12th IFAC International Symposium on
Advanced Control of Chemical Processes
July 14-17, 2024. Toronto, Canada

# Sensitivity-based Adaptive Sampling for Physics-Informed Neural Networks

**Shuji Chang** * **Piyush Agarwal** ** **Chris McCready** **
**Luis Ricardez-Sandoval** * **Hector Budman** *

* *Department of Chemical Engineering, University of Waterloo, Waterloo, Canada, N2L3G1*
** *Sartorius Corporate Research, Sartorius, Toronto, ON, Canada*

**Abstract:** For training the Physics-Informed Neural Networks (PINNs), the allocation of collocation points in the geometric domain plays a pivotal role in determining the model's performance. We present a novel sampling method tailored for optimal point allocation in PINNs. The method involves an initial meshing of the domain, followed by a calculation of the sensitivity matrix relating the losses for each mesh element to local changes in the locations of the training points. Subsequently, based on the principles of A-optimal experimental design, the sampling probability is dynamically redistributed over the domain. In this way, areas of high sensitivity and corresponding physical significance receive further representation in the training data. Preliminary results illustrate the effectiveness of the proposed method when applied to the problem of developing flow between two parallel plates. This sensitivity-based sampling (SBS) is shown to increase the overall precision of PINNs since it can specifically capture sharp gradients in critical points within the geometric domain.

*Keywords:* Deep Learning, PINNs, Design of Experiments (DOE), Optimal Design, Surrogate, Predictive Modeling, Adaptive Sampling

## 1. INTRODUCTION

Deep learning modeling approaches have recently gained attention for solving physical transport problems with spatio-temporal variations. A major challenge for the training of deep learning models is the need for a large amount of data for training. To address this challenge, Raissi et al. (2019) proposed a deep learning method called Physics-Informed Neural Networks (PINNs). This method uses prior knowledge, *e.g.*, first-principles mass and energy balances, experimental data, initial condition, boundary conditions, etc., thus partially overcoming the limitations of data scarcity by constraining the Neural Network (NN) model (Raissi et al. (2019)). The training of PINNs is based on the minimization of a loss function related to the satisfaction of some or all of the following: **i**- the partial differential equations defining the target problem in continuous time or spatial domains; **ii**- initial/boundary conditions at specific time/space coordinates **iii**- *in-silico* data from simulations, such as CFD; and **iv**- measured data from experiments (Eq. 1).

$$MSE_{PDE} = \frac{1}{N_{PDE}} \sum_{i=1}^{N_{PDE}} (f(t_i, x_i))^2$$

$$MSE_{IC/BC} = \frac{1}{N_{IC/BC}} \sum_{i=1}^{N_{IC/BC}} (u_{NN}(t_i, x_i) - u_i)^2$$

$$MSE_{simulation} = \frac{1}{N_{sim.}} \sum_{i=1}^{N_{sim.}} (u_{NN}(t_i, x_i) - u_i)^2$$

$$MSE_{experiment} = \frac{1}{N_{exp.}} \sum_{i=1}^{N_{exp.}} (u_{NN}(t_i, x_i) - u_i)^2$$

$$\begin{aligned} Loss = {} & \theta_{PDE} * MSE_{PDE} + \theta_{IC/BC} * MSE_{IC/BC} \\ & + \theta_{simulation} * MSE_{simulation} \\ & + \theta_{experiment} * MSE_{experiment} \end{aligned} \quad (1)$$

$t_i$ and $x_i$ define the coordinates of the collocation points in time and spatial domains used for training; $f$ is a linear summation of each term of the objective PDEs where the variables and their nonlinear operations (*e.g.*, divergence, gradient, Laplacian operator) therein are replaced by the output values ($u_{NN}$) of the neural network and its non-linear derivatives using automatic differentiation; $\theta$ is a weight for a specific loss term.

PINNs are trained by minimizing the total loss function (refer to Eq. 1). This problem poses important challenges. A key challenge is related to the collocation points to be chosen for PINN training. The proper location of the points for training the PINN is key for obtaining a model that provides accurate predictions over the entire domain of the problem. Conventionally, the collocation points are chosen uniformly across the domain, which means assigning the same importance to every region within the domain. This may not be optimal in terms of convergence and model accuracy. A second challenge is that different terms in the loss function may have divergent/convergence

⋆ This research is supported by Sartorius and Mitacs through the Mitacs-Accelerate Program.

properties, leading to suboptimal training results. This necessitates a judicious choice of weights for each term within the loss function to avoid emphasizing one term over another.

Regarding the distribution of collocation points, some studies developed sampling methods tailored to optimize the locations of training points for improved convergence and accuracy. For example, uniform, random, and adaptive sampling methods have been reported in different studies (Nabian et al. (2021); Wu et al. (2023); Fang et al. (2023)) Importance sampling or residual-based sampling is a very popular approach where sampling probabilities are assigned to each region in proportion to the loss thus resulting in improved convergence and accuracy (Nabian et al. (2021); Wu et al. (2023); Mao and Meng (2023)).

However, those methods only consider the effect of collocation points on the residual with the model available at the current epoch. In contrast, in this work, we propose a methodology that takes into account changes in the loss function resulting from models obtained at different epochs. Our approach is motivated by the recognition that changing the collocation points will result, through backpropagation, in the change of the weights and consequently of the trained models obtained at two different epochs. This approach enables the influence of points at any location on the resulting residuals or errors at other locations across the geometric domain, which it is not possible with importance sampling methods that rely solely on a current model for sampling. The results of the case study will illustrate the superiority of our approach as compared to conventional sampling methods.

The strategy proposed in the current study is based on a combination of sensitivity analysis and an optimal experimental design approach where the sensitivity is defined as follows:

$$\begin{aligned} \text{Sensitivity} &= \frac{dLoss}{dX} \\ &= \sqrt{\left(\frac{dLoss_{x,y}}{dx}\right)^2 + \left(\frac{dLoss_{x,y}}{dy}\right)^2} \end{aligned} \tag{2}$$

We hypothesize that, within the entire domain, assigning higher sampling probabilities to regions with higher sensitivity of the loss function with respect to changes of their coordinates will generate a training set with higher information density. This will enhance the local and overall training efficiency of the model. Moreover, following the normalization of the variables, we show that training with a high-information-density dataset ensures that each part of the loss function is adequately considered, thereby eliminating overfitting/underfitting related to any specific objective.

Our sensitivity-based sampling (SBS) consists of three steps: 1. Mesh the domain into elements (it is not possible to consider sensitivity for each point because of computational complexity); 2. Compute the sensitivity and construct a sensitivity matrix $\mathbf{S}$ for all the mesh elements; 3. Based on this sensitivity matrix, allocate sampling probabilities to each mesh element and allocate points accordingly.

This work is organized as follows: Section 2 reviews PINNs' theory. Section 3 details the proposed methodology. Section 4 presents the case study. Section 5 discusses the case study results, and Section 6 presents concluding remarks.

## 2. PRELIMINARIES

### *2.1 Physics-Informed Neural Networks*

Physics-Informed Neural Networks (PINNs) seamlessly integrate physical principles, represented as ordinary (ODEs) or partial differential equations (PDEs), into a supervised deep learning algorithm. The neural networks (NN) are trained to satisfy fundamental physical constraints including mass and momentum balances and boundary and initial conditions. This is achieved by training the network to minimize loss functions involving all these constraints (Eq. 1).

A typical PINN formulation (Raissi et al. (2019)) consists of two parts: a trainable approximator network for the solution, and a residual network for reconstructing the physical laws during the training procedure. The neural network inputs are spatial $x, y, z$ and time ($t$) coordinates from the simulation domain and the outputs are field values (*e.g.*, velocity $u$, pressure $p$, temperature $T$) at the given coordinates. The field values generated by the approximator inform the residual network which encodes the PDEs through the automatic differentiation capability of the neural network. This operation leverages the chain rule in neurons coupled with nonlinear computations, making it fast and efficient. The reconstructed partial differential equations are enforced by the different terms in the loss functions which are weighted and summed up with other loss terms (Eq. 1).

## 3. PROPOSED METHODOLOGY

The key element of our method lies in the sensitivity analysis we proposed, along with the sampling probability allocation based on an optimal experimental design approach to find the locations of the sampling points. The motivation for this approach is as follows. Let us assume that at a particular learning epoch during the training, the loss function is equal to $\mathcal{L}^0$, and the sensitivity matrix composed of local sensitivities as defined Eq. 2 is $S$. Then, following a simple Taylor expansion of the loss function about the value of the loss $\mathcal{L}^0$ achieved at a current epoch, the change in the loss with respect to changes in the positions of the sampling points is as follows:

$$\mathcal{L} - \mathcal{L}^0 = \mathbf{S} \cdot \delta X \tag{3}$$

where $\delta X$ is a vector of changes in the point's position with respect to the coordinates used to obtain a loss of $\mathcal{L}^0$. Then, since the final goal is to drive the final loss ($\mathcal{L} \to 0$), from Eq. 3 we obtain:

$$-\mathcal{L}^0 = \mathbf{S} \cdot \delta X \tag{4}$$

This problem can be viewed as a regression problem with respect to the changes in the sampling point positions where the latter can be interpreted as the regression parameters. Then, optimal design approaches can be applied

to find the changes in the positions of the collocation points to be used for training. The reason that A-optimal ($\max \mathbf{Tr}(\mathbf{S}^T\mathbf{S})$) design is preferred here over other optimality criteria, is due to the screening objective relevant to the current work. A-optimality focuses on penalizing subsets of factors, *i.e.*, in our case different locations within the geometric domain, that have the most impact on the loss function. The expectation for finding such locations is two-fold: the overall loss will be minimized and particular locations with large sensitivity will be emphasized.

### *3.1 Calculation of local and global sensitivity matrices*

We propose two methods for the sensitivity-based analysis: **i**- local sensitivity-based or **ii**- global sensitivity-based.

**i**- The elements of the local sensitivity matrix are the derivatives of the local loss function at the center of a mesh element with respect to changes in the location of the corresponding collocation point *in situ*. These derivatives ($\mathbf{S}_{i,i} = d\mathcal{L}_i/dX_i$) can be calculated by automatic differentiation to compose a diagonal sensitivity matrix, $\mathbf{S}$.

**ii**- The global sensitivity considers the network's weight updates ($\Delta w$) through a user prescribed number of training epochs. This procedure involves implementing one or more steps of back-propagation operations. The rationale for global sensitivity analysis is that the changes in the location of a collocation point will affect in subsequent epochs all the weights in the network. Accordingly, the change in the location of a particular collocation point at a particular epoch will affect the loss function at all other points of the domain in subsequent epochs. This clearly contrasts with the local sensitivity analysis where the change in location of a particular point only affects the local sensitivity, i.e. the sensitivity at that particular point and at that particular epoch. (Eq. 5: illustrates the basic weight update operation in a single layer NN, where $\alpha$ is the learning rate, $N$ is the batch size, $\sigma$, and $\sigma'$ are activation function, and its derivative, $b$ is the bias.).

$$\begin{aligned}
&\Delta w = -\alpha \frac{\partial \mathcal{L}}{\partial w} \\
&\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2 \\
&y_i = \sigma(w \cdot x_i + b) \\
&\frac{\partial \mathcal{L}}{\partial w} = \frac{-2}{N}\sum_{i=1}^{N}\left[(\hat{y}_i - y_i) \times \sigma'(w \cdot x_i + b) \times x_i\right]
\end{aligned} \tag{5}$$

Due to the weight updates, it is not possible to use the automatic differentiation method of the network between two different epochs. Instead, we perform a numerical evaluation of the global sensitivity by calculating the change in the losses between two different epochs with respect to the changes in the positions corresponding to these losses. The number of epochs in the future that are considered for calculation of the global sensitivity following changes in locations of training points is thereafter referred to as a prediction horizon $h$ and it is considered as a user-defined parameter in this proposed method. The resultant loss changes are then normalized by each perturbation $(\delta x, \delta y)$ to obtain the resulting sensitivities as :

$$\begin{aligned}
\text{Sensitivity}_{i,j} &= \frac{\partial Loss_i}{\partial X_j} \\
&= \sqrt{\frac{(\mathcal{L}^{k+h}(x_i + \delta x_i, y_i + \delta y_i) - \mathcal{L}^k(x_j, y_j))^2}{(\delta x_j)^2 + (\delta y_j)^2}}
\end{aligned} \tag{6}$$

Based on the calculated sensitivities, a square global sensitivity matrix can be formulated as follows, $\mathbf{S}$ ($\mathbf{S}_{i,j} = \partial\mathcal{L}_i/\partial X_j$, $i$ and $j$ are the indexes of row (loss) and column (perturbation) respectively).

### *3.2 Proposed sampling based on sensitivity matrices*

Utilizing the sensitivity matrices derived from the preceding step, an A-optimal experimental design criterion is applied to determine the sampling probability allocation. Let the sensitivity matrix, $\mathbf{S}_{i,j} = \partial\mathcal{L}_i/\partial X_j$. Then, a positive definite matrix can be obtained, $(\mathbf{S}^T\mathbf{S})_{i,j} = (\partial\mathcal{L}_i/\partial X_j)^2$ (Fisher information matrix). Then, maximization of the trace of this matrix, *i.e.*, $\mathbf{S}$ is equivalent to maximization of sensitivity.

$$\mathbf{Tr}\left(\mathbf{S}^T\mathbf{S}\right) = \sum_{j=1}^{m}\sum_{i=1}^{m}\left(\frac{\partial Loss_i}{\partial X_j}\right)^2 \tag{7}$$

As decision variables for the optimization problem we use a fraction $\lambda$ and formulate accordingly the following optimization problem:

$$\begin{aligned}
\max_{\lambda_i} \quad & \mathbf{Tr}\left(\sum_{i=1}^{m}\lambda_i \mathbf{v}_i^T\mathbf{v}_i\right) \\
\text{s.t.} \quad & \lambda^{UB} \geq \lambda_i \geq \lambda^{LB}, \forall\ i = 1, ..., m \\
& \sum_{i=1}^{m}\lambda_i = 1
\end{aligned} \tag{8}$$

$\mathbf{v}_i$ is the row vector of sensitivity matrix $\mathbf{S}$, with dimensions $m \times m$; $m$ also represents the number of regions obtained through meshing. This parameter can be adjusted to control the resolution of the mesh depending on the complexity of the specific problem under study.

For the local sensitivity approach for which the sensitivity matrix is diagonal, the trace is computed as follows:

$$\mathbf{Tr}\left(\sum_{i=1}^{m}\lambda_i \mathbf{v}_i^T\mathbf{v}_i\right) = \sum_{i=1}^{m}\lambda_i\left(\frac{dLoss_i}{dX_i}\right)^2 \tag{9}$$

The derived fraction, $\lambda_i$, serves as the sampling probability ($p_i$) for position $i$.

For the global sensitivity approach, the trace to be maximized is as follows:

$$\mathbf{Tr}\left(\sum_{i=1}^{m}\lambda_i \mathbf{v}_i^T\mathbf{v}_i\right) = \sum_{j=1}^{m}\sum_{i=1}^{m}\lambda_i\left(\frac{\partial Loss_i}{\partial X_j}\right)^2 \tag{10}$$

To obtain the optimal sampling probabilities ($p_j$) for each location $j$, the fraction of points $\lambda_i$ allocated to each row $i$ is further distributed proportionally across columns $j$ based on their respective sensitivity values $\partial\mathcal{L}_i/\partial X_j$.

$$p_j = \sum_{i=1}^{m}\lambda_i \frac{\partial\mathcal{L}_i/\partial X_j}{\sum_{j=1}^{m}\partial\mathcal{L}_i/\partial X_j} \tag{11}$$

A notable difference between the local approach and the global approach is that the former can only allocate

sampling probabilities at one position that affects the loss at that position whereas the latter approach can allocate sampling probabilities at one position that affects that same position or another different position in the domain. For instance, the global approach can calculate the sensitivity of the boundary condition loss with respect to coordinates' changes in collocation points located away from the boundary.

**Algorithm 1** Sensitivity-Based Adaptive Sampling Global Approach (SBS-G)

Generate $m$ grids for the training data set(s) uniformly and have their geometric centers $\{X_j\}_{j=1}^{m}$, and initialize the sampling probability $p_j = 1/m$ for each grid.
Set up the model and specify the maximum epoch number $E$, the size of mini-batch $N$, the update step length $T$, and the size of horizons $H$.
**for** $k = 0 : E - 1$ **do**
    Randomly sampling $N$ collocation points. Allocate the number of points to each region $j$ with respect to their local probability $p_j$ as $N_j = p_j * N$.
    Training the neural network.
    **if** $T \mod (k-1) = 0$ **then**
        Record the model's parameters $\theta^k$.
        **for** repeats in $1, \ldots, 10$ **do**
            Calculate the loss $\mathcal{L}_j^k$ at grids' centers $X_j$.
            **for** $h = 0 : H - 1$ **do**
                Calculate the loss at grids' centers and train the neural network.
            **end for**
            Calculate the loss $\mathcal{L}_j^{k+h}$ at new coordinates $X_j + \delta X_j$
            Calculate the global sensitivity numerically and record (Eq. 6).
            Recover the model back to $\theta^k$.
        **end for**
        Calculate the averaged global sensitivity $\partial \mathcal{L}_i / \partial X_j$ from 10 records and build the sensitivity mastrix $\mathbf{S}$.
        Solve the optimization problem (Eq. 8) and get optimal fraction $\lambda_i$.
        Update the probabilities, $p_j$ from $\lambda_i$ (Eq. 11).
    **end if**
**end for**

## 4. PROBLEM DESCRIPTION & PROGRAM SETUP

### *4.1 Case study description: developing flow between two parallel plates*

To validate the effectiveness of our method, a two-dimensional incompressible developing flow between two parallel plates was used as a case study. For simplicity, steady-state conditions are assumed.

It is assumed that fluid enters between two parallel plates with a uniform velocity profile (Fig. 1). The plates are assumed to be infinitely large in the direction perpendicular to the x-y plane. Due to the no-slip boundary conditions, the fluid velocity is zero at the walls (plates). The flow is gradually decelerated in the x direction close to the walls, while the velocity of the fluid at the midsection has to gradually increase to satisfy mass conservation.

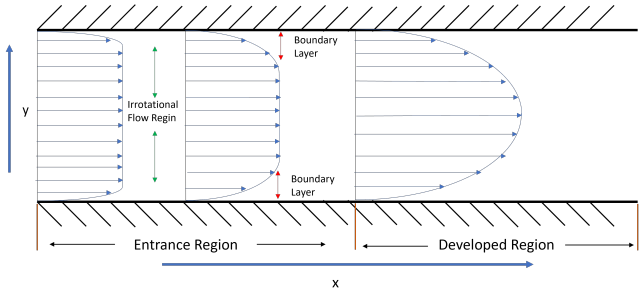


Fig. 1. The development of velocity between two parallel no-slip boundaries.)

In the flow developing region, the velocity exhibits large gradients within a boundary layer region but is almost flat within an irrotational flow region where gradients are small (Fig. 1). In the direction of velocity, the region before the cross-section where the boundary layers from the two walls merge at the center is called the entrance region. This entrance region is then followed by the developed region wherein the velocity profile remains approximately constant between the plates(Fig. 1) for subsequent axial locations.

Since the steady state is considered for simplicity, only two spatial coordinates, $x$ and $y$, are considered as inputs to the PINN and the outputs are the velocity in $x$ direction ($u$), velocity in $y$ direction ($v$), and pressure ($p$). This system is described by the two-dimensional incompressible continuity and momentum equations:

$$\begin{aligned}
\text{Continuity:} \quad & \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \\
\text{Momentum (x):} \quad & \rho\left(u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y}\right) + \frac{\partial p}{\partial x} \\
& - \mu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) = 0 \\
\text{Momentum (y):} \quad & \rho\left(u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y}\right) + \frac{\partial p}{\partial y} \\
& - \mu\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) = 0
\end{aligned} \tag{12}$$

$\rho$ is the fluid's density and $\mu$ is the dynamic viscosity.

### *4.2 Setup of PINN*

In this study, a fully connected neural network was constructed using the PyTorch framework. The hyperparameters were found by cross-validation. The resulting network is made of six hidden layers with 400 neurons each. For better convergence, the self-scalable tanh (Stan) activation function was used. To ensure the conservation of mass and reduce the number of terms in the loss function (Eq. 1), the velocities were calculated by automatic differentiation of the vector potential ($\psi$) (Eq. 13). This reduced the number of output variables to two, the potential $\psi$ and the pressure field $p$.

$$\begin{aligned}
u = \frac{\partial \psi}{\partial y} \quad v = -\frac{\partial \psi}{\partial x} \\
\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = \frac{\partial^2 \psi}{\partial x \partial y} - \frac{\partial^2 \psi}{\partial x \partial y} \equiv 0
\end{aligned} \tag{13}$$

To train the PINN, collocation points were assigned within a 2D domain situated in the narrow gap between two plates, where $x \in [0, 1]$ and $y \in [0, 0.015]$. We grouped these collocation points into 5 subsets according to the different terms occurring in the loss function as follows: **1.** Inlet boundary ($x = 0$; $y \in (0, 0.015)$); **2.** Upper wall boundary ($x \in [0, 1]$; $y = 0.015$); **3.** Lower wall boundary ($x \in [0, 1]$; $y = 0$); **4.** Outlet boundary ($x = 1$; $y \in (0, 0.015)$); **5.** PDE equations ($x \in (0, 1)$; $y \in (0, 0.015)$). Also, the differences between the solution from CFD (simulation) and PINN were used within an additional term in the loss functions. Since the terms in the loss function related to the PDE and boundary conditions have different magnitudes, a normalization procedure was carried out. For this purpose, all variables were formulated in non-dimensional form so as to ensure that all values are of the order of 1 ($O(1)$). We define the scale factors $x_s = 1$, $y_s = y_{max}/2 = 0.0075$, $u_s = u_{inlet} = 0.02$, $v_s = (u_s * y_s)/x_s = 0.00015$, $p_s = \rho * u_s^2 = 0.4$, and $p_{ref} = p_{atm} = 0$. The fluid density $\rho$ is 1000 and the dynamics viscosity is 0.0001.

During the training process, which spans 3,000 epochs, the Adam optimizer is employed for the initial 1,000 epochs and then, the optimization is transitioned to the L-BFGS optimizer as done in an earlier study (Markidis (2021)). Following the normalization of the variables as explained above, all loss functions' weights are set to 1 and remain unchanged throughout the training process, except for the loss of the CFD predictions. Since the CFD loss term serves only as a reference and it is desired to minimize its effect on the final model, its weight undergoes gradual annealing for the initial 1,000 epochs from 0.1, stabilizing at 0.035 for the subsequent 2,000 epochs during L-BFGS training.

## 5. RESULTS AND DISCUSSION

The probabilities are assigned for five different subsets corresponding to the inlet condition, wall condition, and PDE(x) loss respectively. To implement our sampling method, we uniformly divided the sampling regions for the inlet boundary, upper and lower wall boundaries, and PDE into 50, 34, and 40 ($x(5) \times y(8)$) segments or grids, respectively. In each training epoch, mini-batches of 600, 400, and 600 collocation points are sampled for inlet, wall, and PDE respectively according to the probabilities calculated for each of these sub-regions. The points' locations are allocated randomly within each sub-region.

For evaluating our method and comparing it to other sampling methods, the same mesh is used for 4 different cases: **1.** Uniform (the probabilities for each region are the same and constant); **2.** Local (probabilities are updated based on local sensitivity analysis); **3.** Global (probabilities are updated based on global sensitivity analysis); **4.** Residual-based (probabilities are updated based on the importance sampling (residual-based) proposed by Nabian et al. (2021)). All four cases start from the same probability distribution at the initial epoch of training and subsequently, probabilities are updated every 50 epochs for cases 2,3 and 4 above. For uniform sampling, the probabilities are not changed. The log of training errors (mean squared error, MSE) is recorded to evaluate the training process. For these 4 cases, the Mean Square Error (MSE) is calculated based on points drawn from points distributed uniformly within the corresponding mesh element, regarded as validation. This avoids biased calculations towards particular locations within a mesh element.

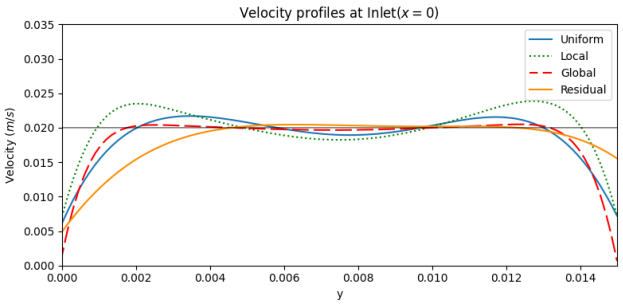


Fig. 2. The prediction of the velocity profiles at the inlet ($x = 0$).

### 5.1 Mean squared error

The global approach demonstrated superior training convergence and predictive performance compared to the other three methods since it results in the lowest MSE in both the training and validation datasets (Table 1). As expected, the global sensitivity approach requires longer GPU time due to the sensitivity calculation.

Table 1. Mean squared errors at convergence

| Dataset | Uniform | Local | Global | Residual |
|---|---|---|---|---|
| Training | 0.066 | 0.090 | 0.049 | 0.063 |
| Validation | 0.064 | 0.068 | 0.061 | 0.142 |
| GPU time (s) | 10380.87 | 11198.72 | 17226.59 | 7910.32 |

### 5.2 Prediction performance

The prediction of the specified inlet condition where the sharp gradients are expected, i.e. at the inlet next to the walls, is of particular interest since errors are expected to be larger at this location. Moreover, the accuracy of the inlet condition will be crucial to ensure that the correct prescribed amount of flow rate is being fed into the system. From the numerical point of view, it is particularly difficult to reconcile the no-slip condition at the inlet with the satisfaction of the momentum equations at locations immediately adjacent to the plates (see Eq. 1). Moreover, the inlet section will be very important for predicting the highest shear stresses that will limit the design.

As shown in Figure 2, the prediction of velocity ($u$) by the PINN trained with the global sensitivity approach (red dashed) approximated the specified inlet condition with the best accuracy as compared to the other sampling methods. Among all methods, the velocity predicted at the wall ($y = 0$ or 0.015) by the model based on the global sensitivity was the closest to zero thus approaching the actual no-slip boundary condition. Moreover, the model based on the global sensitivity-based sampling generated the closest profile to a uniform profile within the inlet's irrotational flow region as compared to the other sampling methods. As anticipated, distinct gradients emerge within

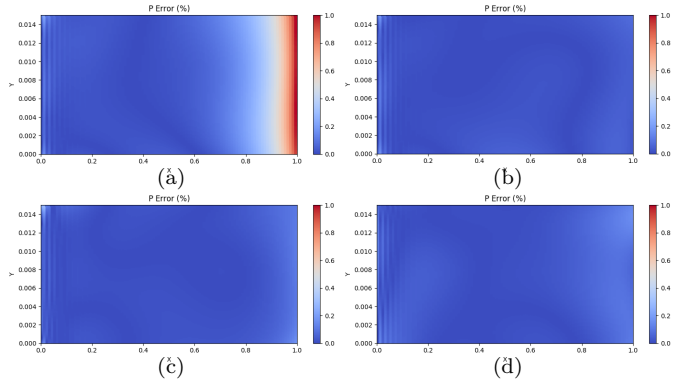


Fig. 3. The relative error of pressure ($p$). Red color means the error is large. (a) Uniform; (b) Local; (c) Global; (d) Residual-based.

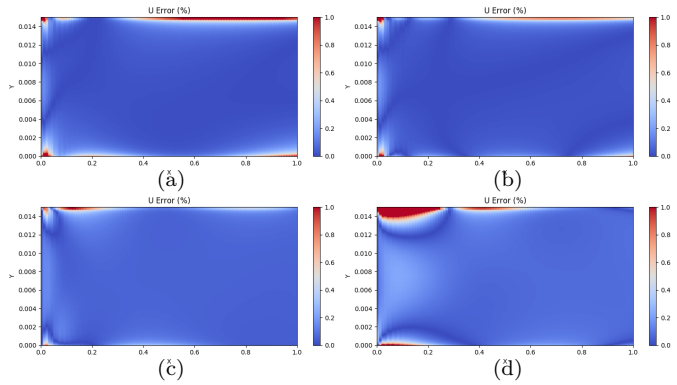


Fig. 4. The relative error of velocity in x direction ($u$). Red color means the error is large. (a) Uniform; (b) Local; (c) Global; (d) Residual-based.

both boundary layers. Compared with the other three cases, the global sensitivity-based sampling was able to resolve the trade-offs between the different terms of the loss functions (Eq. 1).

The results by CFD were used as a reference to calculate the relative error. It should be noticed that a key advantage of PINN is its significantly faster execution time (milliseconds) than CFD (20 mins for this case). The predicted pressure $p$ by the PINN trained with uniform random sampling exhibited a larger error at the region close to the outlet (Fig. 3a), represented by the red color regions. In comparison, the models trained with sampling points chosen by the other 3 approaches, local sensitivity, importance sampling, and global sensitivity, achieved a relatively good prediction of pressure as shown by the lack of red color regions indicating smaller errors (Fig. 3).

For this particular case study, we focused on the accuracy of the prediction of velocity in the $x$ direction (flow direction) since the velocities perpendicular to the flow are close to zero. Specifically, the two regions of interest are the entrance region and the boundary layers, where significant velocity gradients occur. For these two regions, the errors resulting from the models trained with points chosen with either the local or global sensitivity approaches were the smallest as shown by the reduced areas of red color regions for these two methods as compared to the other sampling methods (Fig. 4b,c).

## 6. CONCLUSIONS

We proposed a novel sampling method for locating collocation points in the training of PINNs based on sensitivity analysis. For the chosen case study, developing flow between two parallel plates, our proposed sensitivity-based sampling by global analysis (SBS-G) showed significant improvement in both training convergence and overall prediction performance as compared to other sampling methods. This method enabled more accurate predictions at the inlet section between the plates, characterized by sharp gradients, as well as more accurate predictions of pressure and velocity profiles. Furthermore, the proposed global sensitivity analysis served to effectively distribute the sampling density across several sections of the geometric domain, e.g. boundaries versus interior points. Consequently, our global approach to sensitivity-based sampling successfully resolves conflicts between various training objectives by optimizing the sampling probability distribution for different objective functions related to boundary conditions and partial differential equations. The superior performance of the proposed approach is particularly evident for model predictions at the inlet section between the plates, where it simultaneously satisfies conditions for a uniform irrotational core flow region, symmetric boundary layers with velocity gradients, and no-slip wall conditions. Further studies are conducted to investigate the effectiveness of sensitivity-based sampling for the training of PINNs in more complex geometries such as bioreactors.

## ACKNOWLEDGEMENTS

We sincerely appreciate Prof. Jeff Gostick, for his generous support of computational resources for PINN training.

## REFERENCES

Fang, Q., Mou, X., and Li, S. (2023). A physics-informed neural network based on mixed data sampling for solving modified diffusion equations. *Sci Rep*, 13, 2491.

Mao, Z. and Meng, X. (2023). Physics-informed neural networks with residual/gradient-based adaptive sampling methods for solving pdes with sharp solutions. *arXiv preprint*, arXiv:2302.08035.

Markidis, S. (2021). The old and the new: Can physics-informed deep-learning replace traditional linear solvers? *Front Big Data*, 41, 669097.

Nabian, M., Gladstone, R., and Meidani, H. (2021). Efficient training of physics-informed neural networks via importance sampling. *Computer aided Civil Eng*, 36, 962–977.

Raissi, M., Perdikaris, P., and Karniadakis, G. (2019). 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.

Wu, C., Zhu, M., Tan, Q., Kartha, Y., and Lu, L. (2023). A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks. *Comput Methods Appl Mech Eng*, 403, 115671.