. $|\mathsf{S}_\Omega| = 2 \times 10^4$



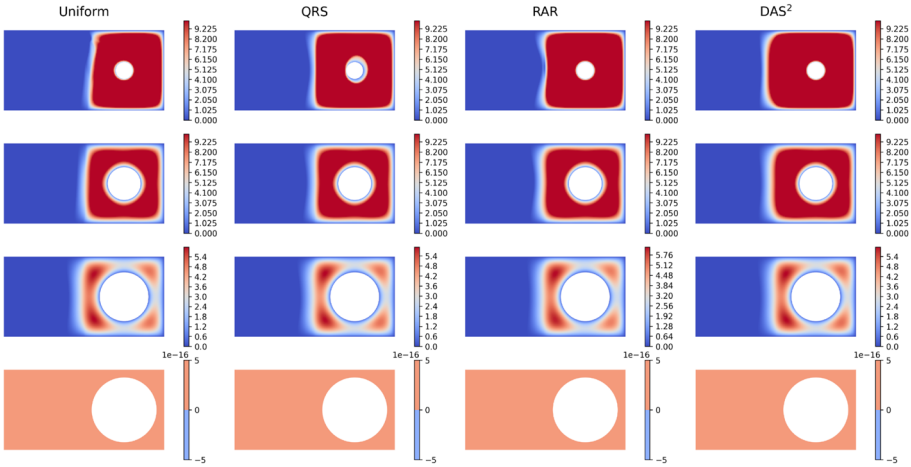
(a) $\mathsf{S}^g_{\Omega,1}$ in RAR (b) $\mathsf{S}^g_{\Omega,2}$ in RAR (c) $\mathsf{S}^g_{\Omega,4}$ in RAR

(d) $\mathsf{S}^g_{\Omega,1}$ in DAS^2 (e) $\mathsf{S}^g_{\Omega,2}$ in DAS^2 (f) $\mathsf{S}^g_{\Omega,4}$ in DAS^2

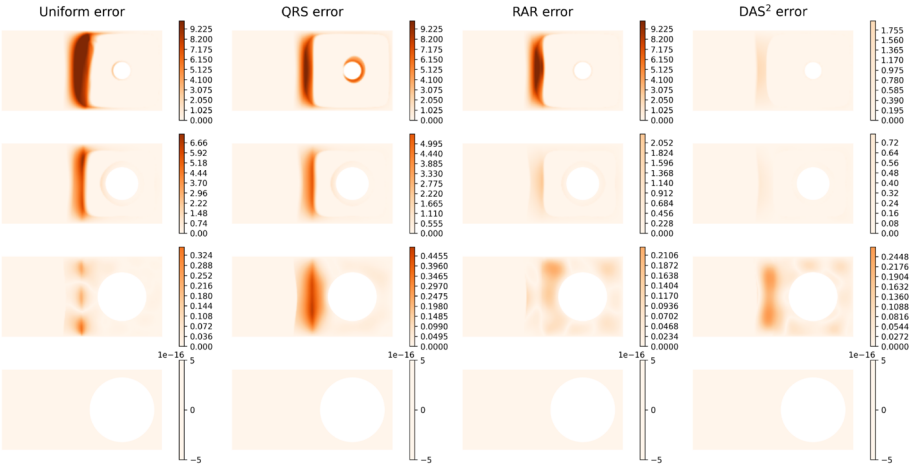
Fig. 10 The evolution of $\mathsf{S}^g_{\Omega,k}$ in RAR and DAS^2 for the parametric optimal control problem (no points inside the frustum), $|\mathsf{S}_\Omega| = 2 \times 10^4$

respectively. It is clear that for this test problem DAS^2 has a better performance than the other three (uniform, RAR and QRS) sampling strategies. From the left plot of Fig. 8, it is clear that, as the number of samples increases, the relative error of DAS^2 decreases faster than those of the uniform sampling strategy, QRS and RAR. Figure 8(b) shows the error evolution of different sampling strategies and it is seen that DAS^2 eventually yields a smaller error than the other three sampling methods for the same sample size. Figure 9 shows the error evolution of DAS^2 at each adaptivity iteration step k. It is seen that, as k increases, the relative error decreases quickly, implying that DAS^2 is effective.

Figure 10 shows the evolution of the training set ($|\mathsf{S}_\Omega| = 2 \times 10^4$) of DAS^2 with respect to adaptivity iterations $k = 1, 2, 4$ (we use 2000 points in $\mathsf{S}^g_{\Omega,k}$ for visualization), where the initial training set $\mathsf{S}_{\Omega,0}$ consists of uniform samples on Ω. Note that ξ_1 denotes the radius of the circle and ξ_2 is the desired state in Ω_2. We use different colors to identify ξ_2 in Fig. 10. It can be seen that DAS^2 can effectively capture the information of singularity since the data points generated by DAS^2 are concentrated on the area where large residuals are located (see $\mathsf{S}^g_{\Omega,1}$ and $\mathsf{S}^g_{\Omega,2}$ in DAS^2), while RAR is not able to capture the variation in residual well



(a) Solutions $u(\boldsymbol{x}, \boldsymbol{\xi})$ with several realizations of $\boldsymbol{\xi}$, the parametric optimal control problem.



(b) The corresponding absolute point-wise errors for the above solutions, the parametric optimal control problem.

Fig. 11 The results of parametric optimal control problem: the solutions and absolute point wise errors of the uniform sampling method, qusi-random sampling (QRS), RAR and DAS^2 for different realizations. In each subplot, the first line corresponds to $\boldsymbol{\xi} = (0.10, 2.5)$, the second line corresponds to $\boldsymbol{\xi} = (0.20, 2.0)$, the third line corresponds to $\boldsymbol{\xi} = (0.30, 1.5)$, and the fourth line corresponds to $\boldsymbol{\xi} = (0.40, 0.5)$

enough. Finally, nearly uniform samples are generated to augment the training set in DAS^2 because one can obtain a flat residual profile after four adaptivity iterations. Figure 11(a) shows the optimal control solution $u(\boldsymbol{x}, \boldsymbol{\xi})$ obtained using different sampling methods. We choose several different parameters $\boldsymbol{\xi}$ for visualization. For validation, the absolute errors between different sampling strategies and the dolfin-adjoint solver are plotted in Fig. 11(b). It can be seen that the DAS^2 has a better performance than the other three sampling strategies. Table 2 shows the inference time and the relative error for the uniform sampling strategy, QRS, RAR and DAS^2. It is seen that DAS^2 performs much better than the other three sampling strategies especially when the sample size is relatively large.

Table 2 The parametric optimal control problem: comparison of different sampling strategies for neural network surrogate models.

Sampling strategy	$\lvert S_\Omega \rvert$			
	0.5×10^4	1×10^4	1.5×10^4	2×10^4
Uniform (0.1s)	0.92	0.67	0.49	0.29
QRS (0.1s)	0.66	0.63	0.36	0.20
RAR (0.1s)	0.95	0.77	0.37	0.15
DAS^2 (0.1s)	0.89	0.37	0.20	0.06

The relative error of DAS^2 decays faster than other sampling strategies. Once the training of surrogate models is finished, the optimal solution for any parameter can be computed efficiently, which is much faster than the dolfin-adjoint solver (the dolfin-adjoint solver needs 18804 s while the neural network surrogate model based on DAS^2 only needs 0.1 s)

5.4 Surrogate Modeling for Parametric Lid-Driven Cavity Flow Problems with a Varying *Re*

Finally, we consider the lid-driven cavity flow problem governed by the following steady-state incompressible Navier–Stokes equations

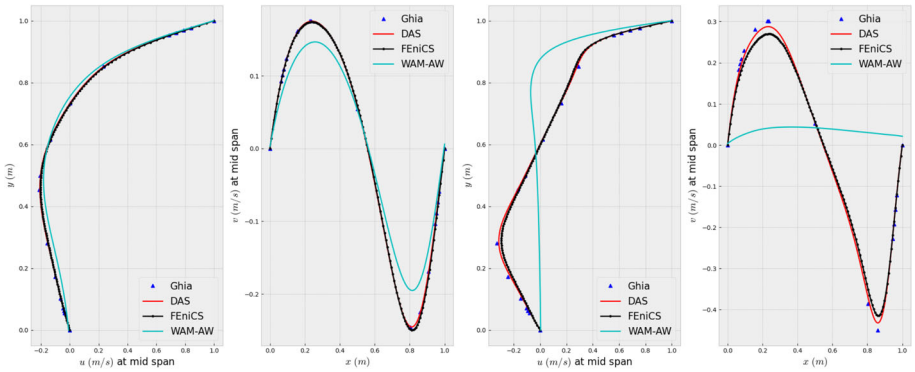
$$\begin{cases} \boldsymbol{u}(\boldsymbol{x},\xi)\cdot\nabla\boldsymbol{u}(\boldsymbol{x},\xi)+\nabla p(\boldsymbol{x},\xi)=\dfrac{1}{Re(\xi)}\Delta\boldsymbol{u}(\boldsymbol{x},\xi) & \text{in } \Omega, \\ \nabla\cdot\boldsymbol{u}(\boldsymbol{x},\xi)=0 & \text{in } \Omega, \\ \boldsymbol{u}(\boldsymbol{x},\xi)=\boldsymbol{g}(\boldsymbol{x},\xi) & \text{on } \partial\Omega, \end{cases} \tag{20}$$

where $\boldsymbol{u}(\boldsymbol{x},\xi)=[u(\boldsymbol{x},\xi),v(\boldsymbol{x},\xi)]^\mathsf{T}$ and $p(\boldsymbol{x},\xi)$ are the flow velocity field and the scalar pressure respectively. Here, we consider a parametric problem in terms of the Reynolds number, where we assume that $Re(\xi)=\xi\in\Omega_p$. The physical domain is $\Omega_s=[0,1]\times[0,1]$. The velocity profile $\boldsymbol{u}=[1,0]^\mathsf{T}$ is imposed on the top boundary ($y=1$ where $\boldsymbol{x}=[x,y]^\mathsf{T}$), and $\boldsymbol{u}=[0,0]^\mathsf{T}$ is imposed on all other boundaries, i.e., for $[\boldsymbol{x},\xi]\in\partial\Omega$

$$\boldsymbol{g}(\boldsymbol{x},\xi)=\begin{cases}[1,0]^\mathsf{T}, y=1;\\ [0,0]^\mathsf{T}, \text{ otherwise.}\end{cases}$$

The lid-driven cavity problem is a benchmark in computational fluid dynamics. However, even for a fixed relatively low Reynolds number, the existing neural-network-based methods are not able to achieve a comparable accuracy with the baseline obtained by classical numerical methods [25]. In this study, we use the proposed DAS^2 method to obtain accurate all-at-once solutions of the parametric lid-driven cavity flow problem with Reynolds numbers from an interval. Two intervals are considered: one is [100, 1000] and the other one is [400, 3200].

We first evaluate the performance of DAS^2 with the non-parametric lid-driven cavity flow problem, where we consider $Re=100,400$. In such a scenario, DAS^2 reduces to DAS. After that, we use DAS^2 to solve the parametric lid-driven cavity flow problem to obtain all-at-once solutions, where we consider $Re\in[100,1000]$ and $[400,3200]$. To measure the quality of DAS^2, we compare DAS^2 with the classical numerical methods presented in the literature [25], the FEniCS solver [1, 44], and the WAM-AW method proposed in a recent literature [30]. Unlike DAS, WAM-AW is an adaptive collocation point movement approach based on interacting particle methods for solving low-regularity PDEs, and we also use this method as a baseline of neural-network-based methods.



(a) $Re = 100$. (b) $Re = 400$.

Fig. 12 The velocity components at the location of mid-span lines for the deterministic lid-driven cavity flow problems, $Re = 100, 400$

Since there are three quantities (u, v and p) to be determined in equation (20), we construct a neural network $\boldsymbol{u}_{\boldsymbol{\theta}}$ with three outputs to represent u, v and p. For the deterministic problem (fixed Reynolds numbers), we choose a five-layer fully connected neural network $\boldsymbol{u}_{\boldsymbol{\theta}}(\boldsymbol{x})$, where each hidden layer has 20 neurons. For KRnet, we set $K = 2$ and the configuration for the affine coupling layers remains the same as the previous experiments. The number of epochs for training both $\boldsymbol{u}_{\boldsymbol{\theta}}(\boldsymbol{x})$ and $p_{\mathsf{KRnet}}(\boldsymbol{x}; \boldsymbol{\theta}_f)$ is set to $N_e = 3000$. The optimizer for training $\boldsymbol{u}_{\boldsymbol{\theta}}(\boldsymbol{x})$ is BFGS, the optimizer for training $p_{\mathsf{KRnet}}(\boldsymbol{x}; \boldsymbol{\theta}_f)$ is ADAM with learning rate 0.0001 and the batch size is set to $m = 100$ (for $Re = 100$) or $m = 500$ (for $Re = 400$). For $Re = 100$, the number of adaptivity iterations is set to $N_{\text{adaptive}} = 5$ with $n_r = 200$, resulting in $|\mathsf{S}_\Omega| = 1000$. For the boundary term, 400 boundary points are uniformly sampled on $\partial\Omega_s$ with 100 points for each edge. To address the incompatibility of boundary conditions, we multiply the boundary residual on the top boundary by a weight function $1 - 2|x - 0.5|$ [50]. For the WAM-AW method proposed in the literature [30], we exactly keep the setup of their work and run their open source code, where the number of collocation points is also set to $|\mathsf{S}_\Omega| = 1000$ and 100 data points for each edge of boundary (these settings are the same as in [30]). For $Re = 400$, the number of adaptivity iterations is set to $N_{\text{adaptive}} = 10$ with $n_r = 500$, resulting in $|\mathsf{S}_\Omega| = 5000$. We uniformly sample 256 points for each edge on the boundary. For the WAM-AW method, we also set $|\mathsf{S}_\Omega| = 5000$ and 256 boundary points are sampled on each edge of the boundary. For all cases, we discretize in space using the Q_3-Q_2 finite element method implemented in FEniCS with a uniform 129×129 grid to obtain a reference solution.

Figure 12 shows the velocity at the location of the mid-span line, which is usually used to assess the accuracy of solutions. Specifically, for $x = 0.5$, we plot the velocity component u with respect to y and for $y = 0.5$ we plot the velocity component v with respect to x. In Fig. 12, we compare the results of DAS, FEniCS, WAM-AW with the benchmark results given in Ghia et. al [25]. It is seen that the results given by DAS are consistent with those given by Ghia and FEniCS, while the results of WAM-AW do not agree with the reference results, especially for $Re = 400$. To further illustrate the effectiveness of DAS, we plot the evolution of random samples during training in Fig. 13, where the left plot shows $\mathsf{S}^g_{\Omega,2}$ and $\mathsf{S}^g_{\Omega,4}$ for $Re = 100$ and the right plot shows $\mathsf{S}^g_{\Omega,2}$ and $\mathsf{S}^g_{\Omega,9}$ for $Re = 400$. It can be seen that DAS yields samples that are consistent with both the problem properties and the

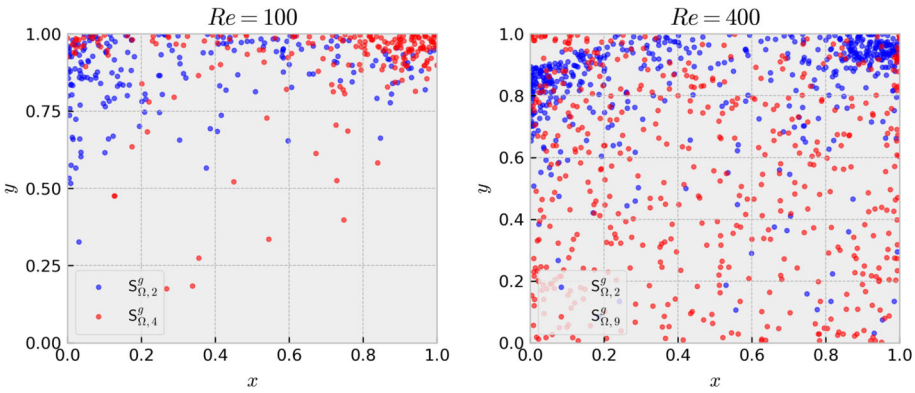
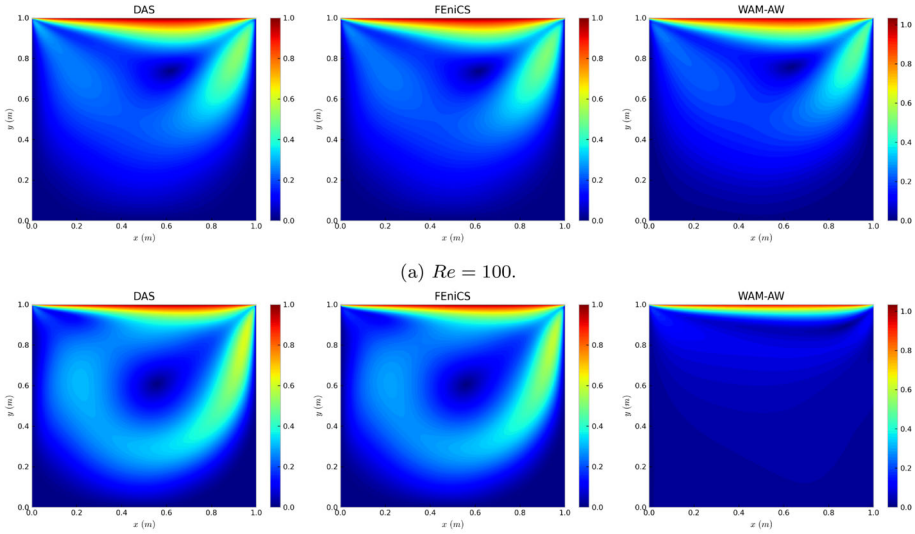


Fig. 13 The random samples in $S^g_{\Omega,k}$ for the deterministic lid-driven cavity flow problems. Left: $S^g_{\Omega,2}$ (blue) and $S^g_{\Omega,4}$ (red) for $Re = 100$; Right: $S^g_{\Omega,2}$ (blue) and $S^g_{\Omega,9}$ (red) for $Re = 400$



(a) $Re = 100$.

(b) $Re = 400$.

Fig. 14 The visualization of $|\boldsymbol{u}| = \sqrt{u^2 + v^2}$ for the deterministic lid-driven cavity flow problems

approximation, where the initial training set consists of random samples generated by Latin hypercube sampling. For example, at $k = 2$, most of the samples in $S^g_{\Omega,2}$ are located in the upper corners, where the velocity field changes abruptly and large residuals occur. As k increases, the residual profile becomes more uniform after the localized information is well captured, which implies that random samples can be added more uniformly. Figure 14 shows the image of $|\boldsymbol{u}| = \sqrt{u^2 + v^2}$, where $Re = 100$ and $Re = 400$ are considered. Compared with the reference solution given by FEniCS, DAS provides an accurate prediction of the flow velocity for $Re = 100$ while WAM-AW has a little loss of accuracy. For $Re = 400$, the results given by DAS are still accurate while the results given by WAM-AW are not physically correct.

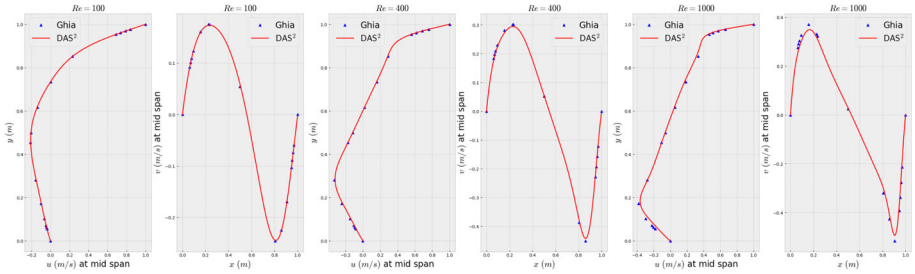


Fig. 15 The velocity components at the location of mid-span lines for surrogate modeling of parametric lid-driven cavity flow problems ($Re \in [100, 1000]$). The results for $Re = 100, 400, 1000$ are chosen for visualization

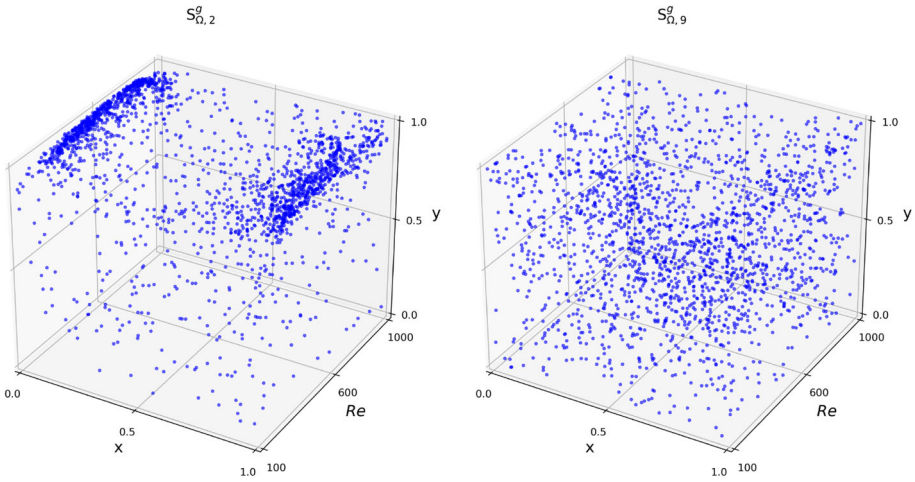


Fig. 16 The random samples in $S^g_{\Omega,k}$ (2000 samples are displayed) for surrogate modeling of parametric lid-driven cavity flow problems, $Re \in [100, 1000]$

Next, we look at the surrogate modeling of parametric lid-driven cavity flow problems. The architecture of neural networks and the setting of adaptive sampling for surrogate modeling need to be modified since solving such parametric problems is more difficult than deterministic ones. We use one five-layer fully connected neural network $\boldsymbol{u}_{\boldsymbol{\theta}}(\boldsymbol{x}, \xi)$ with three outputs to approximate the parametric solutions $u(x, y, \xi)$, $v(x, y, \xi)$, $p(x, y, \xi)$ respectively, where each hidden layer has 32 neurons. For adaptive sampling, we use the joint PDF model induced by KRnet in DAS^2. For KRnet, we set $K = 3$ and take $L = 6$ affine coupling layers. For each affine coupling layer, a two-layer fully connected neural network is employed where each hidden layer has 32 neurons. The number of epochs for training the surrogate model and KRnet is set to $N_e = 5000$. The optimizer for training the surrogate model $\boldsymbol{u}_{\boldsymbol{\theta}}$ is BFGS, and the optimizer for training KRnet is ADAM with a learning rate 0.0001. The number of adaptivity iterations is set to $N_{\text{adaptive}} = 10$ with $n_r = 1 \times 10^4$, resulting in the total number of collocation points $|S_\Omega| = 1 \times 10^5$. For the boundary term, 16384 boundary points are sampled on each edge of the boundary. The batch size is set to $m = 5000$.

Figure 15 shows the velocity profile given by the trained surrogate model $\boldsymbol{u}_{\boldsymbol{\theta}}(\boldsymbol{x}, \xi)$ at the location of the mid-span line for some selected Reynolds numbers $\xi = 100, 400, 1000$. From

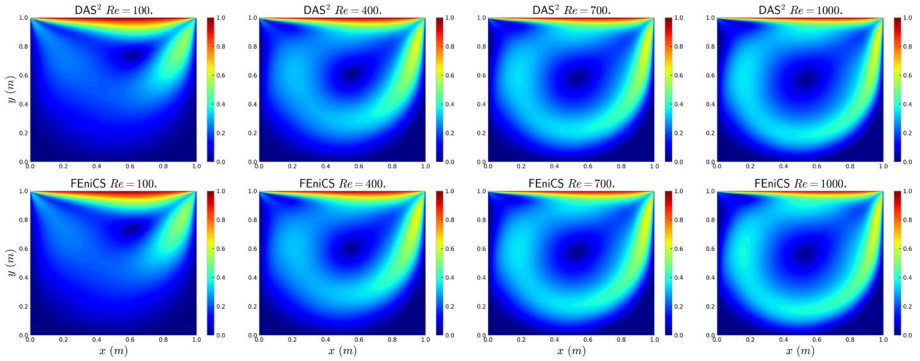


Fig. 17 The visualization of $|\boldsymbol{u}| = \sqrt{u^2 + v^2}$ for surrogate modeling of parametric lid-driven cavity flow problems, $Re \in [100, 1000]$. The l_2 relative errors are 1.5%, 1.1%, 3.1%, 4.8% for $Re = 100, 400, 700, 1000$ respectively

Fig. 15, it is clear that the results of DAS^2 are consistent with Ghia's data [25], implying that our DAS^2 approach is able to provide an accurate surrogate model for fast inference. Figure 16 shows the evolution of the training set ($|\mathsf{S}_\Omega| = 10^5$) of DAS^2 with respect to adaptivity iterations $k = 2, 9$, where the initial training set $\mathsf{S}_{\Omega,0}$ consists of random samples generated by Latin hypercube sampling. $\mathsf{S}^g_{\Omega,2}$ indicates that the residual concentrates on the two upper corners for any $Re \in [100, 1000]$. As the adaptivity iteration k increases, the residual profile becomes more flat as shown by the distribution of $\mathsf{S}^g_{\Omega,9}$, which is expected since more collocation points are added to the two upper corners to reduce the errors over there. Figure 17 shows the image of $|\boldsymbol{u}| = \sqrt{u^2 + v^2}$, where $Re = 100, 400, 700, 1000$ are used for visualization. Here, we again use the Q_3-Q_2 finite element method implemented in FEniCS with a uniform 129×129 grid to obtain the reference solutions for $Re = 100, 400, 700, 1000$. It is seen that DAS^2 provides an accurate prediction of the flow velocity even for $Re = 1000$. The l_2 relative errors, which are evaluated on the 129×129 uniform grid, are 1.5%, 1.1%, 3.1%, 4.8% for $Re = 100, 400, 700, 1000$ respectively. It is worth noting that the inference time of DAS^2 is 0.02 s, while the computation time of FEniCS is 309.94 s to obtain the four solutions for $Re = 100, 400, 700, 1000$.

To further investigate the performance of DAS^2, we consider a more challenging case with $Re \in [400, 3200]$. When the Reynolds number is sufficiently large, the training process becomes quite sensitive to the initialization of neural networks. To alleviate this issue, we use the surrogate model trained for $Re \in [100, 1000]$ as the initialization to train the surrogate model for $Re \in [400, 3200]$. The number of adaptivity iterations is set to $N_{\text{adaptive}} = 5$ with $n_r = 5 \times 10^4$, resulting in a total number $|\mathsf{S}_\Omega| = 2.5 \times 10^5$ of collocation points. Other settings are set to be the same as the case for $Re \in [100, 1000]$. As demonstrated in Fig. 18 and Fig. 19, the surrogate model for $Re \in [400, 3200]$ agrees very well with Ghia's data [25], where the l_2 relative errors are 1.6%, 2.6%, and 6.5% for $Re = 400$, 1000, and 3200, respectively.

6 Conclusions

In this paper, we have developed a deep adaptive sampling approach for surrogate modeling (DAS^2) of parametric differential equations, generalizing the previous work DAS to compli-

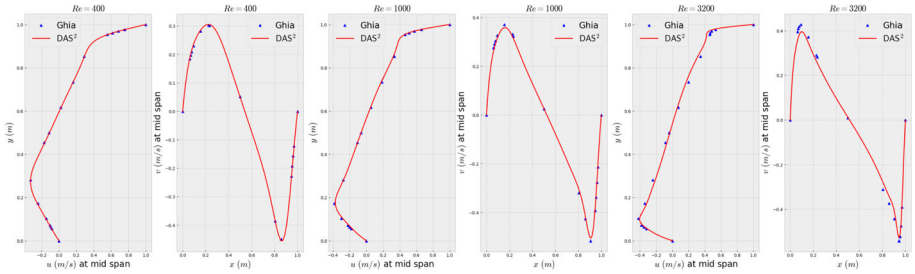


Fig. 18 The velocity components at the location of mid-span lines for surrogate modeling of parametric lid-driven cavity flow problems ($Re \in [400, 3200]$). The results for $Re = 400, 1000, 3200$ are chosen for visualization

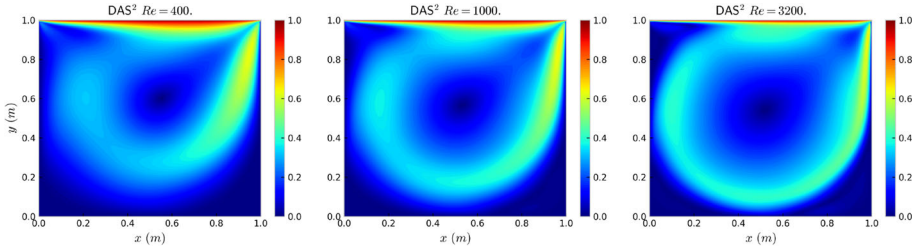


Fig. 19 The visualization of $|\boldsymbol{u}| = \sqrt{u^2 + v^2}$ for surrogate modeling of parametric lid-driven cavity flow problems, $Re \in [400, 3200]$. The l_2 relative errors are $1.6\%, 2.6\%, 6.5\%$ for $Re = 400, 1000, 3200$ respectively

cated parametric settings. It has been shown that DAS^2 not only provides a fast inference for parametric differential equations without labeled data but also yields an accurate prediction for low-regularity problems thanks to the adaptive sampling procedure. Similar to DAS, the framework of DAS^2 also utilizes a deep generative model to generate collocation points that are consistent with the residual-induced distribution. Unlike DAS, DAS^2 handles the low regularity from both spatial and parametric spaces. The joint PDF (marginal PDF) model for both spatial and parametric variables (only the parametric variable), which is induced by the residual, provides effective samples to reduce the statistical errors from the discretization of the loss functional. Based on this, the accuracy of all-at-once solutions of parametric differential equations can be improved significantly.

We pay particular attention to the following observations: First, due to the physics-informed model with adaptive sampling, we have not used any simulation data for the training process. Second, the procedure of adaptive sampling is independent of the structure of the surrogate model unless the model is defined on a certain set of collocation points, and the adaptive sampling strategy can be flexibly chosen based on the nature of the parametric problem. Third, deep generative modeling plays an important role in DAS^2. Deep generative modeling outperforms classical density models or sampling strategies in the sense that it effectively merges density approximation and sample generation for an arbitrary high-dimensional distribution. DAS^2 will find many applications because it provides a general way to improve the training set and any improvement in the model structure can be further refined by adaptive sampling.

Appendix

A Proof of Theorem 1

Proof Since Θ is compact, there exists a δ-net $\bar{\mathsf{N}} = \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_{N_\delta}\}$ with the following property: for all $\boldsymbol{\theta} \in \Theta$, there exists $1 \le i \le N_\delta$ such that $\|\boldsymbol{\theta} - \boldsymbol{\theta}_i\|_\infty \le \delta$ [73]. For a given $\varepsilon \in (0, 1)$, we set $\delta = \varepsilon^2/(4\mathfrak{L})$. Moreover, the number of parameters of $\bar{\mathsf{N}}$ is at most $(4a\mathfrak{L}/\varepsilon^2)^D$. By Assumption 1, for $\boldsymbol{\theta}, \boldsymbol{\nu} \in \Theta$ with $\|\boldsymbol{\theta} - \boldsymbol{\nu}\|_\infty \le \delta$, we have

$$\sup_{\boldsymbol{\theta},\boldsymbol{\nu}} |J_r(u_{\boldsymbol{\nu}}) - J_r(u_{\boldsymbol{\theta}})| + \sup_{\boldsymbol{\theta},\boldsymbol{\nu}} |J_{r,N}(u_{\boldsymbol{\nu}}) - J_{r,N}(u_{\boldsymbol{\theta}})| \le 2\mathfrak{L}\,\|\boldsymbol{\theta} - \boldsymbol{\nu}\|_\infty \le \frac{\varepsilon^2}{2}. \tag{21}$$

For each $1 \le i \le N_\delta$, noting that $J_r(u_{\boldsymbol{\theta}_N^*}) = J_r(u_{\boldsymbol{\theta}_N^*}) - J_r(u_{\boldsymbol{\theta}_i}) + J_r(u_{\boldsymbol{\theta}_i}) - J_{r,N}(u_{\boldsymbol{\theta}_i}) + J_{r,N}(u_{\boldsymbol{\theta}_i}) - J_{r,N}(u_{\boldsymbol{\theta}_N^*}) + J_{r,N}(u_{\boldsymbol{\theta}_N^*})$, it follows that

$$\begin{aligned} J_r(u_{\boldsymbol{\theta}_N^*}) \le\ & |J_r(u_{\boldsymbol{\theta}_N^*}) - J_r(u_{\boldsymbol{\theta}_i})| + |J_r(u_{\boldsymbol{\theta}_i}) \\ & - J_{r,N}(u_{\boldsymbol{\theta}_i})| + |J_{r,N}(u_{\boldsymbol{\theta}_i}) - J_{r,N}(u_{\boldsymbol{\theta}_N^*})| + J_{r,N}(u_{\boldsymbol{\theta}_N^*}). \end{aligned} \tag{22}$$

Next, the infinite set Θ of trainable parameters is discretized by the δ-net, then we use the estimate of loss for the δ-net and combine it with the union bound to give the final estimate. Let $\mathcal{P} : \Theta \mapsto \bar{\mathsf{N}}$ be a projection (in the ℓ_∞ sense) onto $\bar{\mathsf{N}}$, i.e., $\mathcal{P}(\boldsymbol{\theta}) = \bar{\boldsymbol{\theta}}$ where $\bar{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\nu}\in\bar{\mathsf{N}}} \|\boldsymbol{\theta} - \boldsymbol{\nu}\|_\infty$. Consider the following events for $1 \le i \le N_\delta$:

$$\begin{aligned} E_1 &= \{J_r(u_{\boldsymbol{\theta}_N^*}) \le \varepsilon^2 + J_{r,N}(u_{\boldsymbol{\theta}_N^*})\}, \\ E_{2,i} &= \{J_r(u_{\boldsymbol{\theta}_i}) \le \frac{\varepsilon^2}{2} + J_{r,N}(u_{\boldsymbol{\theta}_i})\}, \\ E_{3,i} &= \{\mathcal{P}(\boldsymbol{\theta}_N^*) = \boldsymbol{\theta}_i\}, \\ E_4 &= \{\exists\, i \in \{1, \ldots, N_\delta\} : J_r(u_{\boldsymbol{\theta}_i}) \le \frac{\varepsilon^2}{2} + J_{r,N}(u_{\boldsymbol{\theta}_i}) \text{ and } \mathcal{P}(\boldsymbol{\theta}_N^*) = \boldsymbol{\theta}_i\} \end{aligned}$$

By (21) and (22), we known that if event E_4 occurs, then event E_1 occurs. Indeed, we have $\mathcal{P}(\boldsymbol{\theta}_N^*) = \boldsymbol{\theta}_i$ if E_4 occurs, which implies that $\|\boldsymbol{\theta}_N^* - \boldsymbol{\theta}_i\|_\infty \le \delta$ due to the property of the δ-net. By (22) and using inequality (21) derived from the property of Lipschitz continuity, we can obtain $J_r(u_{\boldsymbol{\theta}_N^*}) \le \varepsilon^2 + J_{r,N}(u_{\boldsymbol{\theta}_N^*})$, which means that E_1 occurs. Hence, we have $E_4 \subseteq E_1$, implying that

$$\mathbb{P}(E_4) \le \mathbb{P}(E_1). \tag{23}$$

According to the definition of δ-net, we have

$$\sum_i \mathbb{P}(E_{3,i}) = 1. \tag{24}$$

By Assumption 2 and the Hoeffding inequality, we obtain

$$\mathbb{P}(E_{2,i}) \ge 1 - \exp\left(\frac{-N_r\varepsilon^4}{2c^2}\right). \tag{25}$$

Noting that $E_4 = \cup_{i=1}^{N_\delta}(E_{2,i} \cap E_{3,i})$, and combining (23), (24) and (25), we have

$$\begin{aligned}\mathbb{P}(E_1) \geq \mathbb{P}(E_4) &= \sum_{i}^{N_\delta} \mathbb{P}(E_{2,i} \cap E_{3,i}) = \sum_{i=1}^{N_\delta} \big(\mathbb{P}(E_{2,i}) + \mathbb{P}(E_{3,i}) - \mathbb{P}(E_{2,i} \cup E_{3,i})\big) \\ &\geq 1 + \sum_{i=1}^{N_\delta} \big(\mathbb{P}(E_{2,i}) - 1\big) \\ &\geq 1 - N_\delta \exp(\frac{-N_r \varepsilon^4}{2c^2}) \\ &\geq 1 - (4a\mathfrak{L}/\varepsilon^2)^D \exp(\frac{-N_r \varepsilon^4}{2c^2}),\end{aligned}$$

which gives that

$$J_r(u_{\boldsymbol{\theta}_N^*}) \leq \varepsilon^2 + J_{r,N}(u_{\boldsymbol{\theta}_N^*})$$

with probability at least $1 - (4a\mathfrak{L}/\varepsilon^2)^D \exp(-N_r\varepsilon^4/2c^2)$. □

B Proof of Theorem 2

Proof Noting that

$$\boldsymbol{\theta}_N^{*,(k+1)} = \arg\min_{\boldsymbol{\theta}} \frac{1}{N_r} \sum_{i=1}^{N_r} \frac{r^2(\boldsymbol{x}^{(i)}, \boldsymbol{\xi}^{(i)}; \boldsymbol{\theta})}{p_{\mathsf{KRnet}}(\boldsymbol{x}^{(i)}, \boldsymbol{\xi}^{(i)}; \boldsymbol{\theta}_f^{*,(k)})}.$$

Since $\boldsymbol{\theta}_N^{*,(k+1)}$ is the optimal solution at the $(k+1)$-th stage and $\boldsymbol{\theta}_N^{*,(k)}$ is used for initialization, we can obtain

$$J_{r,N}(u_{\boldsymbol{\theta}_N^{*,(k+1)}}) = \frac{1}{N_r} \sum_{i=1}^{N_r} \frac{r^2(\boldsymbol{x}^{(i)}, \boldsymbol{\xi}^{(i)}; \boldsymbol{\theta}_N^{*,(k+1)})}{p_{\mathsf{KRnet}}(\boldsymbol{x}^{(i)}, \boldsymbol{\xi}^{(i)}; \boldsymbol{\theta}_f^{*,(k)})} \leq \frac{1}{N_r} \sum_{i=1}^{N_r} \frac{r^2(\boldsymbol{x}^{(i)}, \boldsymbol{\xi}^{(i)}; \boldsymbol{\theta}_N^{*,(k)})}{p_{\mathsf{KRnet}}(\boldsymbol{x}^{(i)}, \boldsymbol{\xi}^{(i)}; \boldsymbol{\theta}_f^{*,(k)})}. \tag{26}$$

Plugging $p_{\mathsf{KRnet}}(\boldsymbol{x}, \boldsymbol{\xi}; \boldsymbol{\theta}_f^{*,(k)}) = c_k r^2(\boldsymbol{x}, \boldsymbol{\xi}; \boldsymbol{\theta}_N^{*,(k)})$ into (26), we have

$$J_{r,N}(u_{\boldsymbol{\theta}_N^{*,(k+1)}}) \leq \frac{1}{c_k}.$$

Noting that $J_{r,N}(u_{\boldsymbol{\theta}_N^{*,(k+1)}})$ is a random variable and taking its expectation, it follows that

$$\mathbb{E}(J_{r,N}(u_{\boldsymbol{\theta}_N^{*,(k+1)}})) \leq \frac{1}{c_k} = \int_\Omega r^2(\boldsymbol{x}, \boldsymbol{\xi}; \boldsymbol{\theta}_N^{*,(k)}) d\boldsymbol{x} d\boldsymbol{\xi} = \mathbb{E}(J_{r,N}(u_{\boldsymbol{\theta}_N^{*,(k)}})),$$

which completes the proof. □

Acknowledgements K. Tang has been supported by the Natural Science Foundation of Hunan Province (2024JJ6003). J. Zhai is supported by the start-up fund of ShanghaiTech University (2022F0303-000-11). X. Wan has been supported by NSF grant DMS-1913163. C. Yang has been supported by NSFC grant 12131002.

Data Availability The datasets and code of this study are available at https://github.com/MJfadeaway/DAS-2

Declarations

Conflict of interest The authors declare that they have no Conflict of interest.

References

1. Alnæs, M., Blechta, J., Hake, J., Johansson, A., Kehlet, B., Logg, A., Richardson, C., Ring, J., Rognes, M.E., Wells, G.N.: The FEniCS Project Version 1.5. Archive of Numerical Software **3**(100) (2015)
2. Anagnostopoulos, S.J., Toscano, J.D., Stergiopulos, N., Karniadakis, G.E.: Residual-based attention and connection to information bottleneck theory in PINNs. arXiv preprint arXiv:2307.00379v1 (2023)
3. Berg, J., Nyström, K.: A unified deep artificial neural network approach to partial differential equations in complex geometries. Neurocomputing **317**, 28–41 (2018)
4. Bonito, A., Cohen, A., DeVore, R., Guignard, D., Jantsch, P., Petrova, G.: Nonlinear methods for model reduction. ESAIM Math. Model Numer. Anal. **55**(2), 507–531 (2021)
5. Boyaval, S., Le Bris, C., Lelievre, T., Maday, Y., Nguyen, N.C., Patera, A.T.: Reduced basis techniques for stochastic problems. Arch. Comput. Methods Eng. **17**, 435–454 (2010)
6. Chakraborty, S., Adhikari, S., Ganguli, R.: The role of surrogate models in the development of digital twins of dynamic systems. Appl. Math. Model. **90**, 662–681 (2021)
7. Chaturantabut, S., Sorensen, D.C.: Nonlinear model reduction via discrete empirical interpolation. SIAM J. Sci. Comput. **32**(5), 2737–2764 (2010)
8. Chen, X., Cen, J., Zou, Q.: Adaptive trajectories sampling for solving PDEs with deep learning methods. arXiv preprint arXiv:2303.15704 (2023)
9. Chen, Y., Jiang, J., Narayan, A.: A robust error estimator and a residual-free error indicator for reduced basis methods. Comput. Math. Appl. **77**(7), 1963–1979 (2019)
10. Cohen, A., Farhat, C., Somacal, A., Maday, Y.: Nonlinear compressive reduced basis approximation for PDE's. HAL preprint, 04031976 (2023)
11. Cui, T., Marzouk, Y., Willcox, K.: Scalable posterior approximations for large-scale Bayesian inverse problems via likelihood-informed parameter and state reduction. J. Comput. Phys. **315**, 363–387 (2016)
12. Daw, A., Bu, J., Wang, S., Perdikaris, P., Karpatne, A.: Mitigating propagation failures in physics-informed neural networks using retain-resample-release (r3) sampling. arXiv preprint arXiv:2207.02338 (2023)
13. De Boer, P.T., Kroese, D.P., Mannor, S., Rubinstein, R.Y.: A tutorial on the cross-entropy method. Ann. Op. Res. **134**(1), 19–67 (2005)
14. De Ryck, T., Mishra, S.: Error analysis for physics-informed neural networks (PINNs) approximating Kolmogorov PDEs. Adv. Comput. Math. **48**(6), 79 (2022)
15. De Ryck, T., Mishra, S.: Generic bounds on the approximation error for physics-informed (and) operator learning. Adv. Neural Inf. Process. Syst. **35**, 10945–10958 (2022)
16. Dinh, L., Sohl-Dickstein, J., Bengio, S.: Density estimation using real NVP. arXiv preprint arXiv:1605.08803 (2016)
17. Yu, B.: The deep Ritz method: a deep learning-based numerical algorithm for solving variational problems. Commun. Math. Stat. **6**(1), 1–12 (2018)
18. Elman, H.C., Liao, Q.: Reduced basis collocation methods for partial differential equations with random coefficients. SIAM/ASA J. Uncertain. Quantif. **1**(1), 192–217 (2013)
19. Feng, Y., Tang, K., Wan, X., Liao, Q.: Dimension-reduced KRnet maps for high-dimensional Bayesian inverse problems. arXiv preprint arXiv:2303.00573 (2023)
20. FuksHamdi, O., Tchelepi, T.: Limitations of physics informed machine learning for nonlinear two-phase transport in porous media. J. Mach. Learn. Model Comput. **1**(1), 19–37 (2020)
21. Gao, H., Sun, L., Wang, J.X.: Phygeonet: physics-informed geometry-adaptive convolutional neural networks for solving parameterized steady-state PDEs on irregular domain. J. Comput. Phys. **428**, 110079 (2021)
22. Gao, W., Wang, C.: Active learning based sampling for high-dimensional nonlinear partial differential equations. J. Comput. Phys. **475**, 111848 (2023)
23. Gao, Z., Tang, T., Yan, L., Zhou, T.: Failure-informed adaptive sampling for PINNs, part II: combining with re-sampling and subset simulation. Commun. Appl. Math. Comput. (2023)
24. Gao, Z., Yan, L., Zhou, T.: Failure-informed adaptive sampling for pinns. SIAM J. Sci. Comput. **45**(4), A1971–A1994 (2023)
25. Ghia, U., Ghia, K.N., Shin, C.: High-Re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method. J. Comput. Phys. **48**(3), 387–411 (1982)

26. Ghosh, S., Anantha Padmanabha, G., Peng, C., Andreoli, V., Atkinson, S., Pandita, P., Vandeputte, T., Zabaras, N., Wang, L.: Inverse aerodynamic design of gas turbine blades using probabilistic machine learning. J. Mech. Des. **144**(2) (2022)
27. Han, J., Cai, Z., Wu, Z., Zhou, X.: Residual-quantile adjustment for adaptive training of physics-informed neural network. In: Proceedings of 2022 IEEE International Conference on Big Data (Big Data), pp. 921–930. IEEE Computer Society, Osaka, Japan (2022)
28. Han, J., Jentzen, A., Weinan, E.: Solving high-dimensional partial differential equations using deep learning. Proc. Natl. Acad. Sci. **115**(34), 8505–8510 (2018)
29. Hao, Z., Liu, S., Zhang, Y., Ying, C., Feng, Y., Su, H., Zhu, J.: Physics-informed machine learning: a survey on problems, methods and applications. arXiv preprint arXiv:2211.08064 (2022)
30. Hou, J., Li, Y., Ying, S.: Enhancing PINNs for solving PDEs via adaptive collocation point movement and adaptive loss weighting. Nonlinear Dyn. **111**(16), 15233–15261 (2023)
31. Jagtap, A.D., Kharazmi, E., Karniadakis, G.E.: Conservative physics-informed neural networks on discrete domains for conservation laws: applications to forward and inverse problems. Comput. Methods Appl. Mech. Eng. **365**, 113028 (2020)
32. Jiao, Y., Li, D., Lu, X., Yang, J.Z., Yuan, C.: GAS: A Gaussian mixture distribution-based adaptive sampling method for PINNs. arXiv preprint arXiv:2303.15849 (2023)
33. Jorge, N., Stephen, J.W.: Numerical Optimization. Spinger (2006)
34. Kapteyn, M.G., Knezevic, D.J., Huynh, D., Tran, M., Willcox, K.E.: Data-driven physics-based digital twins via a library of component-based reduced-order models. Int. J. Numeri. Methods Eng. **123**(13), 2986–3003 (2022)
35. Kärcher, M., Tokoutsi, Z., Grepl, M.A., Veroy, K.: Certified reduced basis methods for parametrized elliptic optimal control problems with distributed controls. J. Sci. Comput. **75**(1), 276–307 (2018)
36. Karniadakis, G.E., Kevrekidis, I.G., Lu, L., Perdikaris, P., Wang, S., Yang, L.: Physics-informed machine learning. Nat. Rev. Phys. **3**(6), 422–440 (2021)
37. Kingma, D., Ba, J.: Adam: A Method for Stochastic Optimization. In: International Conference on Learning Representations (ICLR) (2015)
38. Kingma, D.P., Dhariwal, P.: Glow: Generative flow with invertible 1x1 convolutions. In: Advances in Neural Information Processing Systems, pp. 10215–10224 (2018)
39. Li, J., Marzouk, Y.M.: Adaptive construction of surrogates for the Bayesian solution of inverse problems. SIAM J. Sci. Comput. **36**(3), A1163–A1186 (2014)
40. Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., Anandkumar, A.: Neural operator: Graph kernel network for partial differential equations. arXiv preprint arXiv:2003.03485 (2020)
41. Li, Z., Kovachki, N.B., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., Anandkumar, A.: Fourier Neural Operator for Parametric Partial Differential Equations. In: International Conference on Learning Representations (2021)
42. Li, Z., Zheng, H., Kovachki, N., Jin, D., Chen, H., Liu, B., Azizzadenesheli, K., Anandkumar, A.: Physics-informed neural operator for learning partial differential equations. arXiv preprint arXiv:2111.03794 (2021)
43. Liao, Q., Li, J.: An adaptive reduced basis ANOVA method for high-dimensional Bayesian inverse problems. J. Comput. Phys. **396**, 364–380 (2019)
44. Logg, A., Mardal, K.A., Wells, G.: Automated solution of differential equations by the finite element method: The FEniCS book, vol. 84. Springer (2012)
45. Lu, L., Jin, P., Pang, G., Zhang, Z., Karniadakis, G.E.: Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. Nat. Mach. Intell. **3**(3), 218–229 (2021)
46. Lu, L., Meng, X., Mao, Z., Karniadakis, G.E.: DeepXDE: a deep learning library for solving differential equations. SIAM Rev. **63**(1), 208–228 (2021)
47. Mao, Z., Jagtap, A.D., Karniadakis, G.E.: Physics-informed neural networks for high-speed flows. Comput. Methods Appl. Mech. Eng. **360**, 112789 (2020)
48. Mitusch, S., Funke, S., Dokken, J.: dolfin-adjoint 2018.1: automated adjoints for FEniCS and Firedrake. J. Open Sour. Softw. **4**(38), 1292 (2019)
49. Negri, F., Rozza, G., Manzoni, A., Quarteroni, A.: Reduced basis method for parametrized elliptic optimal control problems. SIAM J. Sci. Comput. **35**(5), A2316–A2340 (2013)
50. NVIDIA: Nvidia modulus: physics-informed neural networks. https://developer.nvidia.com/modulus (2021). Accessed: 2023-06-26
51. Peng, W., Zhou, W., Zhang, X., Yao, W., Liu, Z.: Rang: A residual-based adaptive node generation method for physics-informed neural networks. arXiv preprint arXiv:2205.01051 (2022)
52. Quarteroni, A., Manzoni, A., Negri, F.: Reduced basis methods for partial differential equations: an introduction, vol. 92. Springer (2015)

53. Raissi, M., Perdikaris, P., Karniadakis, G.E.: Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J. Comput. Phys. **378**, 686–707 (2019)
54. Raissi, M., Yazdani, A., Karniadakis, G.E.: Hidden fluid mechanics: learning velocity and pressure fields from flow visualizations. Science **367**(6481), 1026–1030 (2020)
55. Rubinstein, R.Y., Kroese, D.P.: The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation and machine learning. Springer (2013)
56. Sheng, H., Yang, C.: PFNN: A penalty-free neural network method for solving a class of second-order boundary-value problems on complex geometries. J. Comput. Phys. p. 110085 (2020)
57. Sirignano, J., Spiliopoulos, K.: DGM: a deep learning algorithm for solving partial differential equations. J. Comput. Phys. **375**, 1339–1364 (2018)
58. Stuart, A.M.: Inverse problems: a Bayesian perspective. Acta Numer. **19**, 451–559 (2010)
59. Subramanian, S., Kirby, R.M., Mahoney, M.W., Gholami, A.: Adaptive self-supervision algorithms for physics-informed neural networks. arXiv preprint arXiv:2207.04084 (2022)
60. Sun, L., Gao, H., Pan, S., Wang, J.X.: Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data. Comput. Methods Appl. Mech. Eng. **361**, 112732 (2020)
61. Tang, K., Wan, X., Liao, Q.: Deep density estimation via invertible block-triangular mapping. Theor. Appl. Mech. Lett. **10**, 143 (2020)
62. Tang, K., Wan, X., Liao, Q.: Adaptive deep density approximation for Fokker-Planck equations. J. Comput. Phys. **457**, 111080 (2022)
63. Tang, K., Wan, X., Yang, C.: DAS-PINNs: a deep adaptive sampling method for solving high-dimensional partial differential equations. J. Comput. Phys. **476**, 111868 (2023)
64. Tang, K., Zhai, J., Wan, X., Yang, C.: Adversarial adaptive sampling: unify PINN and optimal transport for the approximation of PDEs. arXiv preprint arXiv:2305.18702 (2023)
65. Torzoni, M., Tezzele, M., Mariani, S., Manzoni, A., Willcox, K.E.: A digital twin framework for civil engineering structures. Comput. Methods Appl. Mech. Eng. **418**, 116584 (2024)
66. Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., et al.: SciPy 1.0: fundamental algorithms for scientific computing in Python. Nat. Methods **17**(3), 261–272 (2020)
67. Wan, X., Tang, K.: Augmented KRnet for density estimation and approximation. arXiv preprint arXiv:2105.12866 (2021)
68. Wang, S., Wang, H., Perdikaris, P.: Learning the solution operator of parametric partial differential equations with physics-informed DeepONets. Sci. Adv. **7**(40), eabi8605 (2021)
69. Wang, S., Yu, X., Perdikaris, P.: When and why PINNs fail to train: a neural tangent kernel perspective. J. Comput. Phys. **449**, 110768 (2022)
70. Wang, X., Yin, P., Zhang, B., Yang, C.: AONN-2: An adjoint-oriented neural network method for PDE-constrained shape optimization. arXiv preprint arXiv:2309.08388 (2023)
71. Wen, Y., Vanden-Eijnden, E., Peherstorfer, B.: Coupling parameter and particle dynamics for adaptive sampling in neural galerkin schemes. arXiv preprint arXiv:2306.15630 (2023)
72. Winkler, C., Worrall, D., Hoogeboom, E., Welling, M.: Learning likelihoods with conditional normalizing flows. arXiv e-prints pp. arXiv–1912 (2019)
73. Wright, J., Ma, Y.: High-dimensional data analysis with low-dimensional models: principles, computation, and applications. Cambridge University Press (2021)
74. Wu, C., Zhu, M., Tan, Q., Kartha, Y., Lu, L.: A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks. Comput. Methods Appl. Mech. Eng. **403**, 115671 (2023)
75. Xia, Y., Zabaras, N.: Bayesian multiscale deep generative model for the solution of high-dimensional inverse problems. J. Comput. Phys. **455**, 111008 (2022)
76. Xiu, D.: Numerical methods for stochastic computations: a spectral method approach. Princeton University Press (2010)
77. Xiu, D.: Stochastic collocation methods: a survey. Handbook of Uncertainty Quantification. Springer, Cham (2016)
78. Xiu, D., Karniadakis, G.E.: The Wiener-Askey polynomial chaos for stochastic differential equations. SIAM J. Sci. Comput. **24**(2), 619–644 (2002)
79. Yin, P., Xiao, G., Tang, K., Yang, C.: AONN: An adjoint-oriented neural network method for all-at-once solutions of parametric optimal control problems. arXiv preprint arXiv:2302.02076 (2023)
80. Yu, T., Yong, H., Liu, L., et al.: MCMC-PINNs: a modified Markov chain Monte-Carlo method for sampling collocation points of PINNs adaptively. Authorea Preprints (2023)

81. Zhai, J., Dobson, M., Li, Y.: A deep learning method for solving Fokker-Planck equations. In: Proceedings of the 2nd Mathematical and Scientific Machine Learning Conference, *Proceedings of Machine Learning Research*, vol. 145, pp. 568–597. PMLR, Princeton University, Princeton, NJ, USA (2022)
82. Zhu, Y., Zabaras, N.: Bayesian deep convolutional encoder-decoder networks for surrogate modeling and uncertainty quantification. J. Comput. Phys. **366**, 415–447 (2018)
83. Zhu, Y., Zabaras, N., Koutsourelakis, P.S., Perdikaris, P.: Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. J. Comput. Phys. **394**, 56–81 (2019)