Adaptive Global Modeling using Neural Networks with Deep Ensembles and Space-filling Sequences

Pavankumar Koratikere[1][0009−0007−1491−4654], Leifur Leifsson[1][0000−0001−5134−870X], Slawomir Koziel[2,3][0000−0002−9063−2647], and Anna Pietrenko-Dabrowska[3][0000−0003−2319−6782]

[1] School of Aeronautics and Astronautics, Purdue University, West Lafayette, Indiana 47907, USA
{pkoratik@purdue.edu, leifur@purdue.edu}
[2] Engineering Optimization & Modeling Center, Department of Engineering, Reykjavík University, Menntavegur 1, 102 Reykjavík, Iceland
koziel@ru.is
[3] Faculty of Electronics Telecommunications and Informatics, Gdansk University of Technology, Narutowicza 11/12, 80-233 Gdansk, Poland
anna.dabrowska@pg.edu.pl

Abstract. Global approximation models are often used in design and analysis activities in lieu of expensive simulations. Surrogate modeling with adaptive sampling is an efficient approach for creating these global models. There is a growing interest in global neural network (NN) modeling since it can approximate complex functions and can handle large datasets. Hence, this work proposes a novel method, called separate adaptive sampling (SAS), for creating global models using NNs. SAS performs exploration and exploitation using two criteria, unlike existing methods which use a single criterion. An exploration point is obtained from a space-filling sampling algorithm, while an exploitation point is obtained by maximizing the uncertainty in the NN prediction. Three existing global modeling algorithms are used for comparison. These algorithms are demonstrated on three test cases. The first two test cases are analytical functions with 3 and 8 dimensions, while the third test case is a physics-based airfoil modeling problem consisting of 16 dimensions. SAS performs best for the analytical cases while achieving comparable performance for the third case. Moreover, the NN training time for each method is nearly constant as the number of samples increase.

Keywords: global modeling · neural networks · adaptive sampling · space-filling sampling

1 Introduction

Computer simulations have become an essential tool for analyzing and designing systems. These simulations, such as computational fluid dynamics, solve a set of

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97557-8_25

mathematical equations that represent a complex phenomenon. Hence, the computational resources required for these simulations is often considerable, limiting their direct application in design activities. This issue can be alleviated by using a surrogate model [2], which approximates the simulation output while offering faster evaluation times. Consequently, surrogate models have been widely adopted in various design and analysis activities [14].

Some of the popular models are kriging and neural networks (NNs). Kriging is an interpolating model that approximates function responses based on spatial correlation between samples [2]. However, the computational cost scales cubically with increasing number of samples [17]. Meanwhile, NNs can model complex functions and also scales better with larger datasets [4]. Hence, this work focuses on global modeling using NNs.

A global model approximates the underlying system across the entire design space [10]. There are two primary approaches for creating a global model: one-shot sampling and adaptive sampling [10]. In one-shot sampling, a design of experiment (DOE) is created using a sampling technique such as Latin hypercube sampling (LHS) [12]. The corresponding system output is then evaluated. Next, a surrogate model is created using the dataset and if the model's accuracy is insufficient, then the entire dataset is discarded, and the process is repeated with a larger DOE. This approach is inefficient for global modeling as it discards all the generated samples [1].

In adaptive sampling, a surrogate model is iteratively refined by adding new sample points [3]. Specifically, an initial DOE is created and the system is evaluated to obtain the corresponding observations. Next, a surrogate model is built using the dataset and if the model's accuracy is insufficient, then new samples are added to the DOE using an infill criterion. This iterative process continues until a stopping criterion is met. In this approach, the infill criterion plays a key role by balancing exploration and exploitation. Liu et al. [10] and Fuhg et al. [3] provide a detailed review of global modeling algorithms.

Many approaches have been proposed for global NN modeling. For instance, Gupta et al. [5] proposed an adaptive sampling strategy that uses information matrix and maximin distance criterion to select infill points. However, the computational cost for this method grows nonlinearly with increasing number of samples. Eason and Cremaschi [1] introduced the mixed adaptive sampling algorithm (MASA) that uses uncertainty in the NN prediction and the distance between samples for identifying the next point. Here, k-fold cross-validation was used to compute the uncertainty. However, recently more effective methods have been proposed for estimating this uncertainty such as deep ensembles [9].

In this work, a novel algorithm is proposed for global NN modeling, called separate adaptive sampling (SAS), that performs exploration and exploitation separately. Instead of using a single infill criterion, SAS consists of two separate criteria, one for exploration and the other for exploitation, and hence, adds two points in each iteration. The exploitation point is obtained by maximizing the uncertainty in the NN prediction. The exploration point is selected using a space-filling sampling sequence. SAS is demonstrated on three problems consisting of

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97557-8_25

3, 8, and 16 dimensions. For comparison, MASA is implemented, along with pure exploration- and pure exploitation-based methods.

The remainder of this paper is organized as follows. Section 2 describes the proposed adaptive global modeling technique along with the other three methods. Section 3 outlines the three test problems and discusses the results. Section 4 summarizes this work and provides some recommendations for future work.

2 Methods

This section first describes a general adaptive sampling framework used in this work for creating global NN models. Then, the space-filling sampling method is presented, which is used as a part of the infill criteria. Lastly, it describes four different methods investigated in this work.

2.1 Global surrogate modeling using adaptive approach

A surrogate model is a simple, cheaper to evaluate approximation of a computationally expensive function $f\colon \mathbb{R}^n \to \mathbb{R}$. These models are used in design and analysis tasks that require repeated evaluations of the underlying function. Hence, a global model is often preferred, as it approximates a system over the entire design space. One of the ways to efficiently create a global model is to use an adaptive approach, where the model is gradually refined based on a criterion. Figure 1 illustrates an adaptive global modeling framework used in this work.

The process starts with an initial sampling plan $\mathbf{X}$ created using a DOE technique. The function f is then evaluated to obtain the output $\mathbf{y}$. Next, the iterative phase begins in which a surrogate model is created using the dataset $(\mathbf{X}, \mathbf{y})$. Then, an error metric is calculated using a test dataset to measure the model's accuracy. If the stopping criterion is not met, then a new sample $\mathbf{x}_{new}$ is selected using an infill criterion, and its output y_{new} is evaluated. The new data point $(\mathbf{x}_{new}, y_{new})$ is added to the training data and the surrogate model is

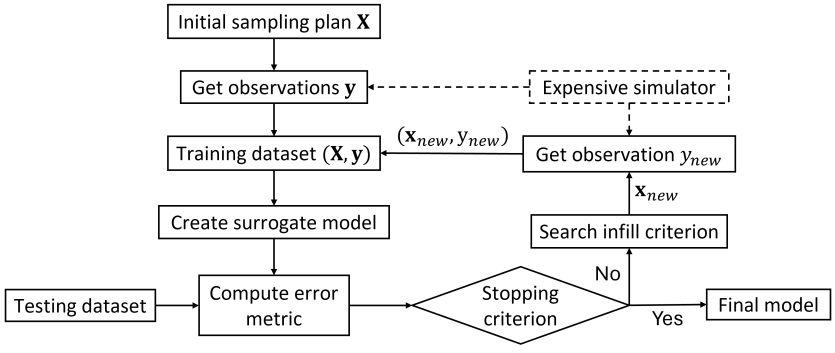

Fig. 1: The adaptive global modeling framework used in this work.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97557-8_25

retrained. This process repeats until the stopping criterion is met. In this work, the error metric is the normalized root mean squared error (NRMSE), which is written as

$$\text{NRMSE} = \frac{1}{y_{max} - y_{min}} \sqrt{\frac{1}{N} \sum_{i=1}^{N} \left[y^{(i)} - \hat{y}^{(i)} \right]^2}. \tag{1}$$

The y and $\hat{y}$ is the true and predicted value, respectively, while N is the number of samples. The y_{max} and y_{min} represent the maximum and minimum y values in the dataset, respectively. The maximum number of infill points is used as the stopping criterion.

Neural network (NN) is a machine learning model inspired by the human brain [4]. It consists of layers of interconnected nodes (neurons), where each node processes the input data using an activation function and passes the output to the next layer. The sigmoid linear unit is used as the activation function in this work. The number of layers and nodes isset aat thethe start aisis not changed during the iterations. Since the number of infill points is much larger than the initial DOE, NN is designed to overfit initially, ensuring enough flexibility at the start. As the training data increases, the model improves and the overfitting decreases.

The NN model is parameterized using weights and biases, which determine the strength of connections between nodes. These parameters are obtained by minimizing the mean squared error between the predicted and true values. In this work, the ADAM optimizer [7] is used for NN training with a learning rate of 10^{-2}. The optimization process is terminated when the NRMSE of the NN model for the training data is less than 10^{-4}. This ensures that the NN interpolates through the data without significant overfitting. Since only a few points are added in each iteration, the NN is not retrained from scratch. Instead, training resumes from the last iteration but with the updated dataset. PyTorch is used to build and train the NN model.

Another important aspect of the framework is the search infill criterion, which selects the next sample point by balancing exploration and exploitation. In this work, four different infill criteria are explored for global surrogate modeling. All of these criteria are described in the subsequent sections.

2.2 Space-filling sequential sampling

In this work, the fully sequential space-filling (FSSF) [16] sampling technique is used within different methods, so it is introduced first. FSSF is a recently proposed two-phase sampling method. In the first phase, a large set of points (known as the candidate set) is generated using a space-filling technique, such as the Sobol' sequence. In the second phase, the desired sampling plan is created by sequentially selecting points from the candidate set using a distance criterion.

The pseudo-code for FSSF is outlined in Algorithm 1 as a class with three functions. The initialization function handles the first phase by defining variables and generating the candidate set $\mathcal{C}$. It takes N_{max} (total required points) and n

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97557-8_25

Algorithm 1 Fully sequential space-filling sampling class [16]

procedure INITIALIZATION(N_{max}, n)
  $M \leftarrow 1000n + 2N_{max}$
  Generate candidate set $\mathcal{C}$ of size M using Sobol' sequence
  Initialize $\mathcal{D}$ of size M
  for $i = 1, \ldots, M$ **do**
    $\mathcal{D}_i \leftarrow$ distance of $\mathbf{x}_i$ from the closest design space boundary
  end for
end procedure
procedure UPDATESAMPLING($\mathbf{x}_{new}$)
  for $j = 1, \ldots, M$ **do**
    $\mathcal{D}_j \leftarrow \min(\mathcal{D}_j, d(\mathbf{x}_j, \mathbf{x}_{new}))$ ▷ update distance criterion based on $\mathbf{x}_{new}$
  end for
end procedure
procedure GENERATESAMPLES(N)
  for $i = 1, \ldots, N$ **do**
    $\mathbf{x}_i \leftarrow$ find the point having maximum value in $\mathcal{D}$
    UpdateSampling($\mathbf{x_i}$)
  end for
  return $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
end procedure

(problem dimension) as inputs. The size of candidate set (M) is computed, and then Sobol' sequence is used to generate M points. Then, the distance criterion is computed for each point in the set.

The other two functions handle the second phase of FSSF. The update sampling function takes a newly added sample point $\mathbf{x}_{new}$ as input and adjusts the distance criterion for each point in $\mathcal{C}$. This ensures that points near $\mathbf{x}_{new}$ will have a lower criterion value and are less likely to be selected in later iterations. This function is called whenever a point is added to the DOE.

The generate samples function takes N, the number of points to be generated, as input and uses an iterative process to generate them. The first step in the loop is to identify the next point $\mathbf{x}_i$ in the sequence , and then call the update sampling function with $\mathbf{x}_i$ as input. This function is used to create the initial DOE and then iteratively add new points without disrupting the space-filling properties.

2.3 Exploration-based global neural network modeling

The exploration-based global NN modeling uses an infill criterion that performs pure exploration. The objective of this infill criterion is to add points in the unexplored regions. This can be achieved by using space-filling sampling techniques. In this work, the FSSF method [16] is used which is shown to have better space-filling properties than Sobol' sequences, Sec. 2.2 describes the FSSF method in detail.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97557-8_25

Algorithm 2 Exploration-based global NN modeling (adapted from [2])

set the value of n, N_{init}, Q
initialization($N_{init} + Q, n$) ▷ initialize FSSF, c.f. Algorithm 1
$\mathbf{X} \leftarrow$ generateSamples(N_{init}) ▷ initial DOE generated using FSSF
$\mathbf{y} = f(\mathbf{X})$ ▷ compute output for $\mathbf{X}$
$q = 0$ ▷ initialize number of infills
while $q \leq Q$ **do** ▷ Q is total infill budget
    $\hat{y}(\mathbf{x}) \leftarrow$ fit a NN model to $(\mathbf{X}, \mathbf{y})$ dataset
    Compute the NRMSE of the $\hat{y}$ model using testing data
    $\mathbf{x}_{new} \leftarrow$ generateSamples(1) ▷ next sample in the sequence $\mathcal{S}$
    $\mathbf{X}, \mathbf{y} \leftarrow \mathbf{X} \cup \mathbf{x}_{new}, \mathbf{y} \cup f(\mathbf{x}_{new})$ ▷ append dataset
    $q = q + 1$ ▷ update number of infills
end while
return $\hat{y}$ model

Algorithm 2 presents a pseudo-code for the exploration-based global NN modeling. The problem dimension n, the initial DOE size N_{init}, and the maximum number of infill points Q are defined at start. The initialization function from Algorithm 1 is then executed with $N_{init} + Q$ and n as inputs. The initial DOE $\mathbf{X}$ is constructed using the FSSF method by running the generate samples function from Algorithm 1. The underlying system is then evaluated to obtain $\mathbf{y}$. The variable q is also initialized to zero, which denotes the number of infill points added to the dataset.

Next, the iterative phase begins, in which the first step is to create a $\hat{y}$ NN model using the dataset $(\mathbf{X}, \mathbf{y})$. Then, the NRMSE of the $\hat{y}$ model is computed using the test dataset. The next infill point $\mathbf{x}_{new}$ is obtained by running the generate samples function with $N = 1$ as input. This new sample and its corresponding output are added to the dataset, and q is incremented by 1. This loop continues until the stopping criterion of maximum number of infill points is met.

2.4 Exploitation-based global neural network modeling

The exploitation-based global NN modeling aims to add infill points where the surrogate model's behavior is uncertain. Algorithm 3 provides a pseudo-code for this method, which is similar to Algorithm 2, but with a few differences. Firstly, the initial DOE can be generated using any sampling technique, but the FSSF method is used here to ensure consistency across all the methods. Secondly, the infill criterion is based on finding the point with the highest uncertainty in the NN prediction.

In this work, the deep ensembles (DE) [9] method is used to estimate the uncertainty due to its simplicity and ease of implementation. The DE approach consists of training an ensemble of NN models on the same dataset. The starting point for the NN training is randomly determined, which results in different predictions for a given input. The final prediction $\hat{y}(\mathbf{x})$ and the uncertainty $\hat{s}(\mathbf{x})$ is the mean and standard deviation of all predictions, respectively. The DE method

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97557-8_25

Algorithm 3 Exploitation-based global NN modeling (adapted from [2])

set the value of n, N_{init}, Q
initialization($N_{init} + Q, n$) ▷ initialize FSSF, c.f. Algorithm 1
$\mathbf{X} \leftarrow$ generateSamples(N_{init}) ▷ initial DOE generated using FSSF
$\mathbf{y} = f(\mathbf{X})$ ▷ compute output for $\mathbf{X}$
$q = 0$ ▷ initialize number of infills
while $q \leq Q$ **do** ▷ Q is total infill budget
    $\hat{y}(\mathbf{x})$, $\hat{s}(\mathbf{x}) \leftarrow$ fit NN models to $(\mathbf{X}, \mathbf{y})$ dataset using DE method
    Compute the NRMSE of the $\hat{y}$ model using testing data
    $\mathbf{x}_{new} \leftarrow \arg\max \hat{s}(\mathbf{x})$
    $\mathbf{X}, \mathbf{y} \leftarrow \mathbf{X} \cup \mathbf{x}_{new}, \mathbf{y} \cup f(\mathbf{x}_{new})$ ▷ append dataset
    $q = q + 1$ ▷ update number of infills
end while
return $\hat{y}$ model

is proposed for quantifying both aleatoric and epistemic uncertainty. However, it is assumed that the function to be modeled is deterministic, and hence, there is no aleatoric uncertainty. This assumption simplifies the DE method to a standard ensembles approach. The ensemble consists of five NN models, refer [9] for more details. In this work, differential evolution [18] is used to maximize the uncertainty. The remaining steps are similar to those in Algorithm 2.

2.5 Mixed adaptive sampling algorithm

The mixed adaptive sampling algorithm (MASA) [1] uses an infill criterion that performs exploration and exploitation using a single criterion. The pseudo-code for MASA is similar to Algorithm 3, with the only difference being the infill criterion. Mathematically, the infill point is obtained as

$$\mathbf{x}_{new} = \arg\max_{\mathbf{x}} \ \hat{s}(\mathbf{x})/s_{max} + d(\mathbf{x}, \mathbf{X})/d_{max}. \tag{2}$$

The exploitation part, $\hat{s}(\mathbf{x})/s_{max}$, accounts for the uncertainty in the NN prediction, while the exploration part, $d(\mathbf{x}, \mathbf{X})/d_{max}$, ensures new point is added in the unexplored regions. The $\hat{s}(\mathbf{x})$ denotes the uncertainty in the NN prediction which is computed using the DE approach discussed in section 2.4. The $d(\mathbf{x}, \mathbf{X})$ denotes the euclidean distance of $\mathbf{x}$ to the closest point in the dataset $\mathbf{X}$. The s_{max} and d_{max} are used to normalize the $\hat{s}(\mathbf{x})$ and $d(\mathbf{x}, \mathbf{X})$, respectively. In this work, s_{max} is obtained by maximizing the $\hat{s}(\mathbf{x})$, and d_{max} corresponds to the euclidean distance between the farthest points in the dataset $\mathbf{X}$. In this work, differential evolution [18] is used to numerically solve (2).

2.6 Separate adaptive sampling algorithm

The proposed separate adaptive sampling (SAS) algorithm performs exploitation and exploration using two different criteria, unlike MASA which uses a single

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97557-8_25

Algorithm 4 Separate adaptive sampling algorithm (this work)

set the value of n, N_{init}, Q
initialization($N_{init} + Q, n$) ▷ initialize FSSF, c.f. Algorithm 1
$\mathbf{X} \leftarrow$ generateSamples(N_{init}) ▷ initial DOE generated using FSSF
$\mathbf{y} = f(\mathbf{X})$ ▷ compute output for $\mathbf{X}$
while $q \leq Q$ **do** ▷ Q is total infill budget
    $\hat{y}(\mathbf{x})$, $\hat{s}(\mathbf{x}) \leftarrow$ fit NN models to $(\mathbf{X}, \mathbf{y})$ dataset using DE method
    Compute the NRMSE of the $\hat{y}$ model using testing data
    $\mathbf{x}_{exploit} \leftarrow \arg\max \hat{\sigma}(\mathbf{x})$ ▷ exploitation
    $\mathbf{X}, \mathbf{y} \leftarrow \mathbf{X} \cup \mathbf{x}_{exploit}, \mathbf{y} \cup f(\mathbf{x}_{exploit})$
    updateSampling($\mathbf{x}_{exploit}$) ▷ update FSSF about exploitation point
    $\mathbf{x}_{explore} \leftarrow$ generateSamples(1) ▷ exploration
    $\mathbf{X}, \mathbf{y} \leftarrow \mathbf{X} \cup \mathbf{x}_{explore}, \mathbf{y} \cup f(\mathbf{x}_{explore})$
    $q = q + 2$ ▷ update number of infills
end while
return $\hat{y}$ model

criterion. Algorithm 4 provides a pseudo-code for SAS which is similar to the previous algorithms. The $\mathbf{x}_{exploit}$ is obtained by maximizing the uncertainty in the NN prediction, similar to the exploitation-based method in Sec. 2.4. Before generating the exploration point, the update sampling function from Algorithm 1 is run with $\mathbf{x}_{exploit}$ as the input. This ensures that exploration point is not added around the exploitation points. Then, the $\mathbf{x}_{explore}$ is generated by running the generate samples function with $N = 1$ as the input. The variable q is incremented by 2 to account for both infill points. All other steps are similar to the previously discussed algorithms.

3 Numerical Experiments

This section presents three test cases used to demonstrate and compare the results of the global NN modeling algorithms described in Sec. 2. Each method is run 10 times to account for the randomness in NN training. The initial DOE is same across all methods to ensure a consistent starting point. The convergence history for all 10 runs is shown as a convergence band. The center-line represents the median, while the upper and lower limits represent 10^{th} and 90^{th} percentile, respectively. For all the problems, testing dataset consists of 50 LHS [12] points.

3.1 Ishigami function

This section presents the Ishigami function and discusses the results of applying different global modeling methods. The Ishigami function [6] is written as:

$$f(\mathbf{x}) = \sin(x_1) + a\sin^2(x_2) + bx_3^4\sin(x_1), \tag{3}$$

where $a = 7$, $b = 0.1$, and $x_i \in [-\pi, \pi]$, $\forall\ i = 1, 2, 3$. The initial DOE consists of 25 samples generated using the FSSF method, and the maximum number of infill points is set to 100. The NN model has three layers, each with 20 neurons.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97557-8_25

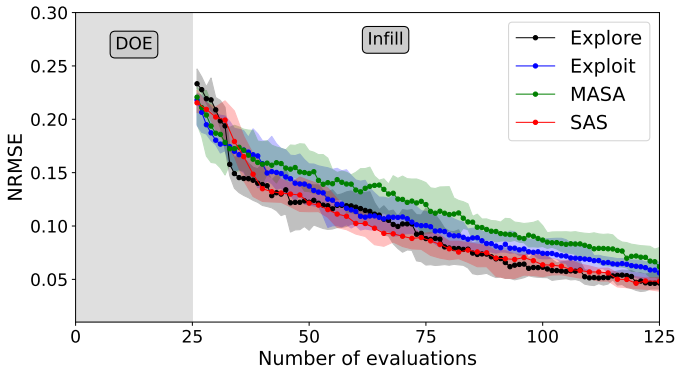

Fig. 2: Convergence history of the NRMSE for the Ishigami function.

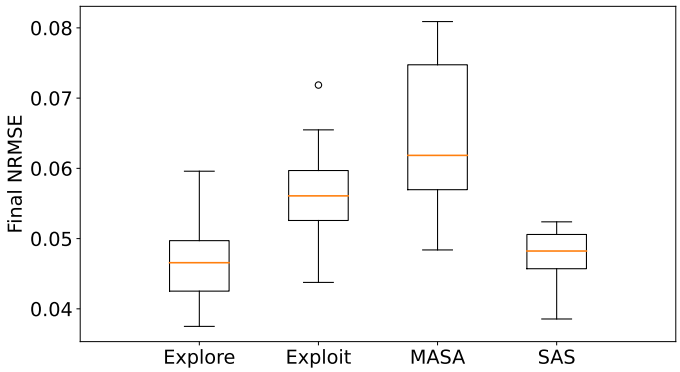

Fig. 3: Final NRMSE of the NN model for the Ishigami function.

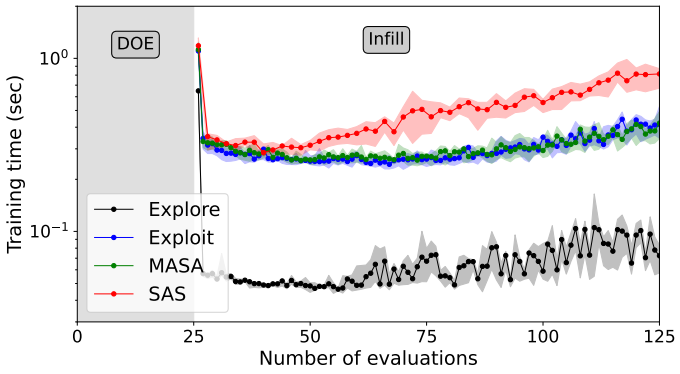

Fig. 4: Training time of the NN model for the Ishigami function.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97557-8_25

Figure 2 shows the convergence history of the NRMSE as the number of samples increase. The exploration-based method and SAS have slightly better convergence rate, while MASA has the slowest convergence. However, all the methods perform comparably. Figure 3 compares the final NRMSE of the NN model obtained by each method. The exploration-based method has the best median NRMSE value but exhibits a large variation. On the other hand, SAS achieves a comparable median NRMSE value with least variation. The exploitation-based method and MASA do not perform as well for this problem. This indicates that pure exploration might be enough for this problem. Figure 4 shows the NN model training time as the number of samples increases. The exploration-based method has the shortest training time since it uses only one NN model, while the other methods use five. All methods scale much better than kriging, which scales cubically with the number of samples [17]. This makes global surrogate modeling with NN attractive for large datasets.

3.2 Borehole function

The borehole function [13] models the flow of water through a borehole and is written as:

$$f(\mathbf{x}) = \frac{2\pi T_u (H_u - H_l)}{\ln(r/r_w) \cdot \left(1 + \frac{2LT_u}{\ln(r/r_w) r_w^2 K_w} + \frac{T_u}{T_l}\right)}, \tag{4}$$

where $\mathbf{x} = \begin{bmatrix} r_w & r & T_u & H_u & T_l & H_l & L & K_w \end{bmatrix}^T$. The upper and lower bound for the variables are provided in Table 1. The function f represents the water flow rate in m^3/yr. The initial DOE consists of 25 samples and the maximum number of infill points is set to 100. The NN model consists of two layers, each having 20 neurons.

Figure 5 shows the evolution of NRMSE as the number of samples are increased. For this test case, SAS outperforms all the other methods, especially after 100 function evaluations. The exploitation-based method and MASA perform comparably to SAS up to 100 evaluations, but their convergence rate slows

Table 1: Bounds for the variables of the borehole function

Variable	Lower bound	Upper bound
r_w (m)	0.05	0.15
r (m)	100	50000
T_u (m^2/yr)	63070	115600
H_u (m)	990	1110
T_l (m^2/yr)	63.1	116
H_l (m)	700	820
L (m)	1120	1680
K_w (m/yr)	9855	12045

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97557-8_25

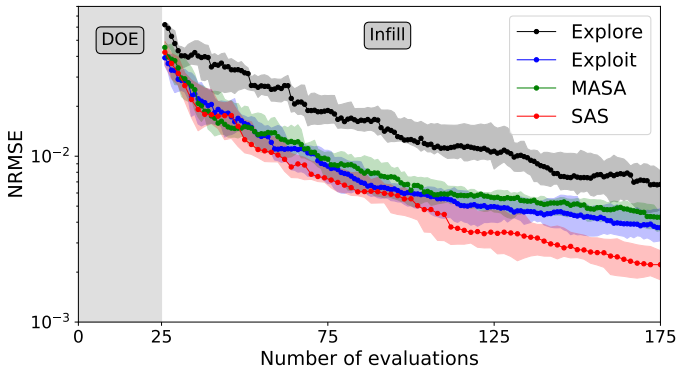

Fig. 5: Convergence history of the NRMSE for the borehole function.

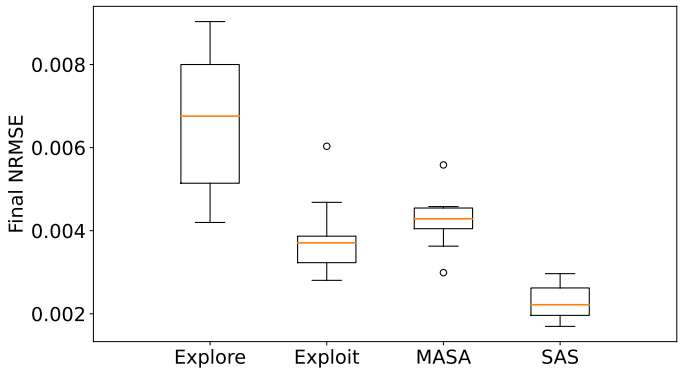

Fig. 6: Final NRMSE of the NN model for the borehole function.

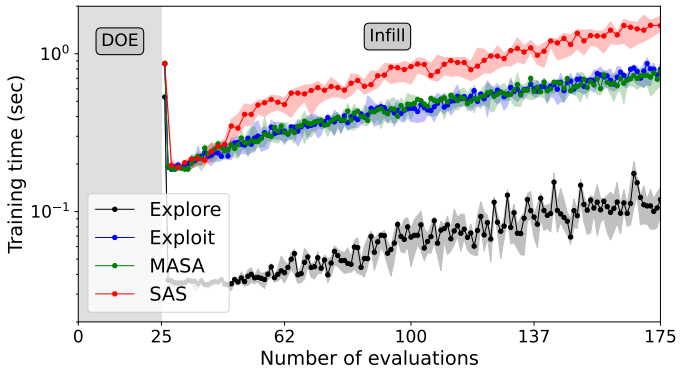

Fig. 7: Training time of the NN model for the borehole function.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97557-8_25

down beyond that point. The exploration-based method has the slowest rate of convergence among all methods. This suggests that performing pure exploration is not enough for this test case.

Figure 6 compares the final NRMSE values obtained by each method. SAS achieves the best final NRMSE as compared to other methods, with the least variation. The exploration-based method has the highest NRMSE value and the most variation. Both exploitation-based method and MASA yield comparable final NRMSE value. This suggests that the distance-based exploration term used in MASA does not improve the performance for this problem. Figure 7 shows how NN training time changes as the number of samples increase. The exploration-based method has the shortest training time since it uses only one NN model, while the other methods use five NN models. Since two infill points are added in each iteration, the training time for SAS is slightly longer.

3.3 Airfoil drag coefficient modeling

The airfoil modeling problem involves predicting the drag coefficient (C_d) of a given airfoil in viscous transonic flow conditions. The variables in the problem include the shape of the airfoil, the angle of attack (α), and the free-stream mach number (M_∞). The airfoil shape is parameterized using the class-shape transformation (CST) [8] technique. In this work, 14 CST coefficients are used to represent the airfoil shape, 7 for the upper surface and 7 for the lower surface. In total, the problem consists of 16 variables.

The upper and lower bounds for the CST coefficients are determined by perturbing the CST coefficients for the RAE 2822 airfoil by ±30%. The $\alpha \in [1.5, 4.5]$ and $M_\infty \in [0.6, 0.8]$, while the Reynolds number (Re_∞) is fixed at 6 million. Figure 8 illustrates these variables and their respective bounds. The flow around the airfoil is computed using ADflow [11], an open-source finite volume solver. The computational grid uses an o-mesh generated using pyHyp [15]. The initial DOE consists of 50 samples generated using the FSSF method and the maximum number of infill points is set to 100. The NN model consists of two hidden layers with 20 neurons in each layer.

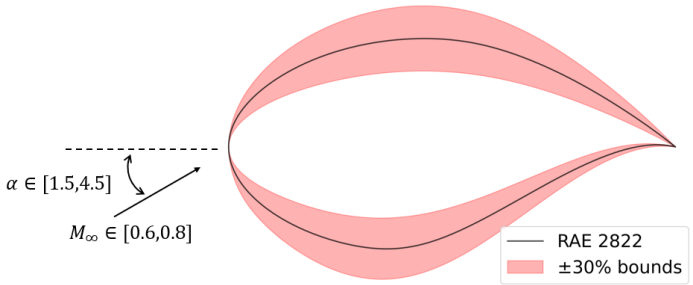

Fig. 8: Schematic describing the design space of the airfoil modeling problem.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97557-8_25

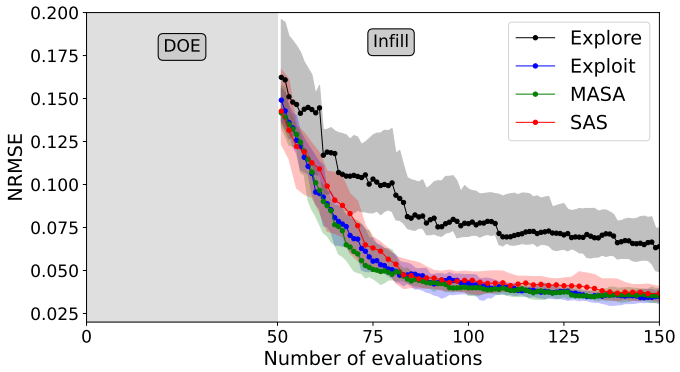

Fig. 9: Convergence history of the NRMSE for the airfoil modeling problem.

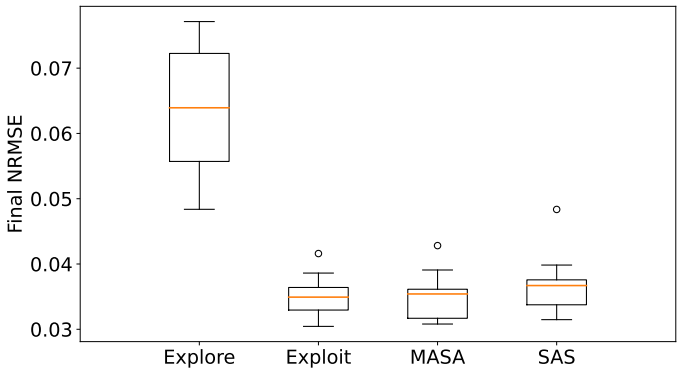

Fig. 10: Final NRMSE of the NN model for the airfoil modeling problem.

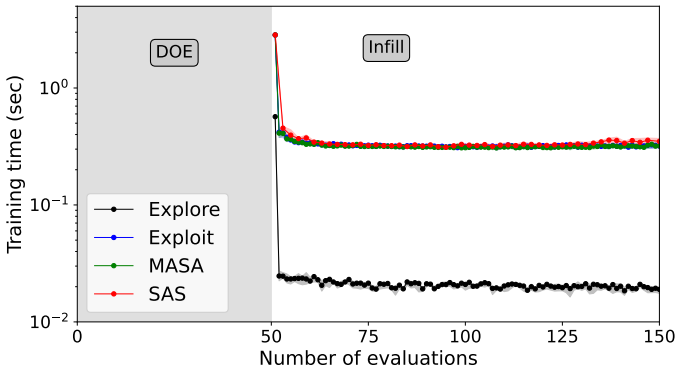

Fig. 11: Training time of the NN model for the airfoil modeling problem.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97557-8_25

Figure 9 shows the evolution of NRMSE as the number of samples increase. The exploitation-based method, MASA, and SAS perform comparably, with similar convergence rates. However, the exploration-based method fails to reduce the NRMSE after 100 function evaluations. Since the C_d is highly nonlinear in the transonic regime, the pure exploration-based strategy does not perform well.

Figure 10 shows the final NRMSE of the NN model. The SAS, MASA, and exploitation-based method have comparable median final NRMSE value, with SAS exhibiting the least variation. While some final NRMSE values from MASA and the exploitation-based method are better than SAS, both methods show larger variation. The exploration-based method has the highest final NRMSE and the largest variation for the reasons described earlier.

Figure 11 shows the NN model training time as the number of samples increase. As in previous test problems, the exploration-based method has the shortest training time. Additionally, the training time for all the methods remains nearly constant, even with 16 variables in the problem. This demonstrates the effectiveness of the NN for the global NN modeling.

4 Conclusion

This work proposes a novel infill criterion for global neural network (NN) modeling. The proposed method, called separate adaptive sampling (SAS), performs exploration and exploitation using different criterion, and hence, adds two infill points in each iteration. The first point is obtained by maximizing the uncertainty in the NN prediction, while the second point is obtained from a space-filling sequential sampling method. Uncertainty in the NN prediction is estimated using the deep ensembles approach. The study also explores the mixed adaptive sampling algorithm, which balances exploration and exploitation with a single criterion, as well as pure exploitation- and pure exploration-based methods.

The algorithms described in this work are demonstrated on three test cases with varying levels of difficulty. SAS performed the best for the first two test cases and performed comparably for the third problem. The pure exploration-based method did not yield good results for highly nonlinear test cases. However, for all problems, NN training time scales well as the dataset size increased, highlighting its advantage over the kriging model.

Future work will focus on testing the proposed method on high-dimensional global modeling problems requiring many infill points. The uncertainty in the NN prediction can be estimated using other methods to improve the exploitation. Moreover, a non-uncertainty based criterion can be explored for exploitation, as SAS decouples exploration and exploitation.

Acknowledgments

This work was supported in part by the U.S. National Science Foundation (NSF) award number 2223732 and by the Icelandic Centre for Research (RANNIS) award number 239858.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97557-8_25

References

1. Eason, J., Cremaschi, S.: Adaptive sequential sampling for surrogate model generation with artificial neural networks. Computers & Chemical Engineering **68**, 220–232 (Sep 2014)
2. Forrester, A.I.J., Sóbester, A., Keane, A.J.: Engineering Design via Surrogate Modelling: A Practical Guide. Wiley (Jul 2008)
3. Fuhg, J.N., Fau, A., Nackenhorst, U.: State-of-the-art and comparative review of adaptive sampling methods for kriging. Archives of Computational Methods in Engineering **28**(4), 2689–2747 (Aug 2020)
4. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016)
5. Gupta, S., Paudel, A., Thapa, M., Mulani, S.B., Walters, R.: Adaptive sampling-based artificial neural network for surrogate modeling. In: AIAA SCITECH 2022 Forum. American Institute of Aeronautics and Astronautics (Jan 2022)
6. Ishigami, T., Homma, T.: An importance quantification technique in uncertainty analysis for computer models. In: [1990] Proceedings. First International Symposium on Uncertainty Modeling and Analysis. p. 398–403. IEEE Comput. Soc. Press
7. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2014)
8. Kulfan, B.M.: Universal parametric geometry representation method. Journal of Aircraft **45**(1), 142–158 (Jan 2008)
9. Lakshminarayanan, B., Pritzel, A., Blundell, C.: Simple and scalable predictive uncertainty estimation using deep ensembles. Advances in neural information processing systems **30** (2017)
10. Liu, H., Ong, Y.S., Cai, J.: A survey of adaptive sampling for global metamodeling in support of simulation-based complex engineering design. Structural and Multidisciplinary Optimization **57**(1), 393–416 (Jun 2017)
11. Mader, C.A., Kenway, G.K.W., Yildirim, A., Martins, J.R.R.A.: Adflow: An open-source computational fluid dynamics solver for aerodynamic and multidisciplinary optimization. Journal of Aerospace Information Systems **17**(9), 508–527 (Sep 2020)
12. Mckay, M.D., Beckman, R.J., Conover, W.J.: A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics **42**(1), 55–61 (Feb 2000)
13. Morris, M.D., Mitchell, T.J., Ylvisaker, D.: Bayesian design and analysis of computer experiments: Use of derivatives in surface prediction. Technometrics **35**(3), 243–255 (Aug 1993)
14. Queipo, N.V., Haftka, R.T., Shyy, W., Goel, T., Vaidyanathan, R., Kevin Tucker, P.: Surrogate-based analysis and optimization. Progress in Aerospace Sciences **41**(1), 1–28 (Jan 2005)
15. Secco, N.R., Kenway, G.K.W., He, P., Mader, C., Martins, J.R.R.A.: Efficient mesh generation and deformation for aerodynamic shape optimization. AIAA Journal **59**(4), 1151–1168 (Apr 2021)
16. Shang, B., Apley, D.W.: Fully-sequential space-filling design algorithms for computer experiments. Journal of Quality Technology **53**(2), 173–196 (2021)
17. Snoek, J., Rippel, O., Swersky, K., Kiros, R., Satish, N., Sundaram, N., Patwary, M., Prabhat, M., Adams, R.: Scalable bayesian optimization using deep neural networks. In: International conference on machine learning. pp. 2171–2180. PMLR (Jul 2015)
18. Storn, R., Price, K.: Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization **11**(4), 341–359 (1997)

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97557-8_25