Fixed-Budget Online Adaptive Learning for Physics-Informed Neural Networks. Towards Parameterized Problem Inference

Thi Nguyen Khoa Nguyen[1,2,3][0000-0002-0834-7186], Thibault Dairay[2], Raphaël Meunier[2], Christophe Millet[1,3], and Mathilde Mougeot[1,4][0009-0009-6346-4519]

[1] Universite Paris-Saclay, ENS Paris-Saclay, CNRS, Centre Borelli, 91190 France
[2] Michelin, Centre de Recherche de Ladoux, Cébazat, 63118 France
[3] CEA, DAM, DIF, F-91297 Arpajon, France
[4] ENSIIE, Évry-Courcouronnes, 91000 France

Abstract. Physics-Informed Neural Networks (PINNs) have gained much attention in various fields of engineering thanks to their capability of incorporating physical laws into the models. The partial differential equations (PDEs) residuals are minimized on a set of collocation points which distribution appears to have a huge impact on the performance of PINNs and the assessment of the sampling methods for these points is still an active topic. In this paper, we propose a Fixed-Budget Online Adaptive Learning (FBOAL) method, which decomposes the domain into sub-domains, for training collocation points based on local maxima and local minima of the PDEs residuals. The numerical results obtained with FBOAL demonstrate important gains in terms of the accuracy and computational cost of PINNs with FBOAL for non-parameterized and parameterized problems. We also apply FBOAL in a complex industrial application involving coupling between mechanical and thermal fields.

Keywords: Physics-informed neural networks · Adaptive learning · Rubber calendering process

1 Introduction

In the last few years, Physics-Informed Neural Networks (PINNs) [8] have become an attractive and remarkable scheme of solving inverse and ill-posed partial differential equations (PDEs) problems. The applicability of PINNs has been demonstrated in various fields of research and industrial applications [3]. However, PINNs suffer from significant limitations. The training of PINNs takes high computational cost, that is, a standard PINN must be retrained for each PDE problem, which is expensive and the numerical physics-based methods can strongly outperform PINNs for forward modeling tasks. There are continuing efforts to overcome this limitation by proposing to combine with reduced order methods so that the model has a strong generalization capacity [2]. Furthermore, the theoretical convergence properties of PINNs are still poorly understood and need further investigations [9]. As PINNs integrate the PDEs constraints by minimizing

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36027-5_36

the PDE residuals on a set of collocation points during the training process, it has been shown that the location of these collocation points has a great impact on the performance of PINNs [1]. To the best of the authors' knowledge, the first work that showed the improvement of PINNs performance by modifying the set of collocation points is introduced by Lu et al. (2021) [5]. This work proposed the Residual-based Adaptive Refinement (RAR) that adds new training collocation points to the location where the PDE residual errors are large. RAR has been proven to be very efficient to increase the accuracy of the prediction but however leads to an uncontrollable amount of collocation points and computational cost at the end of the training process. In this work, we propose a Fixed-Budget Online Adaptive Learning (FBOAL) that fixes the number of collocation points during the training. The method adds and removes the collocation points based on the PDEs residuals on sub-domains during the training. By dividing the domain into smaller sub-domains it is expected that local maxima and minima of the PDEs residuals will be quickly captured by the method. Furthermore, the stopping criterion is chosen based on a set of reference solutions, which leads to an adaptive number of iterations for each specific problem and thus avoids unnecessary training iterations. The numerical results demonstrate that the use of FBOAL help to reduce remarkably the computational cost and gain significant accuracy compared to the conventional method of non-adaptive training points. In the very last months, several works have also introduced a similar idea of adaptive re-sampling of the PDE residual points during the training [1,6,10]. Wu et al. (2023) [10] gave an excellent general review of these methods and proposed two adaptive re-sampling methods named Residual-based Adaptive Distribution (RAD) and Residual-based adaptive refinement with distribution (RAR-D), which are the generalization of all existing methods of adaptive re-sampling for the collocation points. These approaches aim to minimize the PDEs residuals at their global maxima on the entire domain. Besides that, the existing studies did not investigate the parameterized PDE problems (where the parameter of interest is varied). In this study, we first compare the performance of RAD, RAR-D, and FBOAL in an academic test case (Burgers equation). We illustrate a novel utilization of these adaptive sampling methods in the context of parameterized problems. The following of this paper is organized as follows. In section 2, we briefly review the framework of PINNs and introduce the adaptive learning strategy (FBOAL) for the collocation points. We then provide the numerical results of the performance of the studied methods and comparison to the classical PINNs and other adaptive re-sampling methods such as RAD and RAR-D in a test case of Burgers equation. The application to an industrial use case is also represented in this section. Finally, we summarize the conclusions in section 4.

2 Methodology

In this section, the framework of Physics-Informed Neural Networks (PINNs) [8] is briefly presented. Later, Fixed-Budget Online Adaptive Learning (FBOAL) for PDE residual points is introduced.

2.1 Physics-informed neural networks

To illustrate the methodology of PINNs, let us consider the following parameterized PDE defined on the domain $\Omega \subset \mathbb{R}^d$: $\boldsymbol{u}_t + \mathcal{N}_{\mathbf{x}}(\boldsymbol{u}, \boldsymbol{\lambda}) = 0$ for $\mathbf{x} \in \Omega, t \in [0, T]$ where $\boldsymbol{\lambda} \in \mathbb{R}^p$ is the PDE parameter vector with the boundary condition $\mathcal{B}(\boldsymbol{u}, \mathbf{x}, t) = 0$ for $\mathbf{x} \in \partial\Omega$ and the initial condition $\boldsymbol{u}(\mathbf{x}, 0) = g(\mathbf{x})$ for $\mathbf{x} \in \Omega$. In the conventional framework of PINNs, the solution $\boldsymbol{u}$ of the PDE is approximated by a fully-connected feed-forward neural network $\mathcal{NN}$ and the prediction for the solution can be represented as $\hat{\boldsymbol{u}} = \mathcal{NN}(\mathbf{x}, t, \boldsymbol{\theta})$ where $\boldsymbol{\theta}$ denotes the trainable parameters of the neural network. The parameters of the neural network are trained by minimizing the cost function $L = L_{pde} + w_{ic}L_{ic} + w_{bc}L_{bc}$, where the terms L_{pde}, L_{ic}, L_{bc} penalize the loss in the residual of the PDE, the initial condition, the boundary condition, respectively, and w_{ic}, w_{bc} are the positive weight coefficients to adjust the contribution of each term to the cost function: $L_{pde} = \dfrac{1}{N_{pde}} \sum_{i=1}^{N_{pde}} |\hat{\boldsymbol{u}}_{t_i} + \mathcal{N}_{\mathbf{x_i}}(\hat{\boldsymbol{u}}, \boldsymbol{\lambda})|^2$, $L_{ic} = \dfrac{w_{ic}}{N_{ic}} \sum_{i=1}^{N_{ic}} |\hat{\boldsymbol{u}}(\mathbf{x}_i, 0) - g(\mathbf{x}_i)|^2$, and $L_{bc} = \dfrac{w_{bc}}{N_{bc}} \sum_{i=1}^{N_{bc}} |\mathcal{B}(\hat{\boldsymbol{u}}, \mathbf{x_i}, t_i)|^2$ where N_{ic}, N_{bc}, N_{data} denote the numbers of learning points for the initial condition, boundary condition, and measurements (if available), respectively, and N_{pde} denotes the number of residual points (or collocation points or unsupervised points) of the PDE.

We note that PINNs may provide different performances with different network initialization. In this work when comparing the results of different configurations of PINNs, we train PINNs five times and choose the mean and one standard deviation values of the performance criteria of five models for visualization and numerical comparison.

2.2 Adaptive learning strategy for PDEs residuals

Fixed-Budget Online Adaptive Learning Motivated by the work of Lu et al. (2021) [5] which proposed the Residual-based Adaptive Refinement (RAR) that adds progressively during the training more collocation points at the locations that yield the largest PDE residuals, our primary idea is to control the number of potentially added training points by removing the collocation points that yield the smallest PDE residuals so that the number of collocation points remains the same during the training. With this approach, the added points tend to be placed at nearly the same location corresponding to the global maximum value of the PDE residual. We suggest considering a set of sub-domains in order to capture not only the global extrema but also the local extrema of the PDEs residuals. More precisely, we propose a Fixed-Budget Online Adaptive Learning (FBOAL) that adds and removes collocation points that yield the largest and the smallest PDE residuals on the sub-domains during the training (see Algorithm 1). With the domain decomposition step, the algorithm is capable of detecting the local extrema inside the domain. Another remarkable advancement of this method is that in the parameterized problem, the collocation points can be relocated to the values of the parameter for which the solution is more complex (see section 3.1 for an illustration on Burgers equation). To minimize the cost

Algorithm 1 Fixed-Budget Online Adaptive Learning (FBOAL)

```
Require: The number of sub-domains $d$, the number of added and removed points $m$,
    the period of resampling $k$, a testing data set, a threshold $s$.
 1: Generate the set $\mathcal{C}$ of collocation points on the studied domain $\Omega$.
 2: Divide the domain into $d$ sub-domains $\Omega_1 \cup \Omega_2 ... \cup \Omega_d = \Omega$.
 3: for $lr_i$ in $lr$ do
 4:   repeat
 5:     Train PINNs for $k$ iterations with the learning rate $lr_i$.
 6:     Generate a new set $\mathcal{C}'$ of random points inside the domain $\Omega$.
 7:     Compute the PDE residual at all points in the set $\mathcal{C}'$ and the set $\mathcal{C}$.
 8:     On each subdomain $\Omega_i$, take 1 point of the set $\mathcal{C}'$ which yield the largest PDE
        residuals on the subdomain. Gather these points into a set $\mathcal{A}$.
 9:     On each subdomain $\Omega_i$, take 1 point of the set $\mathcal{C}$ which yield the smallest PDE
        residuals on the subdomain. Gather these points into a set $\mathcal{R}$.
10:     Add $m$ points of the set $\mathcal{A}$ which yield the largest errors for the residuals to
        the set $\mathcal{C}$, and remove $m$ points of the set $\mathcal{R}$ which yield the smallest errors
        for the residuals.
11:   until The maximum number of iterations $K$ is reached or the error of the
      prediction on the testing data set of reference is smaller than some threshold $s$.
12: end for
```

function, we adopt Adam optimizer with a learning rate decay strategy, which is proven to be very efficient in training deep learning models. In this work, we choose a set of learning rate values $lr = \{10^{-4}, 10^{-5}, 10^{-6}\}$. The way to divide the domain and the number of sub-domains can play important roles in the algorithm. If we dispose of expert knowledge on the PDEs problem, we can divide the domain as a finite-element mesh such that it is very fine at high-gradient locations and coarse elsewhere. In this primary work, we dispose of no knowledge *a priori* and use square partitioning for the domain decomposition. We note this partitioning is not optimal when dealing with multidimensional space and different scales of spatial and/or temporal dimensions. The stopping criterion is chosen based on a number of maximum iterations for each value of the learning rate and an error criterion computed on a testing data set of reference. The detail of this stopping criterion is specified in each use case. With this stopping criterion, the number of training iterations is also an adaptive number in each specific case, which helps to avoid unnecessary training iterations. We note that when dealing with multi-physics problems which involve systems of PDEs, we separate the set of collocation points into different subsets for each equation and then effectuate the process independently for each subset. Separating the set of collocation points helps to avoid the case when added points for one equation are removed for another. The code in this study is available from the GitHub repository `https://github.com/nguyenkhoa0209/PINNs_FBOAL`.

Residual-based Adaptive Distribution and Residual-based Adaptive Refinement with Distribution Wu et al. (2023) [10] proposed two residual-based adaptive sampling approaches named Residual-based Adaptive Distribution (RAD) and Residual-based Adaptive Refinement with Distribution (RAR-D).

In these approaches, the training points are randomly sampled according to a probability density function which is based on the PDE residuals. In RAD, all the training collocation points are re-sampled. While in RAR-D, only a few new points are sampled and then added to the training data set. The main differences between RAD, RAR-D, and our proposed method FBOAL lie in the domain decomposition step in FBOAL and the percentage of modified collocation points after every time we effectuate the re-sampling step. The decomposition step helps FBOAL to be able to detect local maxima and local minima of the residuals inside the domain (which, however, depends on the way we divide the domain), and thus FBOAL gives equal concentration for all local maxima of the PDE residuals.

3 Numerical results

In this section, we first compare and demonstrate the use of adaptive sampling methods (FBOAL, RAD, and RAR-D) to solve the viscous Burgers equation in both non-parameterized context and parameterized context (i.e. the viscosity is fixed or not). We then illustrate the performance of FBOAL in a realistic industrial case: a system of PDEs that is used in the rubber calendering process. In the following, unless specifying otherwise, to compare the numerical performance of each methodology, we use the relative $\mathcal{L}^2$ error defined as $\epsilon_w = \dfrac{||w - \hat{w}||_2}{||w||_2}$ where w denotes the reference simulated field of interest and $\hat{w}$ is the corresponding PINNs prediction.

3.1 Burgers equation

We consider the following Burgers equation:

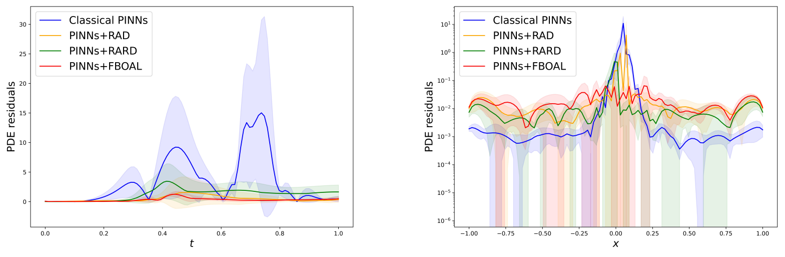
$$\begin{cases} u_t + uu_x - \nu u_{xx} = 0 \quad \text{for } x \in [-1,1], t \in [0,1] \\ u(x,0) = -\sin(\pi x) \\ u(-1,t) = u(1,t) = 0 \end{cases}$$

where ν is the viscosity. For a small value of ν, the solution is very steep close to $x = 0$. For higher values, the solution becomes smoother. Thus when ν is fixed, it is expected that the collocation points are located close to $x = 0$ during the training to better capture the discontinuity. When ν is varied, it is expected that the number of collocation points is more important for smaller values of ν while being located close to $x = 0$. In the following, we assess whether FBOAL can relocate the collocation points to improve the performance of PINNs.

In our experiment, to simplify the cost function and guarantee the boundary and initial conditions, these conditions are forced to be automatically satisfied by using the following representation for the prediction $\hat{\boldsymbol{u}} = t(x-1)(x+1)\mathcal{NN}(.) - \sin(\pi x)$. Interested readers may refer to the work in [4] for the general formulations. With this strategy, we do not need to adjust different terms in the loss function as there is only the loss for PDE residuals which is left. For the architecture of PINNs, we use a feedforward network with 4 hidden layers with 50 neurons per layer with *tanh* activation function. The results are obtained with 50,000 epochs with the learning rate $lr = 10^{-3}$, 200,000 epochs with the learning rate $lr = 10^{-4}$ and 200,000 epochs with $lr = 10^{-5}$. For a fair comparison, the initialization of the training collocation points is the same for all methodologies. The learning data set and the testing data set are independent in all cases.

Non-parameterized problem We first illustrate the performance of adaptive sampling approaches in a context where ν is fixed. We take 10 equidistant values of $\nu \in [0.0025, 0.0124]$. For each ν, we compare the performance of classical PINNs, PINNs with RAD, RAR-D, and PINNs with FBOAL. For the training of PINNs, we take $N_{pde} = 1024$ collocation points that are initialized equidistantly inside the domain. We take a testing set of reference solutions on a 10×10 equidistant spatio-temporal mesh and stop the training when either the number of iterations surpasses $K = 500,000$ or the relative $\mathcal{L}^2$ error between PINNs prediction and the testing reference solution is smaller than the threshold $s = 0.02$. The following protocol allows us to compare fairly all adaptive sampling methods. For the training of FBOAL, we divide the domain into $d = 200$ sub-domains as squares of size 0.1. After every $k = 2,000$ iterations, we add and remove $m = 2\% \times N_{pde} \approx 20$ collocation points based on the PDE residuals. For the training of RAD, we take $k = 1$ and $c = 1$ and effectuate the process after $2,000$ iterations. For the training of RAR-D, we take $k = 2$ and $c = 0$ and after every $2,000$ iterations, 5 new points are added to the set of training collocation points. At the end of the training, the number of collocation points for FBOAL and RAD remains the same as at the beginning ($N_{pde} = 1024$), while for RAR-D, this number increases gradually until the stopping criterion is satisfied.

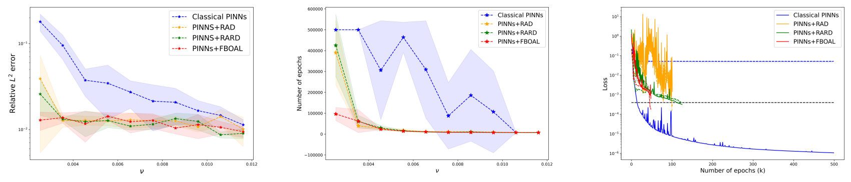
Figure 1 shows the PDE residuals for $\nu = 0.0025$ after the training process on the line $x = 0$ and at the instant $t = 1$ where the solution is very steep. The curves and shaded regions represent the geometric mean and one standard deviation of five runs. On the line $x = 0$ (Figure 1a), with classical PINNs where the collocation points are fixed during the training, the PDE residuals are very high and obtain different peaks (local maxima) at different instants. It should be underlined that all the considered adaptive sampling methods are able to decrease the values of local maxima of the PDEs residuals. Among the approaches where the number of collocation points is fixed, FBOAL is able to obtain the smallest values for the PDE residuals, and for its local maxima, among all the adaptive resampling methodologies. At the instant $t = 1$ (Figure 1b), the same conclusion can be drawn. However, we note that at this instant, the classical PINNs outperform other methods at the zone where there are no discontinuities (the zones that are not around $x = 0$). This is because the collocation points are fixed during the training process for classical PINNs, while with other methods, the collocation points are either re-assigned (with FBOAL and RAD) or some points are added (with RAR-D) to the high gradient regions. Thus, for adaptive resampling methods, the training networks have to additionally balance the errors for high-gradient locations and low-gradient locations and thus diminish the accuracy at the zones where there are low gradients as the cost function is a sum of the PDE errors at all points.



(a) On the line $x = 0$ **(b)** At instant $t = 1$

Fig. 1: Burgers equation: Absolute value of PDE residuals for $\nu = 0.0025$.

To assess the overall performance of PINNs, we evaluate the errors of the prediction on a 256×100 spatio-temporal mesh (validation mesh). Figure 2a shows the relative $\mathcal{L}^2$ error between PINNs predictions and reference solution. Figure 2b shows the number of training iterations for PINNs to meet the stopping criterion. As expected, when ν increases, which means the solution becomes smoother, the accuracy of PINNs in all methodologies increases and the models need less number of iterations to meet the stopping criterion. We note that when ν is large and the solution is very smooth, the classical PINNs are able to give comparable performance to PINNs with adaptative methodologies. However, when ν becomes smaller, it is clear that PINNs with adaptive sampling methods outperform classical PINNs in terms of accuracy and robustness. Among these strategies, FBOAL provides the best accuracy in terms of errors and also needs the least iterations to stop the algorithm. Table 1 illustrates the training time of each methodology for $\nu = 0.0025$. We observe that by using FBOAL a huge amount of training time is gained compared to other approaches. Figure 2c illustrates the cost function during the training process for $\nu = 0.0025$ and the errors of the prediction on the testing mesh. For clarity, only the best cost function (which yields the smallest values after the training process in five runs) of each methodology is plotted. After every $k = 2,000$ epochs, as the adaptive sampling methods relocate the collocation points, there are jumps in the cost function. The classical PINNs minimize the cost function better than other methodologies because the position of collocation points is fixed during the training. This leads to the over-fitting of classical PINNs on the training collocation points and does not help to increase the accuracy of the prediction on the testing mesh. While with adaptive sampling methods, the algorithm achieves better performance on the generalization to different meshes (the testing and validation meshes). Table



(a) Relative $\mathcal{L}^2$ error **(b)** Nb. of training iterations **(c)** Loss for $\nu = 0.0025$

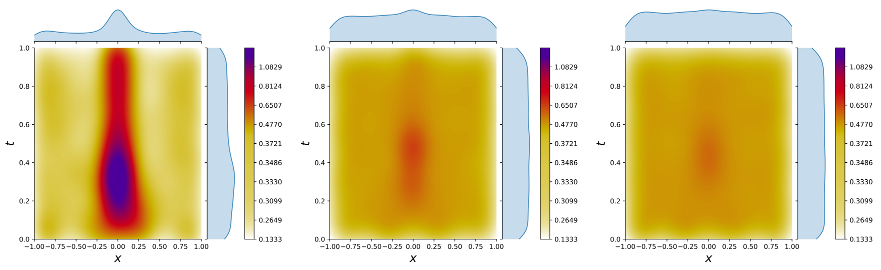
Fig. 2: Burgers equation: Comparison of classical PINNs and PINNs with adaptive sampling approaches. In (c) the solid lines show the cost function during the training, the dashed lines show the errors on the testing mesh, and the black line shows the threshold to stop the training.

1 provides the training time and the number of resampling of each methodology with the hyperparameter values mentioned previously, which are optimal for all adaptive resampling methods. We see that the resampling does affect the training time. More precisely, with RAR-D and RAD, a huge number of resampling is effectuated, which leads to a long training time compared to the classical PINNs (even though these methods need smaller numbers of training iterations). While with FBOAL, the number of resampling is small and we obtain a smaller training time compared to the classical PINNs.

Table 1: Burgers equation: Training time and the number of resampling for $\nu = 0.0025$. The training is effectuated on an NVIDIA V100 GPU card.

	Classical	RAR-D	RAD	FBOAL
Training time (minutes)	33.7 ± 1.5	41.0 ± 5.8	38.8 ± 2.3	$\mathbf{21.5 \pm 2.7}$
Number of resampling	0 ± 0	201 ± 75	210 ± 77	$\mathbf{48 \pm 2}$

In the following, we analyze in detail the performance of FBOAL. Figure 3 illustrates the density of collocation points after the training with FBOAL for different values of ν. We see that, for the smallest value $\nu = 0.0025$, FBOAL relocates the collocation points close to x=0 where the solution is highly steep. For $\nu = 0.0076$, as there is only one iteration of FBOAL that adds and removes points (see Figure 2b), there is not much difference with the initial collocation points but we still see few points are added to the center of the domain, where the solution becomes harder to learn. For the biggest value $\nu = 0.0116$, there is no difference with the initial collocation points as PINNs already satisfy the stopping criterion after a few iterations. For the values of the hyperparameters, empirical tests (not shown here for concision) suggest starting with small values of m (number of added and removed points) and k (period of resampling) (for example $k = 1,000, m = 0.5\% N_{pde}$) and then increase these values to see whether the predictions can become more accurate or not.

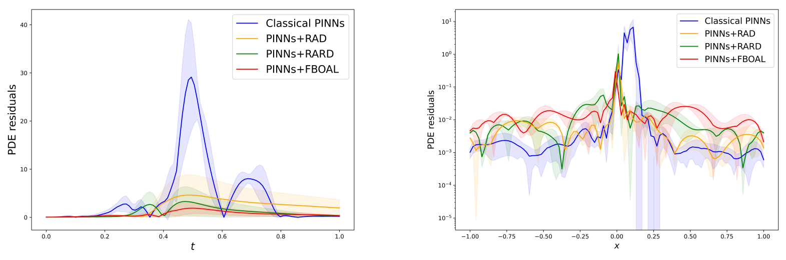


(a) For $\nu = 0.0025$ **(b)** For $\nu = 0.0076$ **(c)** For $\nu = 0.0116$

Fig. 3: Burgers equation: Density of collocation points after FBOAL training.

Parameterized problem We now illustrate the performance of PINNs in a parameterized problem where ν can be varied. In this case, ν is also considered as an input of PINNs, i.e. the network in PINNs is represented as $\mathcal{NN}(\mathbf{x}, t, \nu, \boldsymbol{\theta})$. With this configuration, PINNs can predict the solution for different values of ν in one model. For the training, we take 40 values of $\nu \in [0.0025, 0.0124]$. For each ν, we initialize 1024 equidistant collocation points, which leads to $N_{pde} = 1024 \times 40 = 40,960$ collocation points in total. During the training with FBOAL, the number of total collocation points remains the same, however, the number of collocation points can be varied for each ν. We take a testing set of reference solutions on a 10×10 equidistant spatio-temporal mesh and stop the training when either the number of iterations surpasses $K = 2 \times 10^6$ or the sum of relative $\mathcal{L}^2$ error between PINNs prediction and the testing reference solution of all learning values of ν is smaller than the threshold $s = 0.02 \times 40 = 0.8$. For the training of FBOAL, we divide the domain into $d = 200$ sub-domains as squares of size 0.1. After every $k = 2,000$ iterations, we add and remove $m = 0.5\% \times N_{pde} \approx 200$ collocation points based on the PDE residuals. For the training of RAD, we take $k = 1$ and $c = 1$ and effectuate the process after $2,000$ iterations. For the training of RAR-D, we take $k = 2$ and $c = 0$ and after every $2,000$ iterations, $5 \times 40 = 200$ new points are added to the set of training collocation points. At the end of the training, the number of collocation points for FBOAL and RAD remains the same as the beginning ($N_{pde} = 40,960$), while for RAR-D, this number increases gradually until the stopping criterion is satisfied.

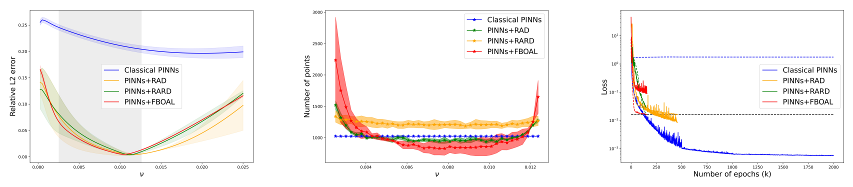
Figure 4 illustrates the absolute value of the PDE residuals for $\nu = 0.0025$ on the line $x = 0$ and at the instant $t = 1$ where the solution is very steep. On the line $x = 0$ (Figure 4a), with classical PINNs, the PDE residuals are very high and obtain different local maxima at different instants. Again, all the considered adaptive sampling methods are able to decrease the values of local maxima of the PDEs residuals. Among these approaches, FBOAL is able to obtain the smallest values for the PDE residuals. We note that the number of collocation points of each methodology is different (see Figure 5b). For $\nu = 0.0025$, FBOAL relocates a very high number of collocation points. This leads to a better performance of FBOAL for $\nu = 0.0025$. At the instant $t = 1$, FBOAL, RAD and RAR-D provide nearly the same performance and are able to decrease the values of local maxima of the PDEs residuals compared to the classical PINNs. Figure 5a shows the relative $\mathcal{L}^2$ error between PINNs prediction and reference



(a) On the line $x = 0$ **(b)** At instant $t = 1$

Fig. 4: Burgers equation: Absolute value of PDE residuals for $\nu = 0.0025$.

solution on a 256×100 mesh. The zone in gray represents the learning interval for ν (interpolation zone). Figure 5b shows the number of collocation points for each ν. We see again in the interpolation zone, that when ν increases, the accuracy of PINNs in all methodologies increases. It is clear that PINNs with adaptive sampling methods outperform classical PINNs in terms of accuracy for all values of ν in both interpolation and extrapolation zones. As expected, with the approaches FBOAL and RAD (where N_{pde} is fixed), the number of collocation points for smaller values of ν is more important than the number for higher ones. This demonstrates the capability of FBOAL and RAD of removing unnecessary collocation points for higher values of ν (whose solution is easier to learn) and adding them for smaller values of ν (as the solution becomes harder to learn). When ν tends to the higher extreme, the number of training points increases as there are fewer training values for ν in this interval. While with RAR-D (where N_{pde} increases gradually), we do not see the variation of the number of collocation points between different values of ν as RAR-D tries to add more points at the discontinuity for all ν. Table 2 provides the training time of and the number of resampling of each methodology. It is clear that RAR-D outperforms the other approaches in terms of computational time as this approach needs very few numbers of resampling to meet the stopping criterion. However, to achieve this gain, RAR-D added for about 40,000 new collocation points after the training, which is not profitable in terms of memory. RAD and FBOAL provide comparable computational training time in this case and they both outperform the classical PINNs while the number of collocation points in total is fixed. Figure 5c illustrates the cost function during the training process and the errors of the prediction on the testing mesh. For clarity, only the best cost function (which yields the smallest number of iterations after the training



(a) Relative $\mathcal{L}^2$ error **(b)** Nb. of collocation points **(c)** Loss function

Fig. 5: Burgers equation: comparison of classical PINNs and PINNs with adaptive sampling approaches. The zone in gray is the learning interval. In (c) the solid lines show the cost function during the training, the dashed lines show the errors on the testing data set, and the black line shows the threshold to stop the training.

process in five runs) of each methodology is plotted. Again, since the position of the collocation points is fixed during the training, classical PINNs minimize the cost function better than PINNs with adaptive sampling methods, which leads to over-fitting on the training data set and does not help to improve the accuracy of the prediction on the testing mesh. We see that FBOAL and RAR-D need a much smaller number of iterations to meet the stopping criterion.

In the following, we analyze in detail the performance of FBOAL. Figure 6 illustrates the density of collocation points after the training with FBOAL with different values of ν. We see that for all values of ν, FBOAL relocates the collocation points close to the discontinuity. For the values of the hyperparameters, empirical tests (not shown here for concision) suggest starting with small values of m (number of added and removed points) and k (period of resampling) (for example $k = 1,000, m = 0.5\% N_{pde}$) and then increase these values to see whether the predictions can become more accurate or not.

Table 2: Burgers equation: Training time and the number of resampling of each methodology. The training is effectuated on an NVIDIA A100 GPU card.

	Classical	RAR-D	RAD	FBOAL
Training time (hours)	13.2 ± 0.0	**1.4 ± 0.3**	9.1 ± 3.5	7.8 ± 1.9
Number of resampling	0 ± 0	**34 ± 8**	204 ± 71	173 ± 25

3.2 Application to calendering process

In the industry of tire manufacturing, calendering is a mechanical process used to create and assemble thin tissues of rubber. From the physical point of view, the rubber is assimilated as an incompressible non-Newtonian fluid flow in the present study (in particular, the elastic part of the material is not considered here). Moreover, only the steady-state regime is considered. The goal of the present study is only to model the 2D temperature, velocity, and pressure fields

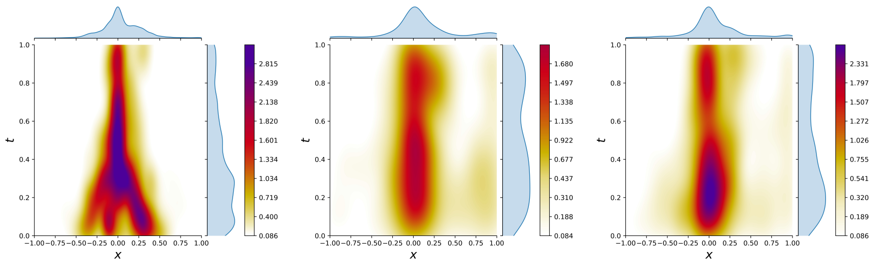

(a) For $\nu = 0.0025$ **(b)** For $\nu = 0.0076$ **(c)** For $\nu = 0.0116$

Fig. 6: Burgers equation: Density of collocation points after FBOAL training.

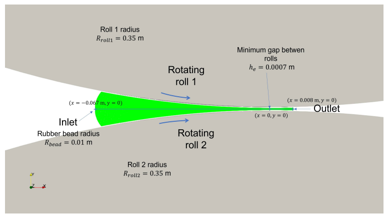
inside rubber materials going through two contra-rotating rolls of the calender as depicted in Figure 7a. A detailed description of the calendering process and its mathematical formulation can be found in [7]. Here we briefly introduce the dimensionless PDEs system that is used in PINNs training process:

$$\begin{aligned}
&\tilde{\nabla}.\Big(2\tilde{\eta}(\tilde{\vec{\boldsymbol{u}}},\tilde{T})\tilde{\bar{\bar{\epsilon}}}(\tilde{\vec{\boldsymbol{u}}})\Big) - \tilde{\vec{\nabla}}\tilde{p} = \vec{0}\\
&\tilde{\nabla}.\tilde{\vec{\boldsymbol{u}}} = 0\\
&\tilde{\vec{\boldsymbol{u}}}\tilde{\vec{\nabla}}\tilde{T} = \frac{1}{Pe}\tilde{\nabla}^2\tilde{T} + \frac{Br}{Pe}\tilde{\eta}(\tilde{\vec{\boldsymbol{u}}},\tilde{T})|\tilde{\gamma}(\tilde{\vec{\boldsymbol{u}}})|^2
\end{aligned}$$

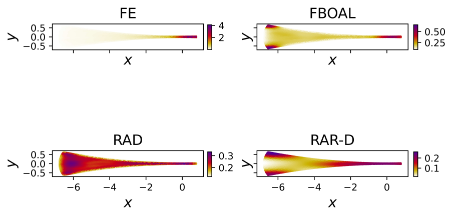
where $\tilde{\vec{\boldsymbol{u}}} = (\tilde{u}_x, \tilde{u}_y)^T$ is the velocity vector, $\tilde{p}$ is the pressure, $\tilde{T}$ is the temperature, $\tilde{\bar{\bar{\epsilon}}}$ is the strain-rate tensor, $\tilde{\eta}$ is the dynamic viscosity. Pe and Br are the dimensionless Péclet number and Brickman number, respectively.

We tackle an ill-posed configuration problem, i.e the boundary conditions of the problem are not completely defined, and we dispose of some measurements of the temperature ($N_T = 500$ measurements, which are randomly distributed inside the domain). The goal is to infer the pressure, velocity, and temperature fields at all points in the domain. We note that here, no information on the pressure field is given, only its gradient in the PDE residual is. Thus the pressure is only identifiable up to a constant. As shown in [7], only the information of the temperature is not sufficient to guarantee a unique solution for the velocity and the pressure fields, we take in addition the knowledge of velocity boundary conditions. The authors also showed that taking the collocation points on a finite-element (FE) mesh, which is built thanks to domain expertise and provides *a priori* knowledge of high gradient location, improves significantly the accuracy of the prediction instead of taking the points randomly in the domain. However, to produce the FE mesh, expert physical knowledge is required. In this work, we show that FBOAL and other adaptive sampling methods RAD and RAD-D are able to infer automatically position for collocation points, and thus improve the performance of PINNs without the need for any FE mesh.

For the architecture of PINNs, we use a feedforward network with 5 hidden layers and 100 neurons per layer. To minimize the cost function, we adopt Adam



(a) Sketch of the 2D configuration and geometrical setup



(b) Density of collocation points

Fig. 7: Calendering process: Configuration and density of training points.

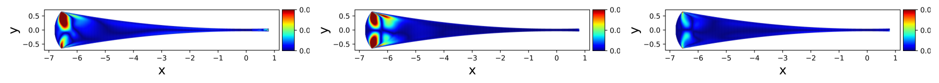
optimizer with the learning rate decay strategy. The results are obtained with 50,000 epochs with the learning rate $lr = 10^{-3}$, 100,000 epochs with the learning rate $lr = 10^{-4}$ and 150,000 epochs with $lr = 10^{-5}$. For the training of PINNs, we suppose to dispose of the boundary condition for the velocity and $N_T = 500$ points of measurements for the temperature that are randomly distributed inside the domain. The number of collocation points is fixed as $N_{pde} = 10,000$ points. For the training of FBOAL, in this primary investigation, we do not divide the domain. After every $k = 25,000$ iterations, we add and remove $m = 2.5\% \times N_{pde} = 250$ collocation points based on the PDE residuals. Again, we note that in this case, since there are four PDEs residuals, we divide the set of collocation points into four separated subsets whose cardinal equal to $N_{pde}/4 = 2,500$, and then add and remove $m/4$ points independently for each equation. For the training of RAD, we choose $k = 1$, $c = 1$ and effectuate the re-sampling after every $25,000$ iterations. For the training of RAR-D, we choose $k = 2$, $c = 0$ after every $25,000$ iterations we add 250 new collocation points to the training set. Figure 7b shows the density of collocation points produced by different methods. The FE mesh is much finer at the output of the calender compared to other methods, where there are high-gradient locations. However, with FBOAL, we see that the collocation points are not only added to the output of the calender, but also to the input where there is contact with the solid rolls. The same conclusion can be drawn with RAR-D. With RAD, as this method redistributes all the collocation points at the same time, the observation for important zones is not as clear as in other methods.

To assess the performance of PINNs, we evaluate the errors of the prediction on a random mesh that contains $N = 162,690$ points. To avoid any artificial high values of the error for fields very close to zero, we use a relative $\mathcal{L}^2$ error that divides the absolute error by the reference field amplitude, which is defined as $\epsilon_w = \dfrac{||w - \hat{w}||_2}{w_{max} - w_{min}}$ where w denotes the reference simulated field of interest and $\hat{w}$ is the corresponding PINNs prediction. Table 3 shows the performance of PINNs with different cases of collocation points in terms of relative $\mathcal{L}^2$ error. We see that PINNs with adaptive sampling methods give better accuracy for all physical fields than classical PINNs with random collocation points. The classical PINNs with collocation points on the FE mesh still produce the most

	ϵ_T	ϵ_{u_x}	ϵ_{u_y}	ϵ_P
Random points	13.6 ± 1.17	24.1 ± 3.77	15.6 ± 2.73	11.9 ± 1.33
FE mesh	**8.54 ± 1.39**	**14.7 ± 2.49**	18.0 ± 2.61	**1.41 ± 0.17**
FBOAL	10.5 ± 1.82	19.3 ± 1.85	**9.63 ± 3.24**	5.13 ± 0.72
RAD	10.2 ± 1.41	20.5 ± 1.96	**9.71 ± 2.55**	5.08 ± 0.76
RAR-D	12.3 ± 1.74	18.7 ± 1.65	11.1 ± 2.90	4.87 ± 0.69

Table 3: Calendering process: Relative $\mathcal{L}^2$ errors between reference solution and PINNs prediction with different cases of collocation points.

accurate prediction for T, u_x and p. However, for the vertical velocity component u_y, FBOAL and RAD give the best prediction. This is because the solution for u_y at the input of the calender has more complex behavior than at other zones. The adaptive sampling methods are able to capture this complexity based on the PDEs residuals. The classical PINNs with the collocation points on FE mesh give the largest errors for u_y since the FE mesh is very coarse at the input of the calender. With adaptive sampling methods, the algorithm is able to detect new zones which produce high errors for the PDEs residuals. Figure 8 illustrates the absolute error between the reference solution and the prediction produced by different methodologies for u_y. Again, we see that with adaptive sampling methods, the errors are significantly reduced compared to the classical approach with the collocation points taken on the FE mesh or random points.



(a) Random points **(b)** FE mesh **(c)** FBOAL

Fig. 8: Calendering process: Absolute error between the reference solution and the prediction produced by different methodologies for the vertical velocity component u_y.

4 Conclusion

In this paper, we introduced a Fixed-Budget Online Adaptive Learning (FBOAL) for the collocation points in PINNs that adds and removes points based on PDEs residuals on sub-domains while fixing the number of training points. The numerical results in academic test cases showed that FBOAL provides better accuracy with fewer iterations than classical PINNs. Besides that, we also compared the performance of FBOAL with other existing methods of adaptive sampling for the collocation points such as Residual-based Adaptive Distribution (RAD) and Residual-based Adaptive Refinement with Distribution (RAR-D). It is shown that in most cases, FBOAL is able to provide a comparable or even better performance than other approaches in terms of accuracy and computation cost. We then apply the methods FBOAL, RAD, and RAR-D to the rubber calendering process simulation. PINNs with these adaptive sampling methods give remarkably better predictions for the vertical velocity component than classical PINNs with the collocation points taken on an expert-designed FE mesh. This promising result demonstrates that the adaptive sampling methods can help to provide *a prior* knowledge of high-gradient locations and improve the conventional numerical solver in the construction of the mesh.

References

1. Daw, A., Bu, J., Wang, S., Perdikaris, P., Karpatne, A.: Rethinking the importance of sampling in physics-informed neural networks. arXiv:2207.02338 (2022)
2. Fu, J., Xiao, D., Fu, R., Li, C., Zhu, C., Arcucci, R., Navon, I.M.: Physics-data combined machine learning for parametric reduced-order modelling of nonlinear dynamical systems in small-data regimes. Computer Methods in Applied Mechanics and Engineering **404**, 115771 (2023)
3. Karniadakis, G.E., Kevrekidis, I.G., Lu, L., Perdikaris, P., Wang, S., Yang, L.: Physics-informed machine learning. Nature Reviews Physics **3**(6), 422–440 (2021)
4. Liu, Z., Yang, Y., Cai, Q.D.: Solving differential equation with constrained multilayer feedforward network. arXiv:1904.06619 (2019)
5. Lu, L., Meng, X., Mao, Z., Karniadakis, G.E.: Deepxde: A deep learning library for solving differential equations. SIAM Review **63**(1), 208–228 (2021)
6. Nabian, M.A., Gladstone, R.J., Meidani, H.: Efficient training of physics-informed neural networks via importance sampling. Computer-Aided Civil and Infrastructure Engineering **36**(8), 962–977 (2021)
7. Nguyen, T.N.K., Dairay, T., Meunier, R., Mougeot, M.: Physics-informed neural networks for non-newtonian fluid thermo-mechanical problems: an application to rubber calendering process. Engineering Applications of Artificial Intelligence **114**, 105176 (2022)
8. 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. Journal of Computational Physics **378**, 686–707 (2019)
9. Shin, Y., Darbon, J., Karniadakis, G.E.: On the convergence of physics informed neural networks for linear second-order elliptic and parabolic type pdes. arXiv:2004.01806 (2020)
10. 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. Computer Methods in Applied Mechanics and Engineering **403**, 115671 (2023)