Available online at www.sciencedirect.com
ScienceDirect

Comput. Methods Appl. Mech. Engrg. 415 (2023) 116226

Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

Gradient and uncertainty enhanced sequential sampling for global fit

Sven Lämmle[a,b,*], Can Bogoclu[c,1], Kevin Cremanns[d], Dirk Roos[b]

[a] ZF Friedrichshafen AG, Graf-von-Soden-Platz 1, Friedrichshafen, 88046, Baden-Wuerttemberg, Germany
[b] Institute of Modelling and High-Performance Computing, Niederrhein University of Applied Sciences, Reinarzstr. 49, Krefeld, 47805, North Rhine-Westphalia, Germany
[c] Zalando SE, Valeska-Gert-Straße 5, Berlin, 10243, Germany
[d] PI Probaligence GmbH, Technology Centre Augsburg, Am Technologiezentrum 5, Augsburg, 86159, Bavaria, Germany

Received 30 April 2023; received in revised form 28 June 2023; accepted 28 June 2023
Available online 22 July 2023

Dataset link: https://github.com/SvenL13/GALE

Abstract

Surrogate models based on machine learning methods have become an important part of modern engineering to replace costly computer simulations. The data used for creating a surrogate model are essential for the model accuracy and often restricted due to cost and time constraints. Adaptive sampling strategies have been shown to reduce the number of samples needed to create an accurate model. This paper proposes a new sampling strategy for global fit called Gradient and Uncertainty Enhanced Sequential Sampling (GUESS). The acquisition function uses two terms: the predictive posterior uncertainty of the surrogate model for exploration of unseen regions and a weighted approximation of the second and higher-order Taylor expansion values for exploitation. Although various sampling strategies have been proposed so far, the selection of a suitable method is not trivial. Therefore, we compared our proposed strategy to 9 adaptive sampling strategies for global surrogate modeling, based on 26 different 1 to 8-dimensional deterministic benchmarks functions. Results show that GUESS achieved on average the highest sample efficiency compared to other surrogate-based strategies on the tested examples. An ablation study considering the behavior of GUESS in higher dimensions and the importance of surrogate choice is also presented.

Keywords: Adaptive sampling; Bayesian optimization; Design of experiments; Gaussian process; ANN

1. Introduction

Computer simulations have become an essential part of engineering and are used for various applications (e.g. design optimization, structural mechanics, material science, etc.). Most of these applications require computationally demanding simulation models. Machine learning (ML) models have emerged to mitigate the computational cost and are used as surrogates. For this task, ML models learn the relationship between the response of the expensive simulations from a dataset containing past observations. Various ML methods were proposed as surrogate models

[*] Corresponding author at: ZF Friedrichshafen AG, Graf-von-Soden-Platz 1, Friedrichshafen, 88046, Baden-Wuerttemberg, Germany.
E-mail addresses: sven.laemmle@zf.com (S. Lämmle), can.bogoclu@zalando.de (C. Bogoclu), kevin.cremanns@probaligence.de (K. Cremanns), dirk.roos@hs-niederrhein.de (D. Roos).

[1] Work done while at Institute of Modelling and High-Performance Computing.

https://doi.org/10.1016/j.cma.2023.116226

(sometimes referred as response surface methods), e.g. as solvers for partial differential equations (PDEs) describing the behavior of high-dimensional vector spaces or fields [1–3], or probabilistic approximators of parametric response functions [4]. Among these types of surrogate models, Gaussian Process Regression (GP) [5] is a popular choice [6,7] because of its flexibility and ability to predict the model uncertainty. However, drawbacks are the selection of a suitable covariance (kernel) function that is application dependent, and the computational burden for larger data sets. Various extensions to GPs such as the deep GPs[8], sparse GPs based on variational inference [9,10], and efficient matrix decomposition [11] were developed to overcome some of these limitations. One particularly promising approach is the combination of GPs with Artificial Neural Networks (ANNs) to Deep Gaussian Covariance Networks (DGCNs) [12,13] to learn the non-stationary hyperparameters of the GP together with combinations of different covariance functions.

ANNs alone can handle large datasets with high-dimensional inputs but lack the ability to predict the model uncertainty, unlike GPs. However, different approaches were developed in the past to make ANNs uncertainty-aware based on dropout [14], Bayesian inference [15–18], or ensembles [19,20].

The design of experiments (DoE) [21], i.e. the choice of samples used for training the surrogate model, is essential to create a globally accurate representation. In industrial applications, the amount of data is limited due to time and cost constraints. Thus, sampling points should be selected to improve the surrogate model accuracy as much as possible. Typically, the best choice of samples is unknown beforehand, and different methods are developed to overcome this problem. One widely used approach is Latin hypercube sampling (LHS) [22] where a DoE is created in advance based on sampling from the partitioned input space. Different modifications were proposed to reduce the correlation between the simulated variables and to ensure a space-filling design, e.g. based around singular-value decomposition [23] or simulated annealing [24]. This approach can be referred to as one-shot sampling since the DoE is created at once. In contrast, *adaptive* sampling techniques try to select new points sequentially. This allows using knowledge from previous iterations represented by the existing surrogate model to guide the adaptive sampling process. In the context of optimization using probabilistic models, this is also referred to as Bayesian optimization (BO) [25,26]. However, this work focuses on the creation of *globally accurate* surrogate models within the design domain, i.e. the space of interest in which the surrogate model will be eventually used, based on adaptive sampling strategies. For this objective, an auxiliary function called acquisition is optimized to identify new experiments one by one. A popular choice for the acquisition function is the variance of the predictive distribution of the surrogate model, which is known within the context of Kriging [27] as Maximum Mean Square Error (MMSE) [21]. New points would then be obtained by selecting the sample with the highest predicted uncertainty. However, an effective adaptive sampling approach should consider two goals simultaneously, as pointed out in [28,29]:

Global Exploration aims to discover the whole design domain and selects samples e.g. with maximum distance to each other.

Local Exploitation selects samples at strategic points, e.g. in regions with large prediction errors, that are important to capture the full function behavior.

Various methods have been proposed, that extend the idea of using the predictive variance for global exploration and encourage local exploitation based on: leave-one-out cross validation (LOOCV) error of the model [30,31], squared response difference [32] or gradient information [33]. [34] combined an adaptive weighted exploration based on triangulation with LOOCV based exploration. Other approaches like the Taylor-Expansion based Adaptive Design (TEAD) [35] combine gradient information with distance based exploration. Another approach is to use the variance among a committee of models to formulate the exploitation criterion [36]. See [37,38] for a comprehensive list.

Related to the investigated strategies are adaptive sampling methods proposed for solving PDEs in the context of physics-informed machine learning [1], especially for Physics-Informed Neural Networks [39–41]. These methods improve the adaptive sampling procedure by minimizing the variance of the residuals between the surrogate approximation and the numerical solution. Those adaptive methods need additional evaluations of the PDE to refine the dataset used for training the surrogate model and require access to some information about the PDEs describing the problem, in contrast to the strategies considered in this study. An overview is given in [39].

Sample efficiency of the different adaptive strategies may vary depending on the response surface, i.e. some methods may perform better at approximating specific response surfaces and worse at others. Selecting the best performing sampling method for the task at hand from the multitude of available algorithms is therefore challenging, since their performance is unpredictable a priori. Hence, a large scale comparative study is necessary to investigate their differences. So far, comparative studies only consider a subset of the algorithms used in this work and mostly for fewer benchmarks [38,42,43]; but for most adaptive sampling methods, the only empirical evaluation is offered by the original work introducing the method (see e.g. [31,33,34]). Therefore, this paper compares some of the recently developed adaptive sampling strategies for global surrogate modeling, based on 1 to 8-dimensional benchmark functions.[2] In addition to the study, a novel acquisition function inspired by Expected Improvement for Global Fit (EIGF) [32] and TEAD [35] is presented, which is called Gradient and Uncertainty Enhanced Sequential Sampling (GUESS). In our study we show, that GUESS provides promising results and an improvement regarding the sample efficiency over the inspired strategies based on our experiments.

The paper is structured as follows: Section 2 starts with the theoretical background, introducing the deployed ML algorithms for surrogate modeling. Section 3 introduces the investigated adaptive sampling strategies. In Section 4, the results of the benchmark study are presented. Finally, in Section 5, concluding remarks are given. Appendices contain complementary results for higher-dimensional problems (Appendix A.1), as well as an ablation study regarding the model choice (Appendix A.2), computational effort (Appendix A.3), implementation details (Appendix B), and the used benchmark functions (Appendix C).

2. Theoretical background

2.1. Surrogate modeling

For a response function $f : \mathbb{R}^n \to \mathbb{R}; \mathbf{x} \mapsto f(\mathbf{x}) = y$ of a simulation model with input $\mathbf{x}$ and output y, the surrogate model $\hat{f}$ approximates the relationship $\hat{f}(\mathbf{x}^*; \mathcal{D}) \approx y^*$ based on the training data $\mathcal{D} = \{(\mathbf{x}_i, y_i) | 1 \leq i \leq m, i \in \mathbb{N}\}$, where $\mathbf{x}_i$ is the i-th row in $\mathbf{X}_{i,:}$, $\mathbf{X} \in \mathbb{R}^{m \times n}$, $\mathbf{y} \in \mathbb{R}^m$ with m and n denoting the number of samples and the number of input parameters, respectively. The indices $(i, :)$ represent the entry in the i-th row of the sample matrix. Testing quantities ($\mathbf{x}^* \notin \mathcal{D}$) are indicated with the asterisk (*) superscript.

Four probabilistic surrogate models are considered in this work: GP, Sparse Variational Gaussian Process Regression (SVGP) [10], DGCN and an ensemble of Probabilistic Neural Networks (PNNs) [18–20]. The prediction of a probabilistic model is a random variable $\hat{f}(\mathbf{x}; \mathcal{D}) = Y \sim p(\tilde{y} | \mathbf{x}, \mathcal{D})$ and the value $\tilde{y}$ used for the further analysis is often the mean μ_Y or a sample from Y. Later, the variance of Y can be utilized to guide the adaptive sampling process to regions where the model has high uncertainty about the possible outcome. In the following a short description of GPs is given. SVGP is introduced in Appendix A.1.2, while DGCN and PNN, together with the results, can be found in Appendix A.2.

2.2. Gaussian process

A GP is a distribution over functions, which can be represented as a collection of infinite number of random variables, a finite number of which follow a joint Gaussian distribution. The choice of kernel function influences the prediction capability of the GP. A common correlation function is the Matérn kernel [44,45] which determines the covariance for a pair of points $\mathbf{x}$ and $\mathbf{x}'$:

$$k(\mathbf{x}, \mathbf{x}', l) = \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu} r}{l} \right)^{\nu} K_\nu \left(\frac{\sqrt{2\nu} r}{l} \right) \sigma_f^2 \tag{1}$$

where $r = \|\mathbf{x} - \mathbf{x}'\|_2$ is the Euclidean distance, $K_\nu(\cdot)$ is a modified Bessel function of the second kind [46], $\Gamma(\cdot)$ is the gamma function, σ_f^2 is the kernel variance and l is the learnable length scale. ν is a positive parameter with popular values 3/2 or 5/2. To train the model parameters $\boldsymbol{\theta}_\mathrm{GP} = \{l, \sigma_\mathrm{n}^2, \sigma_\mathrm{f}^2\}$ we can maximize the marginal log-likelihood

$$\log p(\mathbf{y} | \mathbf{X}, \boldsymbol{\theta}_\mathrm{GP}) = -\frac{m}{2} \log(2\pi) - \frac{1}{2} \log |\mathbf{K}_\mathrm{N}| - \frac{1}{2} \mathbf{y}^T \mathbf{K}_\mathrm{N}^{-1} \mathbf{y} \tag{2}$$

where σ_n^2 is the noise variance and $\mathbf{K}_\mathrm{N} = \mathbf{K} + \sigma_\mathrm{n}^2 \mathbf{I}$ is the covariance matrix, with $\mathbf{K}_{i,:} = [k(\mathbf{x}_i, \mathbf{x}_1, l), \ldots, k(\mathbf{x}_i, \mathbf{x}_m, l)]$ and $\mathbf{I}$ as the identity matrix. In contrast to the maximum-likelihood estimate, it is also possible to use a Bayesian

[2] Source code: https://github.com/SvenL13/GALE.

approach for parameter estimation (see e.g. [47]). Since the GP is defined as a joint Gaussian distribution, the prediction at a point $\mathbf{x}^*$ is a Gaussian variable Y^* with mean μ_{Y^*} and variance $\sigma^2_{Y^*}$:

$$\hat{f}_{\text{GP}}(\mathbf{x}^*; \boldsymbol{\theta}_{\text{GP}}) = \mu_{Y^*} = \mathbf{k}^T \mathbf{K}_{\text{N}}^{-1} \mathbf{y} \tag{3}$$

$$\mathbb{V}\left[\hat{f}_{\text{GP}}(\mathbf{x}^*; \boldsymbol{\theta}_{\text{GP}})\right] = \sigma^2_{Y^*} = k(\mathbf{x}^*, \mathbf{x}^*, l) - \mathbf{k}^T \mathbf{K}_{\text{N}}^{-1} \mathbf{k} \tag{4}$$

where $\mathbf{k}$ denotes the vector of correlations between $\mathbf{x}^*$ and the design points, $\mathbf{k} = [k(\mathbf{x}^*, \mathbf{x}_1, l), \ldots, k(\mathbf{x}^*, \mathbf{x}_m, l)]^T$.

2.3. Leave-one-out cross validation

The k-fold cross validation [48] can be used to estimate the surrogate model accuracy on unseen samples. In the special case for m folds, we obtain the LOOCV, where m observations are made. For every $i \in [1, m]$, a model is trained on the $m - 1$ observations in order from the reduced set $\mathcal{D}_{\neg,i} = \mathcal{D} \backslash \{\mathbf{x}_i, y_i\}$. Then the LOOCV error at a location $\mathbf{x}_i$ can be calculated with the squared error loss as

$$e^2_{\text{LOO}}\left(\mathbf{x}_i; \hat{f}, \mathcal{D}\right) = \left(\tilde{y}_i - \tilde{y}_{\neg,i}\right)^2, \quad \forall i \in [1, m] \tag{5}$$

where $\tilde{y}_i$ is the prediction at $\mathbf{x}_i$ from the model trained on the whole dataset $\mathcal{D}$ and $\tilde{y}_{\neg,i}$ is the prediction from the model trained on the reduced set $\mathcal{D}_{\neg,i}$.

2.4. Fast approximation for GP

Given a set of training samples $\mathcal{D}$ and a GP with mean function $\mu(x) = 0$ and kernel function $k(\cdot)$, we can derive a closed-form solution for the approximated LOOCV error [5,49]

$$\hat{e}_{\text{LOO}}(\mathbf{x}_i) = \frac{\boldsymbol{\Lambda}_{i,:}\mathbf{y}}{\boldsymbol{\Lambda}_{ii}}$$

where $\boldsymbol{\Lambda}$ is the precision matrix $\boldsymbol{\Lambda} = \mathbf{K}^{-1}$ and the subscript ii denotes the entry in the i-th row and i-th column of the matrix. With the fast approximation of e_{LOO}, the computational complexity can be reduced from $\mathcal{O}(m^4)$ to $\mathcal{O}(m^3)$, since the inverse of $\mathbf{K}$ is calculated only once, plus $\mathcal{O}(m^2)$ for the whole LOOCV procedure [5].

3. Adaptive sampling strategies for global fit

Considering an expensive black-box function $f : \mathrm{X} \to \mathrm{Y}$ with output $y \in \mathrm{Y} \subseteq \mathbb{R}$, where the objective is to create a surrogate model $\hat{f}$ that is globally accurate within the design domain $\mathrm{X} \subseteq \mathbb{R}^n$. The aim of adaptive sampling strategies is to form a sequential procedure to suggest new sample points for improving the surrogate model accuracy as much as possible.

A set of initial samples $\mathcal{D}_0 = \{(\mathbf{x}^0_i, y^0_i) | 1 \leq i \leq m_0, i \in \mathbb{N}\}$ is generated first using a space filling sampling approach (e.g. LHS), where $\mathbf{x}^0_i$ is the i-th row in $\mathbf{X}^0_{i,:}$, $\mathbf{X}^0 \in \mathrm{X}^{m_0}$, $\mathbf{y}^0 \in \mathrm{Y}^{m_0}$ with m_0 denoting the number of initial samples. The methods covered in this work employ an acquisition function $\phi : \mathrm{X} \to \mathbb{R}$ that is maximized in order to obtain a new sampling point in the t-th iteration

$$\mathbf{x}^t = \underset{\mathbf{x} \in \mathrm{X}}{\arg\max}\, \phi\left(\mathbf{x}; \mathcal{D}_t, \hat{f}_t\right) \tag{6}$$

where $\hat{f}_t$ is the trained surrogate model using the dataset $\mathcal{D}_t$. From here on, we drop writing the explicit dependence of ϕ on $\mathcal{D}_t$ and $\hat{f}_t$.

Some methods rely on selecting the candidate sample with the highest acquisition value over some candidate points $\mathbf{x}^c \in \mathbf{X}^c$ instead of finding the maximum using an optimization algorithm. The candidate set $\mathbf{X}^c$ is then obtained by sampling (e.g. random sampling, LHS, ...) from X. In this work, we have restricted our view to single-response problems. An overview of adaptive sampling for multi-response models is given in [37].

3.1. Adaptive sampling workflow

The adaptive sampling approach can be described in six steps:

Step 1: Create the initial design points $\mathbf{X}^0$. If needed, create the candidate set $\mathbf{X}^c$ with a sampling method (e.g. LHS) and set $t = 0$.
Step 2: Evaluate the expensive black-box function f at $\mathbf{X}^0$ to obtain the responses $\mathbf{y}^0$ and construct the initial dataset $\mathcal{D}_0$.
Step 3: Train the surrogate model $\hat{f}_t$ using $\mathcal{D}_t$.
Step 4: Maximize the acquisition function ϕ with respect to the design domain $\mathbb{X}$ or the candidate set $\mathbf{X}^c$ to identify a new design point $\mathbf{x}^t$ (Eq. (6)) using $\hat{f}_t$ and $\mathcal{D}_t$.
Step 5: Evaluate the expensive black-box function f at $\mathbf{x}^t$ to receive the response y^t. Extend $\mathcal{D}_t$ with the new data to obtain the dataset $\mathcal{D}_{t+1} = \mathcal{D}_t \bigcup \left\{\left(\mathbf{x}^t, y^t\right)\right\}$. Create a new candidate set $\mathbf{X}^c$ if needed.
Step 6: Examine if a stopping condition is met (e.g. maximum iterations, time constraint, accuracy goal). Otherwise increment t and go to Step 3.

3.2. Acquisition functions

Acquisition functions are constructed to guide the adaptive sampling process in regions that are most beneficial to generate new data. Previous research indicates that the most effective methods use a combination of exploration and exploitation strategies [38]. Additional adjustment factors may also be used to weigh between exploration and exploitation based on past observations. In the following, different methods are presented based on these criteria.

MMSE: The Maximum Mean Square Error (MMSE) introduced for Kriging by Sacks and Schiller [21] is a straightforward approach that is based on the predictive posterior variance of the surrogate model and is therefore purely exploration based

$$\phi_{\mathrm{MM}}(\mathbf{x}) = \sigma_Y^2$$

wMMSE: The Weighted Maximum Mean Square Error (wMMSE) [31] is an extension to the MMSE that considers an additional exploitation criterion. The trade-off between exploration and exploitation is adjusted with an additional factor $\alpha_{\mathrm{WM}} \in \mathbb{R}_+$, where $\alpha_{\mathrm{WM}} > 1$ favors exploration and $\alpha_{\mathrm{WM}} < 1$ exploitation. On average, the authors provided empirical evidence using benchmark functions that $\alpha_{\mathrm{WM}} = 1$ yields the best performance. The acquisition function is defined as

$$\phi_{\mathrm{WM}}(\mathbf{x}) = (\gamma(\mathbf{x}))^{\alpha_{\mathrm{WM}}}\, \sigma_Y^2$$

where $\gamma(\cdot)$ contains information about the surrogate model error at the closest sample within the observed design points $\mathbf{x}_o \in \mathbf{X}^t \in \mathbb{R}^{m_t \times n}$

$$\gamma(\mathbf{x}) := e^2_{\mathrm{LOO}}(\mathbf{x}_o), \ \ \mathbf{x} \in \mathcal{V}_o$$

where $\mathcal{V}_o$ is the Voronoi cell assigned to $\mathbf{x}_o$

$$\mathcal{V}_o = \left\{\mathbf{x} \in \mathbb{X} \middle| \|\mathbf{x} - \mathbf{x}_o\|_2 \leq \|\mathbf{x} - \mathbf{x}_j\|_2, \forall j \neq o\right\}$$

where $o, j \in \{1, \dots, m_t\}$ and m_t is the number of observed samples in the t-th iteration. Therefore, the Voronoi partitioning is used to make e^2_{LOO} available over the domain $\mathbb{X}$ according to the observed samples $\mathbf{x}_o$.

MEPE: The Maximizing Expected Prediction Error (MEPE) [30] uses the cross-validation error e^2_{LOO} for local exploitation and the prediction variance σ_Y^2 for exploration. The acquisition function is defined as

$$\phi_{\mathrm{MEP}}(\mathbf{x}) = \alpha_{\mathrm{MEP}}\gamma(\mathbf{x}; \mathcal{D}_t) + (1 - \alpha_{\mathrm{MEP}})\sigma_Y^2$$

where the balance factor $\alpha_{\mathrm{MEP}} \in \mathbb{R}^{[0,1)}$ weighs exploration and exploitation adaptively, depending on the true and cross-validation error at the latest observed point

$$\alpha_{\mathrm{MEP}} := \begin{cases} 0.5, & t = 0, \\ 0.99 \min\left(0.5 \dfrac{e^2_{\mathrm{true}}\left(\mathbf{x}^{t-1}\right)}{\hat{e}^2_{\mathrm{LOO}}\left(\mathbf{x}^{t-1}\right)}, 1\right), & t > 0 \end{cases}$$

where the true error e^2_{true} is obtained from f and $\hat{f}_{t-1}$ at the previous observed location $\mathbf{x}^{t-1}$.

EIGF: Inspired by the Expected Improvement criterion for global optimization [26], the Expected Improvement for Global Fit (EIGF) [32] is adapted to combine information from the predicted variance together with the squared response difference as

$$\phi_{\mathrm{EI}}(\mathbf{x}) = \left(\hat{f}_t(\mathbf{x}) - f(\mathbf{x}_o)\right)^2 + \sigma_Y^2, \ \mathbf{x} \in \mathcal{V}_o \tag{7}$$

GGESS: Chen et al. [33] modified the EIGF criterion and proposed the Gradient and Geometry Enhanced Sequential Sampling (GGESS), by including additional gradient information to improve the acquisition function

$$\phi_{\mathrm{GG}}(\mathbf{x}) = \left(f(\mathbf{x}_o) - \hat{f}_t(\mathbf{x}) - \nabla_{\mathbf{x}}\hat{f}_t(\mathbf{x})^T(\mathbf{x}_o - \mathbf{x})\right)^2 + \sigma_Y^2, \ \mathbf{x} \in \mathcal{V}_o \tag{8}$$

where $\nabla_{\mathbf{x}}\hat{f}_t(\mathbf{x})$ is the approximate gradient vector of the true function f at $\mathbf{x}$.

TEAD: Similar to GGESS, Taylor-Expansion based Adaptive Design (TEAD) [35] uses the gradient vector $\nabla_{\mathbf{x}}\hat{f}_t(\mathbf{x})$ to find potential samples in regions of high interest. The exploitation criterion is constructed around a Taylor-based approximation of the second- and higher-order Taylor expansion values

$$\delta(\mathbf{x}) = \left|\hat{f}_t(\mathbf{x}) - \hat{f}_t(\mathbf{x}_o) - \nabla_{\mathbf{x}}\hat{f}_t(\mathbf{x}_o)^T(\mathbf{x} - \mathbf{x}_o)\right|, \ \mathbf{x} \in \mathcal{V}_o \tag{9}$$

with $\nabla_{\mathbf{x}}\hat{f}_t(\mathbf{x}_o)^T(\mathbf{x} - \mathbf{x}_o)$ as the first-order Taylor expansion of the model at $\mathbf{x}_o$. $\nabla_{\mathbf{x}} f$ is replaced with the faster approximation $\nabla_{\mathbf{x}}\hat{f}$. The exploration is based on the closest distance between a point $\mathbf{x}$ and the observed data points $\mathbf{X}^t$

$$d_{\min}(\mathbf{x}) = \min_{\mathbf{x}_i \in \mathbf{X}^t} \|\mathbf{x} - \mathbf{x}_i\|_2 \tag{10}$$

The acquisition function for TEAD is the combination of both criteria

$$\phi_{\mathrm{TE}}(\mathbf{x}) = \frac{d_{\min}(\mathbf{x})}{\max_{\mathbf{x}^c \in \mathbf{X}^c} d_{\min}(\mathbf{x}^c)} + \alpha_{\mathrm{TE}}(\mathbf{x}) \frac{\delta(\mathbf{x})}{\max_{\mathbf{x}^c \in \mathbf{X}^c} \delta(\mathbf{x}^c)}$$

where $\alpha_{\mathrm{TE}} \in \mathbb{R}^{[0,1]}$ is an additional adjustment factor

$$\alpha_{\mathrm{TE}}(\mathbf{x}) := 1 - d_{\min}(\mathbf{x})/d_{\max}$$

with $d_{\max}$ as the maximum distance of any two points in X.

MASA: The Mixed Adaptive Sampling Algorithm (MASA) [36] uses a committee of different models $\mathcal{M} = \{\hat{f}^{\mathcal{M}}_1, \ldots, \hat{f}^{\mathcal{M}}_{n_{\mathcal{M}}}\}$ to construct an exploitation criterion based on the variance among the $n_{\mathcal{M}}$ committee members

$$F_{\mathrm{QBC}}(\mathbf{x}; \mathcal{M}) = \frac{1}{n_{\mathcal{M}}} \sum_{i=1}^{n_{\mathcal{M}}} \left(\hat{f}^{\mathcal{M}}_i(\mathbf{x}) - \overline{\hat{f}^{\mathcal{M}}}(\mathbf{x})\right)^2$$

where $\overline{\hat{f}^{\mathcal{M}}}(\mathbf{x}) = \frac{1}{n_{\mathcal{M}}} \sum_{i=1}^{n_{\mathcal{M}}} \hat{f}^{\mathcal{M}}_i(\mathbf{x})$ is the average of the committee prediction. Similar to TEAD, the exploration is based on the minimal distance between the observed and candidate points (Eq. (10)). The acquisition function is given as

$$\phi_{\mathrm{MA}}(\mathbf{x}; \mathcal{M}) = \frac{F_{\mathrm{QBC}}(\mathbf{x}; \mathcal{M})}{\max_{\mathbf{x}^c \in \mathbf{X}^c} F_{\mathrm{QBC}}(\mathbf{x}^c; \mathcal{M})} + \frac{d_{\min}(\mathbf{x})}{\max_{\mathbf{x}^c \in \mathbf{X}^c} d_{\min}(\mathbf{x}^c)}$$

DL-ASED: The Discrepancy Criterion and Leave One Out Error-based Adaptive Sequential Experiment Design (DL-ASED) [34] uses a weighted combination of distance-based exploration together with LOOCV-based exploration. The notion of support points is introduced for the exploration criterion. Therefore, $n+1$ support points $\mathbf{X}^s = \left[\mathbf{x}^s_0, \ldots, \mathbf{x}^s_{n+1}\right]^T$, with $\mathbf{x}^s_i \in \mathbb{X}^t$, are assigned to each candidate $\mathbf{x}^c$, based on a triangulation scheme (see [34]).

The exploration criterion is calculated as

$$g(\mathbf{x}) = v_s^p \prod_{i=1}^{n+1} (\|\mathbf{x} - \mathbf{x}^s_i\|_2 \cdot \|\hat{f}_t(\mathbf{x}) - f(\mathbf{x}^s_i)\|_2) \tag{11}$$

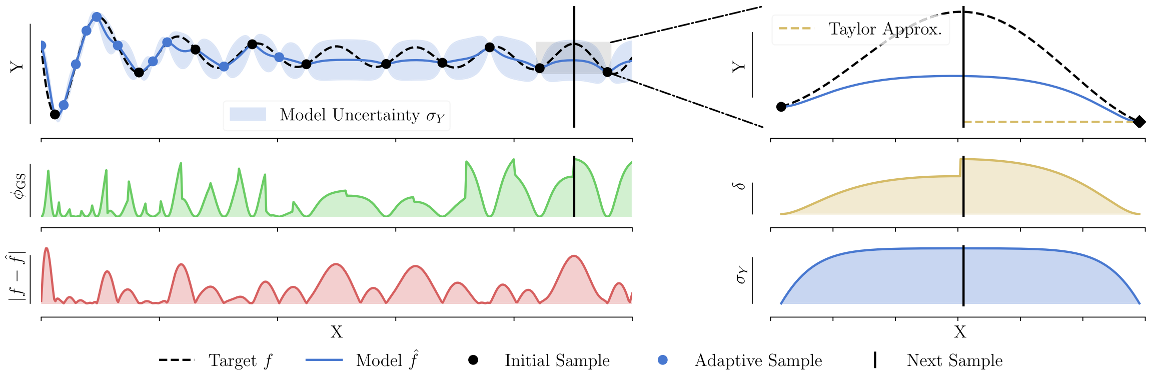

Fig. 1. Illustration of GUESS with GP surrogate model and Matérn kernel ($\nu = 3/2$) for the GramLee function. The acquisition function ϕ_{GS} (🟩) is decomposed into second and higher-order reminders δ (🟨) and predicted standard deviation σ_Y (🟦). Plot in the top right corner shows the fist-order Taylor approximation at the closest observed point $\mathbf{x}_o$ (◆). δ is large in areas with high non-linearity, defined as the difference between surrogate prediction $\hat{f}$ and first order approximation. The step in δ is due to the switch of expansion point, i.e. another observed sample $\mathbf{x}_o$ is closer. As reference, the absolute error between the underlying function and surrogate model is visualized (🟥). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

where $\mathbf{x}_i^s$ is a local support point around $\mathbf{x}$, v_s is the volume of the polyhedron constructed with the support points $\mathbf{X}^s$ and p is a coefficient that is not further described in [34]. In the following $p = 1$ is used. For the local exploitation based on e_{LOO}, an additional model $\hat{f}_{e_{\text{LOO}}} : \mathbb{X} \to \mathbb{R}$ is used to define a map over the entire $\mathbb{X}$. The model approximates the relationship $\hat{f}_{e_{\text{LOO}},t}(\mathbf{x};\, \mathcal{D}_t) \approx e_{\text{LOO},t}$ using the data set $\left\{(\mathbf{x}_i, e_{\text{LOO}}(\mathbf{x}_i)) \;\middle|\; \mathbf{x}_i \in \mathbf{X}^t \text{ for } i = 0, \ldots, m_t\right\}$. The acquisition function is then defined as

$$\phi_{\text{DL}}(\mathbf{x}) = (1 - \alpha_{\text{DL}}) \frac{g(\mathbf{x})}{\max_{\mathbf{x}^c \in \mathbf{X}^c} g(\mathbf{x}^c)} + \alpha_{\text{DL}} \frac{\hat{f}_{e_{\text{LOO}},t}(\mathbf{x};\, \mathcal{D}_t)}{\max_{\mathbf{x}^c \in \mathbf{X}^c} \hat{f}_{e_{\text{LOO}},t}(\mathbf{x}^c;\, \mathcal{D}_t)}, \tag{12}$$

where α_{DL} is the adaptive weight in the t-th iteration

$$\alpha_{\text{DL}} := \begin{cases} 0.5, & t = 0, \\ \min\left(0.5 \dfrac{\zeta_{\text{local}}}{\zeta_{\text{global}}}, 1\right), & t > 0 \end{cases}$$

with $\zeta_{\text{global}} = \sum_{i=1}^{m_t} e^2_{\text{LOO},t}(\mathbf{x}_i) \,/ \sum_{j=1}^{m_{t-1}} e^2_{\text{LOO},t-1}(\mathbf{x}_j)$ as the global and $\zeta_{\text{local}} = e^2_{\text{LOO},t}\left(\mathbf{x}^t\right) / e^2_{\text{LOO},t-1}\left(\mathbf{x}^t\right)$ as the local improvement of e_{LOO} in the t-th iteration. $e^2_{\text{LOO},t-1}$ is the LOOCV error based on the previous model $\hat{f}_{t-1}$ and dataset $\mathcal{D}_{t-1}$.

GUESS: We propose a novel criterion, the Gradient and Uncertainty Enhanced Sequential Sampling (GUESS). The acquisition uses the predicted standard deviation for exploration of the design domain and a Taylor expansion based approximation of the second- and higher-order reminders for exploitation. The exploitation term is weighted by the predicted standard deviation, i.e. it acts as a penalty to avoid choosing samples too close to already observed samples. The acquisition function is given by

$$\phi_{\text{GS}}(\mathbf{x}) = (\delta(\mathbf{x}) + 1)\, \sigma_Y, \;\; \mathbf{x} \in \mathcal{V}_o \tag{13}$$

where $\delta(\mathbf{x})$ is the Taylor-based approximation of the second- and higher-order Taylor expansion values (Eq. (9)). Fig. 1 illustrates different components of GUESS. We perform GP using a Matérn kernel ($\nu = 3/2$) on a toy dataset, where 10 points are sampled with LHS (●) and 10 points are sampled with GUESS (🔵) from the GramLee function. As can be seen in Fig. 1, the second- and higher-order Taylor expansion values in Eq. (9) act as a measure of non-linearity of the function (🟨). The approximation accuracy of the first-order Taylor expansion decreases as the non-linearity of f increases as well as the distance to the expansion point increases [35]. In most scenarios, ϕ_{GS} is dominated by $\delta(\mathbf{x})\sigma_Y$. However, in situations with small gradients, the addition of σ_Y acts as a fallback to locate samples in regions with large model uncertainty. Modifying Eq. (13) by replacing $\delta(\mathbf{x})$ with $\delta(\mathbf{x})^{\alpha_{\text{GS}}}$, where $\alpha_{\text{GS}} \in \mathbb{R}_+$, controls the trade-off between exploration ($\alpha_{\text{GS}} < 1$) and exploitation ($\alpha_{\text{GS}} > 1$). In the following, we

Table 1
Classification of the investigated sampling strategies.

Exploitation	Exploration	
	distance-based	predictive variance-based
no exploitation	LHS	MMSE
LOOCV-based	DL-ASED[a]	MEPE wMMSE[a]
committee-based	MASA[a]	–
geometry-based	TEAD[a]	EIGF GUESS[a] GGESS[a]

[a] Discontinuous acquisition function based on $\mathbf{X}^c$.

considered GUESS only with $\alpha_{\mathrm{GS}} = 1$ which gives a unscaled trade-off, i.e. model uncertainty of the model will decrease with increasing sample size.

3.3. Classification of sampling strategies

Table 1 shows the classification based on the exploration and exploitation components for the different sampling strategies. Most of the presented acquisition functions are originally formulated for Kriging but a majority of them can be adapted without change to other probabilistic models like DGCNs or PNNs. However, some of the methods (MEPE, DL-ASED and wMMSE) rely on calculating the LOOCV error, which can be a computationally demanding task for other surrogate models without a fast approximation available like described in Section 2.4.

4. Benchmark study

A benchmark study using analytical functions was conducted to compare the sampling methods described before. The results are presented in the following. Besides the adaptive strategies, one-shot sampling using LHS was also investigated as a baseline.

4.1. Test scheme

The evaluation was based on 15 different deterministic benchmark functions chosen from optimization literature (see Appendix C). The benchmark functions range from 1 to 8 dimensions and input dimensions were scaled to have unit length. The functions were selected to cover a wide palette of various surface structures (multiple local optima, valley shaped, bowl shaped, etc.) which propose different challenges that could occur in engineering problems.

Initial samples, optimization and other aspects introduce stochasticity to the sampling process. Moreover, assessing the performance of an adaptive sampling strategy on a single run may be misleading due to randomness. Therefore, every benchmark function was repeated 10 times for each method and fixed seeds were used for each run to ensure reproducibility and similar numerical conditions between the evaluated methods.

The acquisition functions in Section 3.2 were maximized in each iteration to propose new points. Some of the investigated methods rely on finding the maximum over a finite set of candidate points $\mathbf{X}^c$ as explained in Section 3. A $(\mu+1)$ evolutionary based algorithm [50] was used to identify the next sampling point for the remaining methods (see Table 1). Further, LHS was used to create the initial $\mathbf{X}^0$, candidate $\mathbf{X}^c$ and test samples $\mathbf{X}^e$. The number of initial samples $m_0 = 10n$ was used as proposed in [51]. The same initial and test samples were used for each method in a repetition. The maximum number of samples $m_{\max}$ was used as stopping criterion. The experimental settings including the dimensions n, numbers of initial samples m_0, candidate samples m_{cand}, testing samples m_{test} and maximum samples $m_{\max}$ are shown in Table 2.

Emphasis has been placed on comparing adaptive schemes with regard to different aspects such as the maximum achieved accuracy after $m_{\max}$ samples, sample-efficiency and variance of performance. The coefficient of determination R^2 [52] is used as a metric to evaluate the model accuracy. It gives an upper bounded metric and

Table 2
Experimental settings depending on the input dimension n of the benchmark functions. m_0, $m_{\max}$, m_{cand} and m_{test} represent the initial sample, maximum sample, candidate sample and test sample sizes in this order.

n	m_0	$m_{\max}$	m_{cand}	m_{test}
1	10	40	5000	100000
2	20	140	10000	100000
3	30	180	15000	100000
4	40	250	20000	100000
6	60	250	30000	100000
8	80	250	40000	100000

is preferred for regression tasks compared to other metrics (mean squared error, mean absolute error, etc.) [53]. R^2 can be computed as

$$R^2(\mathbf{y}, \tilde{\mathbf{y}}) = 1 - \frac{\sum_{i=1}^{m}(y_i - \tilde{y}_i)^2}{\sum_{i=1}^{m}(y_i - \bar{y}_i)^2},$$
$$\text{where } \bar{y} = \frac{1}{m}\sum_{i=1}^{m} y_i.$$

The surrogate model $\hat{f}_t$ was evaluated in each iteration t to compare different adaptive sampling methods. R^2 score was calculated from the true response for the test dataset $\mathbf{y}^e$ and the surrogate prediction $\tilde{\mathbf{y}}_t^e = [\hat{f}_t(\mathbf{x}_1^e), \dots, \hat{f}_t(\mathbf{x}_{m_{\text{test}}}^e)]^T$, with $\mathbf{x}^e \in \mathbf{X}^e$ and $\hat{f}_t$ is the surrogate model trained in the t-th iteration from $\mathcal{D}_t$.

Additionally, we propose $R^2_{\text{Area}}: \mathbb{R}^{s+1} \to \mathbb{R}^{[0,1]}$ as a novel metric to measure the overall performance and sample-efficiency of the adaptive methods over $s = (m_{\max} - m_0 - 1)$ intervals. R^2_{Area} can be computed as

$$R^2_{\text{Area}}(\mathbf{r}^2) = \frac{f_{\text{s}}(\mathbf{r}^2) + f_{\text{tr}}(\mathbf{r}^2)}{s} \tag{14}$$

where f_{s} is the composite Simpson rule [54]

$$f_{\text{s}}(\mathbf{r}^2) = \frac{1}{3}\left(r^2_{\xi(s)} + 2\sum_{k=1}^{s'-1} r^2_{2k+\xi(s)} + r_s^2 + 4\sum_{k=1}^{s'} r^2_{2k-1+\xi(s)} \right)$$

with $s' = (s - \xi(s))/2$, $\mathbf{r}^2 = \left[r_0^2, \dots, r_s^2\right]^T$, and $r_i^2 = \max\left(0, R^2(\mathbf{y}^e, \tilde{\mathbf{y}}_i^e)\right)$. The parity indicator function $\xi(\cdot)$ is defined as

$$\xi(s) := \begin{cases} 0, & \text{if } (-1)^s = 1, \\ 1, & \text{else.} \end{cases}$$

If s is odd, then the trapezoidal rule

$$f_{\text{tr}}(\mathbf{r}^2) = \frac{r_0^2 + r_1^2}{2}\xi(s)$$

is applied to the first interval. R^2_{Area} allows quantifying the entire sampling history into an easy to interpret value.

As surrogate model, GP($\mu(\mathbf{x}), k(\mathbf{x}, \mathbf{x}', l)$) with zero mean function $\mu(\mathbf{x}) = 0$, $k(\cdot)$ as the Matérn kernel ($\nu = 3/2$), and $\sigma_f^2 = 1$ was used to generate the presented results in the following. The output y of the surrogate model were normalized during training and prediction: $y_{\text{norm}} = (y - y_\mu)/y_\sigma$, where y_μ and y_σ are the mean and standard deviation, in this order, calculated from the training data. The normalization was reversed for the estimation of the test metrics. The adjustment factor $\alpha_{\text{WM}} = 1$ was used for wMMSE. The committee in MASA consisted of five GPs with different covariance functions: squared exponential, two Matérn functions ($\nu = 3/2$ and $\nu = 5/2$), dot-product function and a rational quadratic function. A GP model with posterior mean function $\hat{f}_{e_{\text{LOO}}}$ with Matérn kernel ($\nu = 3/2$) was used to approximate LOOCV error in Eq. (12). Moreover, Delaunay triangulation [55] was used to select the support points in Eq. (11). The $m_{\text{edges}} = 2^n$ edges of the design space were sampled after the

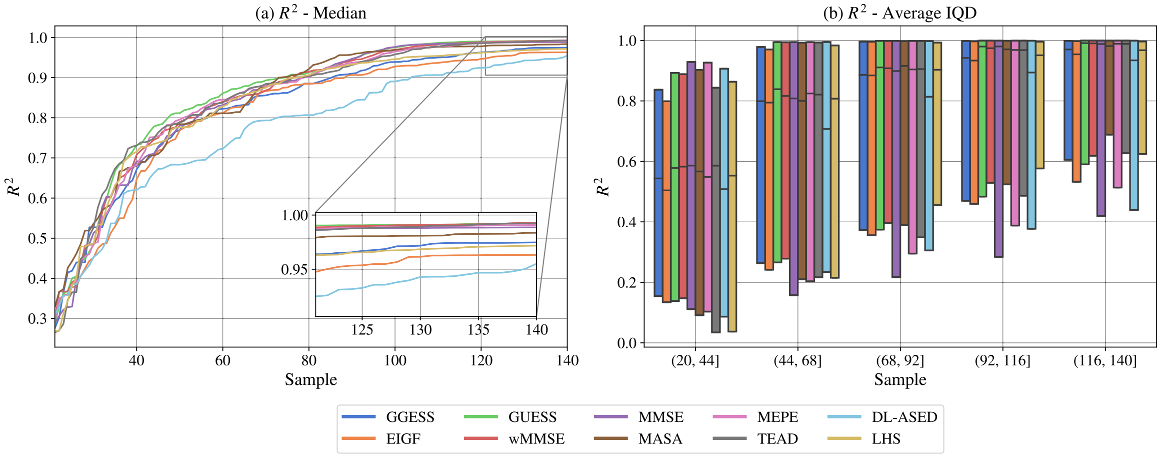

Fig. 2. Comparison of different adaptive sampling strategies with GP surrogate model across 10 repetitions for 2D benchmark functions: BekerLogan, Eggholder, Himmelblau, Branin, DropWave, Michalewicz2D and Schwefel2D (see Appendix C). (a) R^2 median over the conducted benchmarks and repetitions, (b) IQD of the average R^2 within each group; boxes indicate the 25% and 75% quantile and the horizontal line the median.

initial samples in order to calculate the exploration component in Eq. (11) for all $\mathbf{x} \in \mathrm{X}$ and to use DL-ASED. The acquisition function in Eq. (12) was maximized for the following samples. Further implementation details can be found in Appendix B.

4.2. Results for 2 - 4D benchmark functions

One figure with two different plots is presented for each input dimension of the benchmark functions. The left plot shows the R^2 score over the sample number, where the median of the curves resulting from 10 runs on each benchmark function is displayed for each strategy. Rolling max was performed, based on the assumption that the best model from previous iterations is stored and can be used in the case when no subsequent models could improve the accuracy. The interquartile distance (IQD) of the means is displayed for each group of samples on the right plot. The bin size of the grouped samples was selected to be equal for each of the five groups used throughout.

Fig. 2 presents the aggregated results for the 2D benchmark functions. It can be seen that the median model performance does not improve much more after 140 samples. For the Schwefel2D and Eggholder functions, the model accuracy did not converge to a value close to 1 within $m_{\max}$ samples due to the high non-linearity and multimodality of these responses. The variance decreased with increasing number of samples on average for all tested methods (Fig. 2b). Using wMMSE yielded the highest R^2 score, closely followed by GUESS, TEAD and MEPE. The model based on DL-ASED achieved the lowest median score. LHS and most adaptive strategies showed similar speed of accuracy improvement within the first 100 samples. Later, GUESS, wMMSE, MEPE, MMSE and TEAD overperformed LHS and reach on median a R^2 score close to 0.99, as opposed to $R^2 \approx 0.97$ for LHS. GUESS showed significant improvement over GGESS regarding the overall performance based on the R^2 score.

The 3D results are shown in Fig. 3, where it is visible that the methods improved not much more after 140 samples. All adaptive strategies except for EIGF and DL-ASED outperformed the baseline LHS in this set of functions. However, LHS showed good performance in the beginning until 100 samples. The purely exploration based MMSE also worked well with the 3D benchmark functions. Overall, GUESS reached the highest median R^2 score, while DL-ASED showed the worst median results.

Fig. 4 shows the 4D benchmark results. In contrast to the previously seen behavior in Figs. 2 and 3, DL-ASED showed a rapid performance increase between 75 and 125 samples compared to the other methods. This can be explained due to the sampling of the edges described in Section 4.1, since the Rosenbrock4D and StyblinskiTang4D functions have large variances at the edges of the design domain. Although beneficial for some of the investigated functions, the sampling of all edges can become a burden for higher-dimensional problems and requires exponentially more samples as given before (see e.g. Fig. A.6). MASA achieved the highest median R^2 score followed by wMMSE and TEAD for the set of 4D benchmark functions.

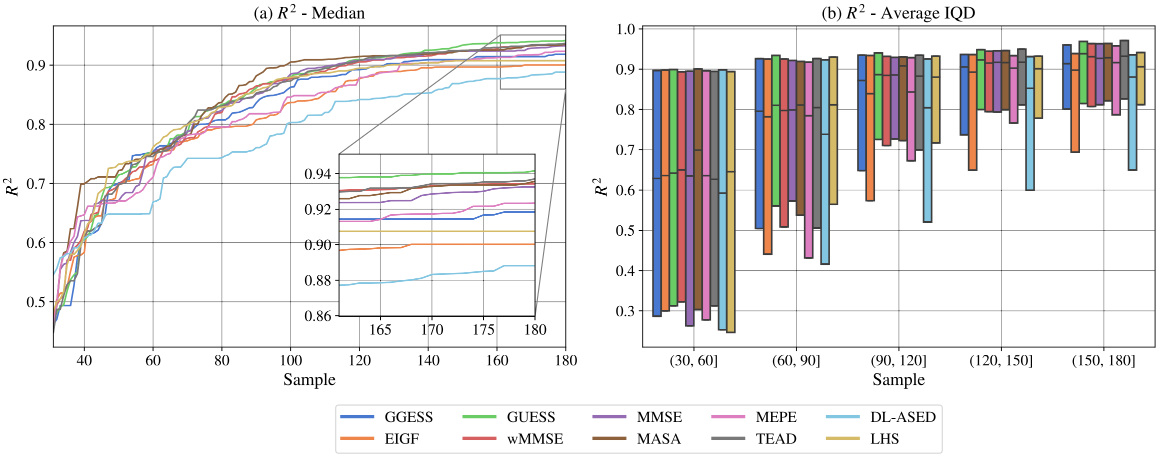

Fig. 3. Comparison of different adaptive sampling strategies with GP surrogate model across 10 repetitions for 3D benchmark functions: ACKLEY3D, ROSENBROCK3D, MICHALEWICZ3D and ISHIGAMI (see Appendix C). (a) R^2 median over the conducted benchmarks and repetitions, (b) IQD of the average R^2 within each group; boxes indicate the 25% and 75% quantile and the horizontal line the median.

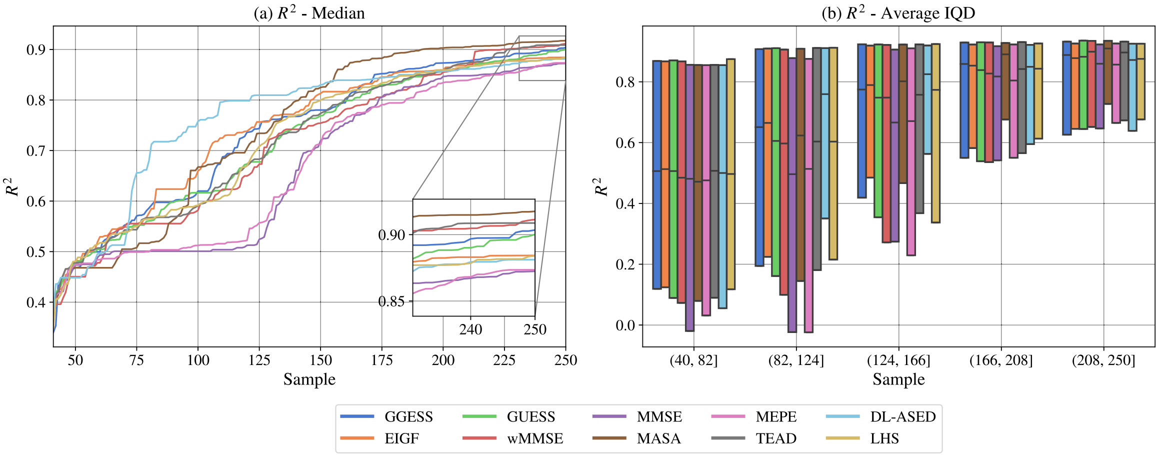

Fig. 4. Comparison of different adaptive sampling strategies with GP surrogate model across 10 repetitions for 4D benchmark functions: ACKLEY4D, ROSENBROCK4D, MICHALEWICZ4D and STYBLINSKITANG4D (see Appendix C). (a) R^2 median over the conducted benchmarks and repetitions, (b) IQD of the average R^2 within each group; boxes indicate the 25% and 75% quantile and the horizontal line the median.

4.3. Results for 1 - 8D benchmark functions

The adaptive sampling strategies were compared across all conducted benchmarks to estimate the overall performance. Results for 6 and 8 dimensions are given in Appendix A.1.1. The 1D benchmark functions with GRAMLEE, HUMPSINGLE and HUMPTWO are not explicitly displayed. Ranks 1 to 10 (smaller is better) were assigned based on the highest R^2 and R^2_{Area} score achieved in a run for each repetition. The mean rank and standard error of the mean are displayed for both scores in Table 3. Additionally, the mean and standard error for the R^2 and R^2_{Area} score were calculated based on the highest metric score in each repetition of the benchmark (Table 3). The results show that GUESS achieved the highest R^2_{Area} score with its respective rank, the second best average rank over all tested cases based on R^2, and the third highest score based on the average R^2. TEAD reached the highest R^2 score and rank, together with the second best R^2_{Area} rank on average. Furthermore, MASA achieved the second highest R^2 and R^2_{Area} on average. The competitive advantage of MASA is the access to multiple models with different covariance functions; as opposed to the other adaptive strategies which are restricted to a GP with Matérn kernel. Nevertheless, no single sampling strategy dominated its competitors across all functions. Moreover, it was observed

Table 3
Average R^2, R^2_{Area} and respective ranks together with the standard errors for all conducted benchmarks, 10 repetitions and GP surrogate model (including results from Appendix A.1.1). Average rank is calculated from the highest R^2 score achieved in a run for each repetition. Bold numbers represent the best, and underlined the second best result.

Method	R^2	R^2 SE	R^2_{Area}	R^2_{Area} SE	Rank R^2	Rank R^2 SE	Rank R^2_{Area}	Rank R^2_{Area} SE
GGESS	0.758	0.020	0.663	0.020	5.742	0.156	5.658	0.163
EIGF	0.744	0.020	0.657	0.020	6.762	0.147	6.369	0.157
GUESS	0.768	0.020	**0.675**	0.020	3.785	0.146	**4.065**	0.141
wMMSE	0.765	0.020	0.671	0.020	4.027	0.142	4.581	0.159
MMSE	0.733	0.020	0.638	0.021	6.665	0.176	6.973	0.177
MASA	0.771	0.019	0.674	0.020	4.988	0.178	4.473	0.153
MEPE	0.752	0.020	0.657	0.020	5.531	0.166	5.973	0.176
TEAD	**0.773**	0.019	0.672	0.020	**3.350**	**0.115**	4.269	**0.140**
DL-ASED	0.718	0.021	0.632	0.021	7.738	0.149	7.162	0.193
LHS	0.758	**0.019**	0.665	**0.020**	6.254	0.171	5.308	0.177

that nearly each investigated method (except for DL-ASED and EIGF) achieved the best position in at least one of the benchmark functions.[2] Which sampling strategy achieved the highest score overall was problem dependent. For example, MMSE achieved good results for 1 and 2-dimensional problems, while performing worse in higher dimensions compared to its competitors. In contrast, GGESS was found to perform better for higher-dimensional problems. We note that although TEAD achieved the highest R^2 score and the respective rank, GUESS provided overall the best performance with less iterations as indicated by R^2_{Area} and achieved on average R^2 scores close to the best-performing strategies. It was found that LHS overperformed nearly half of the adaptive strategies on average regarding R^2 and R^2_{Area}. Most of the strategies considering both exploration and exploitation showed on average improved performance over the purely exploration based MMSE.

5. Conclusion

This work proposed a novel adaptive sampling strategy called GUESS and compared it to some of the most recent adaptive sampling strategies to improve the global model accuracy within a unified framework using various benchmark functions. The proposed acquisition function guiding the sampling process leverages the predicted standard deviation of the surrogate model to both balance the exploitation term based on a approximation of the second and higher-order Taylor expansion values and explore new regions with high predictive variance. The proposed method can be used with any probabilistic model with a heteroscedastic variance estimate and was tested for single-response problems. A straightforward extension to handle multi-response problems could be to maximize a sum of the acquisition function values of each model output. Testing this approach is left for future research.

Due to the low number of comparisons, a large-scale comparative study is conducted to investigate the differences in recent developments. The presented methods were compared using a GP and 1 to 8-dimensional deterministic benchmark functions. GUESS achieved on average the highest sample efficiency regarding model accuracy compared to 9 other sampling strategies, and the second-highest accuracy overall, based on ranking all experiments. No single sampling strategy dominated its competitors across all functions. Nevertheless, the gradient-based methods GUESS and TEAD, as well as the committee-based method MASA, achieved the best performance within the conducted study on average.

Finally, our ablation study shows that the choice of surrogate model can have a great influence on the individual method performance and is therefore an important factor to consider. However, MASA, along with GUESS, provided the best results across all tested models on average. Additionally, we found using a more suitable surrogate model can have a greater impact on the achieved accuracy, and thus the sample efficiency compared to the choice of sampling strategy. In this context, a more suitable model is expected to achieve comparatively higher accuracy, when trained on the same data set as a less suitable model. However, the decision of the most suitable model for the task at hand remains a non-trivial question. Thus, further emphasis should be placed on exploring the influence of the surrogate model choice on adaptive sampling strategies. Moreover, determining the number of initial samples and selecting the stopping criteria are important yet open questions, as only a limited number of studies have addressed these issues.

CRediT authorship contribution statement

Sven Lämmle: Conceptualization, Methodology, Implementation, Writing – original draft. **Can Bogoclu:** Conceptualization, Methodology, Writing – review & editing. **Kevin Cremanns:** Methodology, Writing – review. **Dirk Roos:** Supervision, Writing – review.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The code used in this analysis are available at https://github.com/SvenL13/GALE.

Acknowledgment

This work was supported by ZF Friedrichshafen AG.

Appendix A. Complementary results

A.1. Influence of increasing dimensionality on method performance

We present the results regarding influence of increasing dimensionality on performance of adaptive sampling strategies in the following. Therefore, we compare the methods for 6 and 8-dimensional benchmark functions and study the behavior of GUESS up to 32 dimensions. Furthermore, we discuss challenges and approaches for adaptive sampling to overcome the curse of dimensionality.

A.1.1. Results for 6 - 8D benchmark functions

6 and 8-dimensional benchmarks were conducted to study the influence of increasing dimensionality with constant sample size. Results of the 6D benchmark are shown in Fig. A.5. TEAD achieved the highest accuracy closely followed by wMMSE and GUESS. MMSE underperformed compared to other algorithms. 8D benchmarks results are depicted in Fig. A.6. It can be seen that gradient-based strategies outperformed most non-gradient based methods in the later half and achieved the highest median R^2 score in comparison. Nevertheless, LHS showed comparable performance to all methods and even an overall improvement over non-gradient based strategies regarding the median R^2. This is not very surprising, since the initial small surrogate model accuracy could lead to erroneous information for the acquisition functions and, thus, misguide the sampling procedure. MEPE, MMSE and DL-ASED did not achieve an improvement after 250 samples compared to the initial data set. The exhaustive sampling of 2^8 edges could be the reason for DL-ASED, which lead to not using the acquisition function. Additionally, it could be observed that the variance increased for most methods with newly added samples (A.6b). This contradicts the observed behavior in Section 4.2, where the variance decreased with increasing number of samples. One explanation could be the faster improvement of accuracy for Ackley8D and Rosenbrock8D functions relative to Michalewicz8D and StyblinskiTang8D, leading to an increase in variance.

A.1.2. Behavior of GUESS in Higher Dimensions

We study the behavior of GUESS for higher-dimensional problems up to 32 dimensions and compare it to the baseline LHS. SVGP is used to carry out the benchmarks for dimensions greater than 8. SVGP can mitigate the computational burden and allow for much larger datasets. Instead of conditioning on all available samples, an SVGP model learns a selected subset of so called *inducing points*. This can reduce time complexity from $\mathcal{O}(m^3)$ for GP to $\mathcal{O}(m_\mathrm{u}^3)$ since generally $m_\mathrm{u} \ll m$, where m_u are the number of *inducing variables* $\mathbf{u} \in \mathrm{Y}$, defined at *inducing locations* $\mathbf{z} \in \mathrm{X}$. This already indicates that the computational effort will be constant after observing $m \geq m_\mathrm{u}$ samples, since m_u is constant.

The variational approach defines an approximate posterior over the inducing variables as $q(\mathbf{u}) = \mathcal{N}(\boldsymbol{\mu}_U, \Sigma_U)$, where mean vector $\boldsymbol{\mu}_U \in \mathbb{R}^{m_\mathrm{u}}$ and covariance matrix $\Sigma_U \in \mathbb{R}^{m_\mathrm{u} \times m_\mathrm{u}}$ are variational parameters that have to be inferred. Therefore, inducing locations, variational and GP parameters are learned jointly by maximizing a lower bound to the marginal likelihood called evidence lower bound [10].

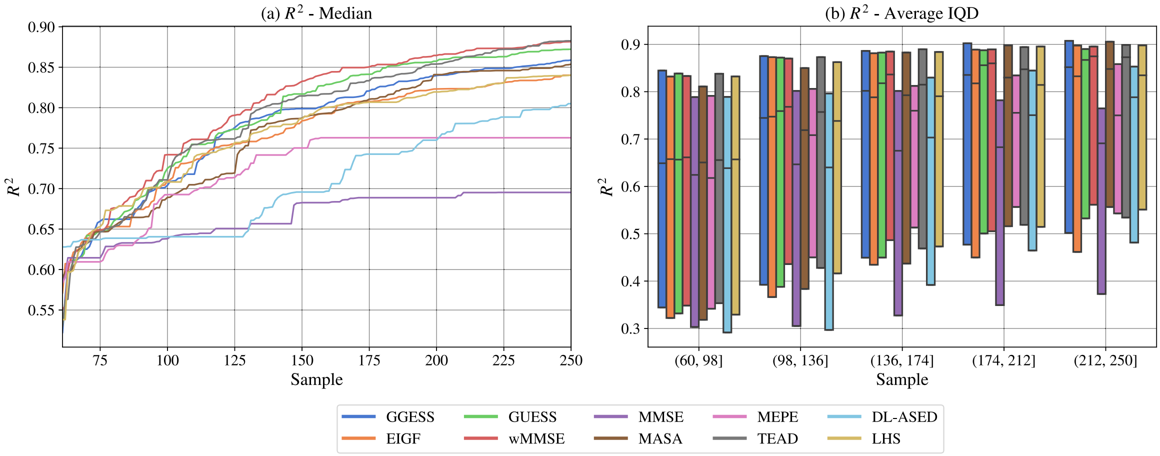

Fig. A.5. Comparison of different adaptive sampling strategies with GP surrogate model across 10 repetitions for 6D benchmark functions: ACKLEY6D, ROSENBROCK6D, MICHALEWICZ6D and HARTMANN (see Appendix C). (a) R^2 median over the conducted benchmarks and repetitions, (b) IQD of the average R^2 within each group; boxes indicate the 25% and 75% quantile and the horizontal line the median.

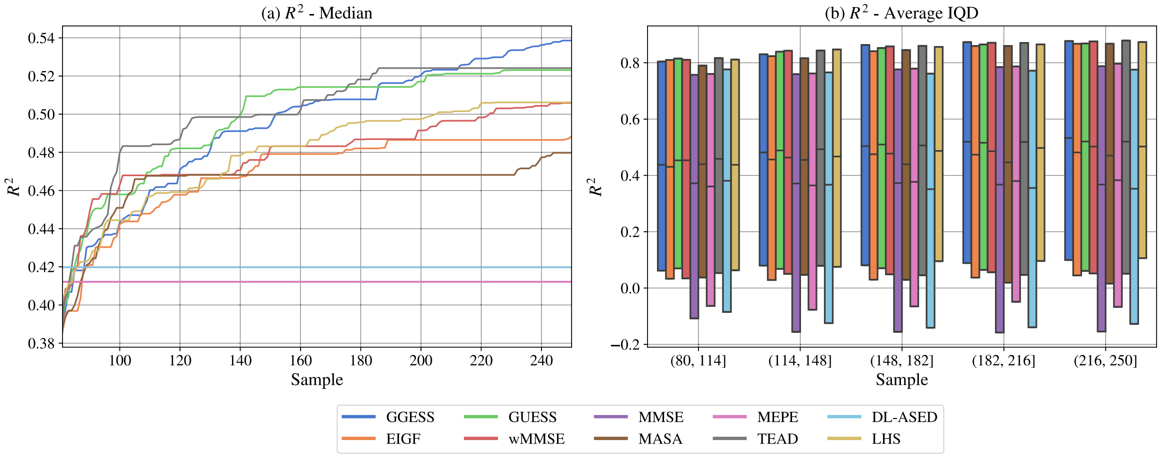

Fig. A.6. Comparison of different adaptive sampling strategies with GP surrogate model across 10 repetitions for 8D benchmark functions: ACKLEY8D, ROSENBROCK8D, MICHALEWICZ8D and STYBLINSKITANG8D (see Appendix C). (a) R^2 median over the conducted benchmarks and repetitions, (b) IQD of the average R^2 within each group; boxes indicate the 25% and 75% quantile and the horizontal line the median.

The approximate posterior mean and variance is given by

$$\hat{f}_{\mathrm{SGP}}(\mathbf{x}^*;\boldsymbol{\theta}_{\mathrm{GP}}) = \mathbf{k}_{\mathrm{Z}}^T \mathbf{K}_{\mathrm{Z}}^{-1} \boldsymbol{\mu}_U$$

$$\mathbb{V}\left[\hat{f}_{\mathrm{SGP}}(\mathbf{x}^*;\boldsymbol{\theta}_{\mathrm{GP}})\right] = \mathbf{K} + \mathbf{k}_{\mathrm{Z}}^T \mathbf{K}_{\mathrm{Z}}^{-1}(\Sigma_U - \mathbf{K}_{\mathrm{Z}})\mathbf{K}_{\mathrm{Z}}^{-1}\mathbf{k}_{\mathrm{Z}}$$

where $\mathbf{k}_{\mathrm{Z}}$ and $\mathbf{K}_{\mathrm{Z}}$ are defined similarly to $\mathbf{k}$ and $\mathbf{K}$ by using inducing locations $\mathbf{z}$ instead of $\mathbf{x}$, respectively. See [10,56] for more details.

Fig. A.7 shows the influence of increasing dimensionality on method performance. Similar experimental settings to Section 4.1 and Appendix A.1.1 were used. For SVGP, we used $m_{\mathrm{u}} = 256$ and Matérn kernel ($\nu = 3/2$). Above 8 dimensions, $m_{\mathrm{cand}} = 80\,000$ were used and the model was trained only every second iteration. Even if the model

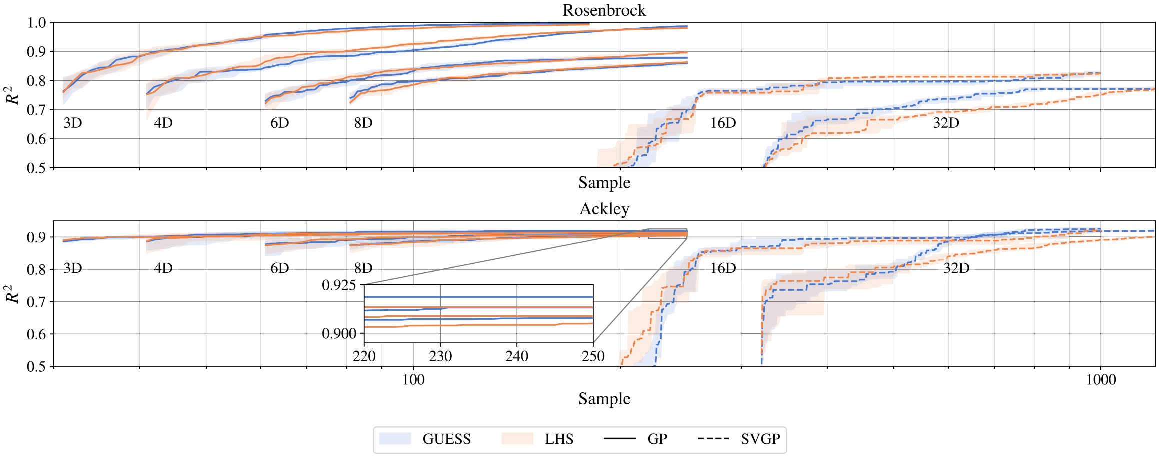

Fig. A.7. Comparison of GUESS and LHS for 3 to 32 dimensions across 10 repetitions of ROSENBROCK (top) and ACKLEY (bottom) benchmark functions (see Appendix C). GP was used for 3 to 8 dimensions and SVGP for the remaining. IQD of the R^2 score across proposed samples is displayed; filled area indicate the 25% and 75% quantile and line the median. Magnified area for ACKLEY shows 4D, 6D and 8D from top to bottom, respectively for GUESS and LHS.

Table A.4
Average number of samples needed to achieve a defined accuracy target (R^2) across dimensions based on results from GUESS and LHS. GP was used for 3 to 8 dimensions and SVGP for the remaining. None of the runs could achieve the accuracy goal within 1200 samples for ROSENBROCK and 32D. Runs underperforming the accuracy target are not considered.

Function	Target R^2	3D	4D	6D	8D	16D	32D
ROSENBROCK	≥0.8	33	46	80	107	547	–
ACKLEY	≥0.9	45	57	103	160	682	844

was not trained at an iteration, new samples are still added to the model but the same kernel parameters as the previous iteration were used. Considering only results from 16 and 32 dimensions, GUESS achieved the highest median R^2 (0.86) and R^2_{Area} score (0.806) on average, in contrast to LHS with $R^2 = 0.855$ and $R^2_{\text{Area}} = 0.792$. It can be observed that with growing dimensionality, the sample size increases to achieve an equivalent R^2 score (see Table A.4).

A.1.3. Challenges and approaches in high dimensions

Adaptive sampling in high dimensions is challenging, yet such problems are common in different real-world applications. The curse of dimensionality concerns both the surrogate model and the optimization of the acquisition function. With increasing dimensionality and sample requirements to achieve sufficient accuracy GP may become infeasible due to $\mathcal{O}(m^3)$ time and $\mathcal{O}(m^2)$ memory complexity [5], therefore sparse approximations [10], or alternatives like batch DGCNs [13] or PNNs have to be used. Optimizing Eq. (6) becomes also increasingly difficult, especially for candidate based acquisition functions, since larger candidate sets have to be used to cover the whole optimization domain. Previous research [33] introduced a partitioning of the optimization domain based on Voronoi cells to identify sampling regions, aiming to improve the optimization efficiency which does not account for the modeling difficulties.

In the related field of BO, different approaches dealing with high dimensionality were proposed [57]. However, most assume the high-dimensional objective function has lower intrinsic dimensionality, i.e. the response function is insensitive to changes in most of the dimensions. Therefore, global sensitivity analysis [58] could be used to select only input variables which have important contribution to the response variability. Alternatively, embeddings based on linear [59,60] or non-linear projections [61,62] could be used to learn a reduced latent representation of the input. However, none of these methods are efficient if the response function is sensitive to all dimensions. Although there is some ongoing effort to solve them [57], such problems remain important opportunities for future research.

A.2. Influence of surrogate model

The adaptive sampling strategies were tested with the 4D benchmark functions using DGCN and PNN as an alternative to GP. The aim of this study was to investigate the influence of the model choice.

A.2.1. DGCN

DGCN is an anisotropic and non-stationary GP, where each input variable has its own length scale l and noise variance prediction σ_{n}^2. l and σ_{n}^2 are learned by an ANN such that for an input $\mathbf{x} \in \mathrm{X}$, the GP parameters can be predicted $\hat{f}_{\mathrm{ANN}} : \mathrm{X} \times \mathrm{X} \to \mathbb{R}^{n_\theta}$, where n_θ are the number of GP parameters. Since the length scale is no longer fixed over the training samples, the covariance function has to be reparameterized using two length scales l and l'. For the Matérn kernel this can be done by replacing r/l in Eq. (1) with $\sqrt{\sum_{i=1}^{n} \left(x_i/l - x_i'/l'\right)^2}$.

Five different covariance functions are combined to approximate complex functions; squared exponential, absolute exponential, two Matérn functions ($\nu = 3/2$ and $\nu = 5/2$) and a rational quadratic function. The free parameters $\mathbf{l}$ of the kernel functions are then learned by the ANN. Stochastic gradient descent (e.g. [63]) is used to obtain ANN parameters by maximizing the marginal log-likelihood

$$\log p(\mathbf{y}|\mathbf{X}, \hat{f}_{\mathrm{ANN}}) = -\frac{1}{2}\mathbf{y}\mathbf{K}_{\mathrm{C}}^{-1}\mathbf{y} - \frac{1}{2}\log|\mathbf{K}_{\mathrm{C}}| - \frac{m}{2}\log(2\pi)$$

where the covariance matrix $\mathbf{K}_{\mathrm{N}}$ in Eq. (2) is replaced with

$$\mathbf{K}_{\mathrm{C}} = \sum_{i=1}^{n_{\mathrm{C}}} K_i(\mathbf{X}, \mathbf{X}; \hat{f}_{\mathrm{ANN}}(\mathbf{X})) + \sigma_{\mathrm{n}}^2(\mathbf{X})\mathbf{I}$$

where $K : \mathrm{X}^m \times \mathrm{X}^m \to \mathbb{R}^{m\times m}$ is the covariance matrix function, n_{C} representing the number of covariance functions used, and $\sigma_{\mathrm{n}}^2 : \mathrm{X}^m \to \mathbb{R}^m$ is obtained by a further ANN (see [13]). Eqs. (3) and (4) can be adapted for prediction at a query $\mathbf{x}^*$ as

$$\hat{f}_{\mathrm{DGCN}}(\mathbf{x}^*; \hat{f}_{\mathrm{ANN}}) = \mathbf{k}_{\mathrm{C}}^T\mathbf{K}_{\mathrm{C}}^{-1}\mathbf{y}$$

$$\mathbb{V}\left[\hat{f}_{\mathrm{DGCN}}(\mathbf{x}^*; \hat{f}_{\mathrm{ANN}})\right] = k_{\mathrm{C}}(\mathbf{x}^*, \mathbf{x}^*; \tilde{\mathbf{l}}^*, \tilde{\mathbf{l}}^*) + \hat{\sigma}_{\mathrm{n}}^2(\mathbf{x}^*) - \mathbf{k}_{\mathrm{C}}^T\mathbf{K}_{\mathrm{C}}^{-1}\mathbf{k}_{\mathrm{C}}$$

with the combination of the different kernel functions $k_{\mathrm{C}}(\mathbf{x}^*, \mathbf{x}^*; \tilde{\mathbf{l}}^*, \tilde{\mathbf{l}}^*) = \sum_{i=1}^{n_{\mathrm{C}}} k_i(\mathbf{x}^*, \mathbf{x}^*; \tilde{\mathbf{l}}^*, \tilde{\mathbf{l}}^*)$ and the vector of correlations as $\mathbf{k}_{\mathrm{C}} = \left[k_{\mathrm{C}}(\mathbf{x}^*, \mathbf{x}_1; \tilde{\mathbf{l}}^*, \tilde{\mathbf{l}}_1), \ldots, k_{\mathrm{C}}(\mathbf{x}^*, \mathbf{x}_m; \tilde{\mathbf{l}}^*, \tilde{\mathbf{l}}_m)\right]^T$. $\hat{\sigma}_{\mathrm{n}}^2(\mathbf{x}^*)$ is the predicted noise and $\tilde{\mathbf{l}}$ the vector of predicted length scales from the ANN. Additional details like the net topology, partial derivatives, etc. can be found in [13].

The results for the 4D benchmark functions are given in Fig. A.8. DGCN achieved a higher accuracy for all sampling strategies compared to GP (Fig. 4). With DGCN and 100 samples, a similar or better median R^2 score could be achieved for all methods, compared to adaptive sampling with GP and 250 samples.

All methods could achieve a median R^2 score close to 1 within 250 samples. It can be seen that MASA performed robust with both surrogate models, achieving the highest R^2 and second highest R^2_{Area} score on average with DGCN. Regarding the performance, it can be seen that MMSE and MEPE performed better with DGCN, while the gradient-based acquisition functions got relatively worse.[2] This is also reflected within the R^2_{Area} ranks, where GGESS, TEAD and GUESS were the only methods losing ranks compared to GP. This could be partially due to the use of forward differences [64] for calculating the gradients of the DGCN model.

We conclude that the impact of the surrogate model choice can be much greater than the adaptive sampling strategy regarding the achieved accuracy, and thus the sample efficiency for the tested case. It can be seen that the choice of surrogate model can change the ranking of the sampling strategies drastically. However, the right choice of adaptive sampling strategy can improve the sample efficiency further, e.g. considering the performance difference between GGESS and EIGF in Fig. A.8.

A.2.2. Other non-kernel methods

Ensembles of bootstrapped PNNs were investigated alongside the kernel-based methods from the previous section to be used as the surrogate model for adaptive sampling strategies. The PNN is an ANN, where the output neurons

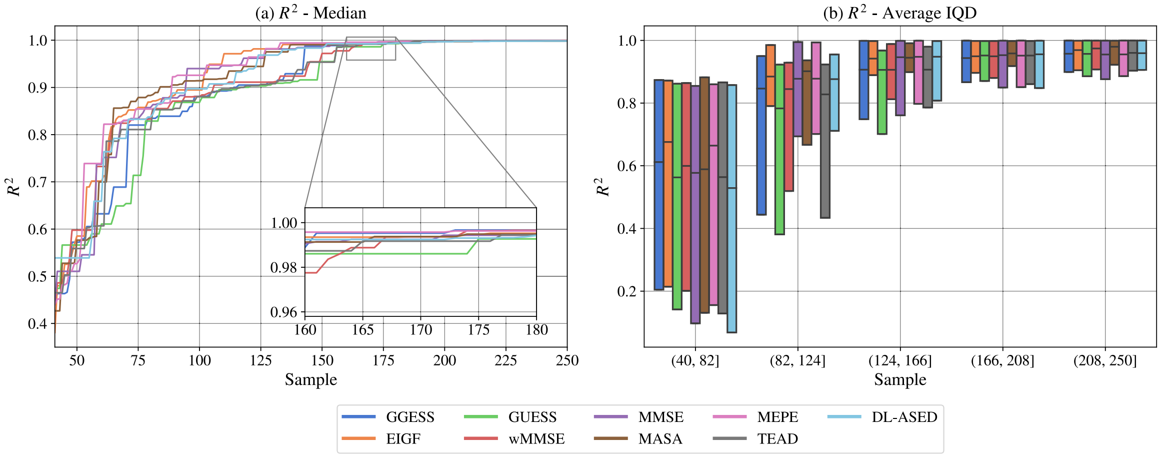

Fig. A.8. Comparison of different adaptive sampling strategies with DGCN surrogate model across 10 repetitions for 4D benchmark functions: ACKLEY4D, ROSENBROCK4D, MICHALEWICZ4D and STYBLINSKITANG4D. (a) R^2 median over the conducted benchmarks and repetitions, (b) IQD of the average R^2 within each group; boxes indicate the 25% and 75% quantile and the horizontal line the median.

parameterize a probability distribution function to capture the aleatoric uncertainty represented in the dataset. A simple feed forward network with parameters $\boldsymbol{\theta}$ and n_z hidden layers can be written as

$$\begin{aligned}\mathbf{z}_0 &= \mathbf{x},\\ \mathbf{z}_i &= \psi_i\left(\mathbf{W}_i\mathbf{z}_{i-1} + \mathbf{b}_i\right) \quad \forall i \in [1, n_z],\\ \mathbf{y} &= \mathbf{z}_{n_z}\end{aligned} \tag{A.1}$$

where $\mathbf{W}_i$ are the weights and $\mathbf{b}_i$ the biases of the i-th layer. $\mathbf{z}_{i>0}$ represents the resulting latent variable after the linear transformations using $\mathbf{W}_i$, $\mathbf{b}_i$ and the non-linear activations $\psi_i(\cdot)$ of the neurons in the i-th hidden layer. For a Gaussian posterior distribution $\mathcal{N}(\mu, \sigma^2)$ with mean μ and variance σ^2, the PNN can be compactly written as

$$\begin{aligned}\mu_{\boldsymbol{\theta}}(\mathbf{x}) &= g_{\boldsymbol{\theta}}^{(1)}(\mathbf{x}),\\ \sigma_{\boldsymbol{\theta}}(\mathbf{x}) &= g_{\boldsymbol{\theta}}^{(2)}(\mathbf{x}),\\ y &\sim \mathcal{N}(\mu_{\boldsymbol{\theta}}(\mathbf{x}), \sigma_{\boldsymbol{\theta}}^2(\mathbf{x}))\end{aligned}$$

where $g_{\boldsymbol{\theta}}$ represents the network (Eq. (A.1)) with $\mathbf{z}_{n_z} \in \mathbb{R}^2$. The superscripts (1), (2) denote the first and second entry of the output vector. Furthermore, a loss proportional to the negative log-likelihood is used to train the model parameters $\boldsymbol{\theta}$ as [20]

$$\text{loss}(\boldsymbol{\theta}) = \sum_{i=1}^{m}\left(\frac{\mu_{\boldsymbol{\theta}}(\mathbf{x}_i) - y_i}{\sigma_{\boldsymbol{\theta}}(\mathbf{x}_i)}\right)^2 + \log\sigma_{\boldsymbol{\theta}}^2(\mathbf{x}_i) \tag{A.2}$$

Ensembles of bootstrapped PNNs can be used to estimate the epistemic model uncertainty without introducing new parameters into the network. This makes the training easier compared to a full Bayesian inference [65]. The ensemble $\mathcal{M}_{\text{PNN}} = \{\hat{f}_{\text{PNN},1}, \ldots, \hat{f}_{\text{PNN},n_{\text{en}}}\}$ can be constructed by sampling m times (with replacement) from $\mathcal{D}$, creating n_{en} different datasets for each of the n_{en} models. The ensemble is a uniformly weighted mixture model. Given a PNN with Gaussian posterior, the predictive mean and variance of the mixture can be calculated at a point $\mathbf{x}^*$ as [19]

$$\hat{f}_{\text{EPNN}}(\mathbf{x}^*; \mathcal{M}_{\text{PNN}}) = \mu_{Y^*} = \frac{1}{n_{\text{en}}}\sum_{j=1}^{n_{\text{en}}}\mu_{\boldsymbol{\theta}_j}(\mathbf{x}^*)$$

$$\mathbb{V}\left[\hat{f}_{\text{EPNN}}(\mathbf{x}^*; \mathcal{M}_{\text{PNN}})\right] = \sigma_{Y^*}^2 = \frac{1}{n_{\text{en}}}\sum_{j=1}^{n_{\text{en}}}\left(\sigma_{\boldsymbol{\theta}_j}^2(\mathbf{x}^*) + \mu_{\boldsymbol{\theta}_j}^2(\mathbf{x}^*)\right) - \mu_{Y^*}^2$$

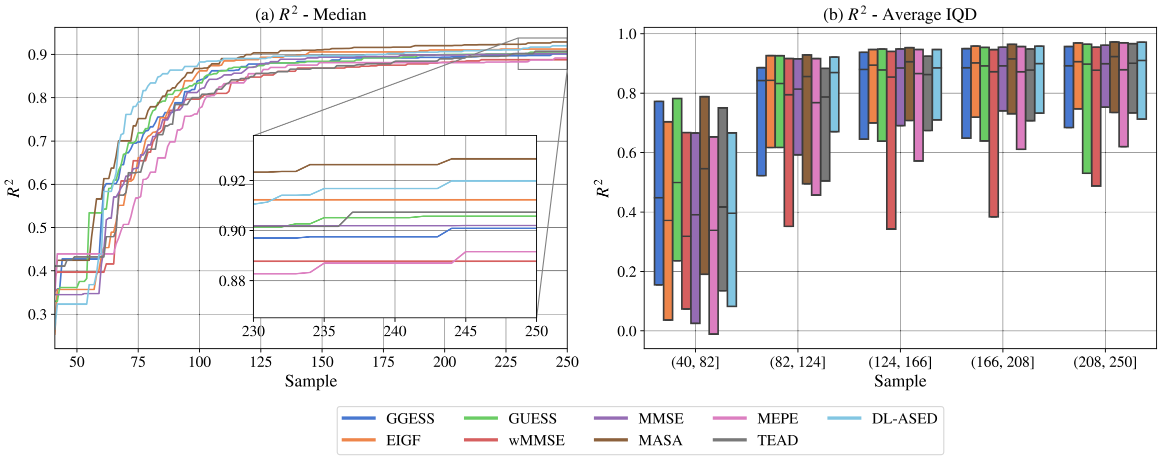

Fig. A.9. Comparison of different adaptive sampling strategies with PNN surrogate model across 10 repetitions for 4D benchmark functions: Ackley4D, Rosenbrock4D, Michalewicz4D and StyblinskiTang4D. (a) R^2 median over the conducted benchmarks and repetitions, (b) IQD of the average R^2 within each group; boxes indicate the 25% and 75% quantile and the horizontal line the median.

The advantage of ANNs and PNNs is the scalability to larger datasets, whereby smaller data can be approximated with e.g. non-linear regression in a Bayesian setting. Difficulties can arise with the selection of the net topology together with the right activation functions types, since it is crucial for the model accuracy, albeit non-trivial.

We tested the 4D benchmark functions with the ensemble PNN. Similar experimental settings to Section 4.1 and Appendix A.1.1 were used to investigate the influence of surrogate model. The squared error was used for MEPE, wMMSE and DL-ASED, since LOOCV (Eq. (5)) can be challenging to compute for ANNs and PNNs. Further implementation details can be found in Appendix B. The results for the 4D benchmark are shown in Fig. A.9. It can be seen that PNN achieved on median similar or better results compared to GP but performed worse than DGCN regarding the R^2 score.

MASA was found to score on average the highest R^2_{Area} value with 0.724 followed by GGESS (0.717) and GUESS (0.716). It can be seen that wMMSE performed worse compared to GP, which could be due to the replacement of the LOOCV. Similar ranks regarding R^2_{Area} can be reported for the 4D benchmark functions compared to GP for most methods.[2] The majority of methods stayed within one rank difference, while MASA, MMSE and DL-ASED performed better with PNN and TEAD got worse.

Considering results from all three models[2] for the 4D benchmark functions, we found that MASA provided the highest R^2 (0.849) and R^2_{Area} (0.724) score across all models on average. Furthermore, GUESS achieved the highest rank regarding R^2, second best R^2 score (0.842), and third highest rank regarding R^2_{Area}. MEPE showed the weakest performance with average R^2 of 0.83 and R^2_{Area} of 0.714 across all models.

A.3. Computational effort

Computational effort besides evaluating the black-box function is mainly driven by the expense of (re-)training the surrogate model (Step 3 in Section 3.1), and evaluation of ϕ for optimization or assessing the candidate points (Step 4 in Section 3.1). Model training is almost identical between adaptive strategies and independent of the acquisition function, except for MASA which has to update the whole ensemble. Therefore, we show in Fig. A.10 a comparison of the wall-clock time for proposing one new sample point (Step 4 in Section 3.1) with the different sampling strategies for the 1 to 8-dimensional benchmark study (Section 4). Benchmarks are run parallelized over all 10 repetitions on a desktop computer with Intel i7-6900k and 48 GB DDR-4 computer memory. The plot is limited to 10 s to improve visibility. Therefore, DL-ASED is not fully visible on the left for 6D (median: 39 s, 25%-quantile: 36 s, 75%-quantile: 41 s) and on the right (median: 18 s at 250 samples). Moreover, for DL-ASED we used only samples proposed by the acquisition function, i.e. we removed samples placed by edge sampling (see Section 4.1). Therefore, no results for 8D are shown for DL-ASED, due to exhausting sampling of the edges, i.e. sampling edges exceeded maximum number of samples.

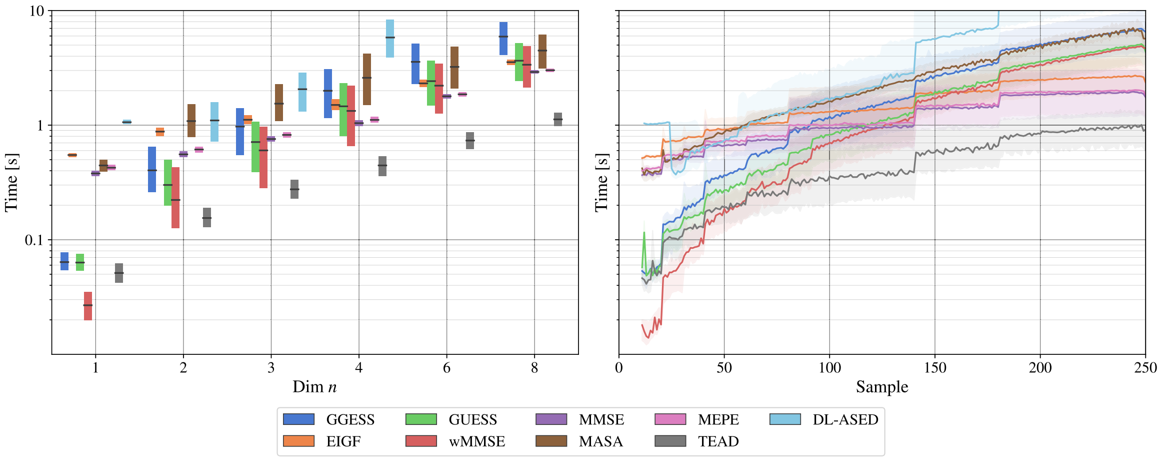

Fig. A.10. Comparison of wall-clock time [s] for proposing one new sample point (Step 4 in Section 3.1) for different adaptive sampling strategies with GP surrogate model for the 1 to 8-dimensional benchmark study (Section 4). Left: IQD of the time across dimensions; boxes indicate the 25% and 75% quantile and the horizontal line the median. Right: IQD of the time across proposed samples; filled area indicate the 25% and 75% quantile and the solid line the median. DL-ASED is not fully visible on the left plot for 6D (median: 39 s, 25%-quantile: 36 s, 75%-quantile: 41 s) and on the right (median: 18 s at 250 samples). For 8D no results for DL-ASED are shown, due to sampling of the edges.

Proposing new samples becomes slower with increasing sample size, partially since GP scales $\mathcal{O}(m^3)$ with sample size. It can be seen that with increasing dimensionality of the benchmark function, the proposal time also increases due to the increase in sample size. However, it should be noted the maximum number of samples was limited for 6 and 8-dimensional benchmarks to 250. As can be seen, TEAD needs comparably low compute for proposing new samples. We found the usage of model uncertainty in the acquisition function to be one of the main driver of computational complexity.

We can further improve the wall time by decreasing the model training frequency. A commonly used strategy for GP would be to train the surrogate model only every i-th iteration. In the remaining iterations, the GP can be conditioned on new observations, i.e. we would build a model including the new observations using the model parameters $\boldsymbol{\theta}_{\mathrm{GP}}$ from the previous iteration. Moreover, since computational effort was not the focus of this study, more efficient implementation of the strategies may improve computation further. For sampling in high dimensions or larger sample sizes, one could also consider approaches mentioned in Appendix A.1.3.

Appendix B. Implementation details

Our code was implemented in Python [66] with several different packages and is available on GitHub[2] for further research. The GP used in this work is based on the implementation in the scikit-optimize library [67] which relies on the library scikit-learn [68]. For training of the GP the default optimizer based on scipy's [69] implementation of the L-BFGS-B algorithm [70] was used with 10 random restarts to maximize the log-marginal likelihood. A small value (10^{-10}) is added to the diagonal of the kernel matrix during fitting to provide numerical stability. A PNN implementation based on [20] was used with two different net topologies and $n_{\mathrm{en}} = 5$. A network with one layer, 64 neurons and Swish activation function were used for Ackley4D, Rosenbrock4D and StyblinskiTang4D. Furthermore, a network with one layer, 48 neurons and Tanh activation function were used for Michalewicz4D. Adam [63] was used throughout with a learning rate of 0.03 to minimize the loss function (Eq. (A.2)). DGCN is a custom implementation based on [13] which uses TensorFlow [71] as backend and five different kernel functions as described in [13]. For SVGP, we used the implementation from [72]. Furthermore, two evolutionary strategies were used for optimizing the continuous acquisition function in Eq. (6), as implemented in pygmo [73] (for GP) and with the differential evolution algorithm in scipy (for DGCN and PNN). The baseline LHS used in the benchmarks is a custom implementation based on [74], where samples are drawn from X without correlation and by maximizing pairwise distance. Additionally, the implementation in the scikit-optimize library [67] was used for all other use cases. R^2_{Area} in Eq. (14) was calculated with the composite Simpson' rule from the library scipy [69]. The parameter

Table C.5
Benchmark functions.

Name	Input	Function
HumpSingle(modified)	$x \in [-1.5, 5]$	$f_{\mathrm{HS}}(\mathbf{x}) = \dfrac{0.05}{(x-4.75)^2+0.004} - \dfrac{0.09}{(x-4.45)^2+0.05} - 6 + 3x$
HumpTwo(modified)	$x \in [-0.5, 5]$	$f_{\mathrm{HT}}(\mathbf{x}) = 5x + \dfrac{0.05}{(x-4.5)^2+0.002} - \dfrac{0.5}{(x-3.5)^2+3.5} - 6$
GramLee [75]	$x \in [-1.5, 1]$	$f_{\mathrm{GL}}(\mathbf{x}) = \dfrac{60\sin(6\pi x)}{2\cos(x)} + (x-1)^4$
BekerLogan [76]	$x_d \in [-10, 10]$, $d = 1, 2$	$f_{\mathrm{BL}}(\mathbf{x}) = (\lvert x_1\rvert - 5)^2 + (\lvert x_2\rvert - 5)^2$
Eggholder [77]	$x_d \in [-512, 512]$, $d = 1, 2$	$f_{\mathrm{EGG}}(\mathbf{x}) = -(x_2+47)\sin\left(\sqrt{\lvert x_2 + 0.5x_1 + 47\rvert}\right) - x_1 \sin\left(\sqrt{\lvert x_1 - (x_2+47)\rvert}\right)$
Himmelblau [78]	$x_d \in [-6, 6]$, $d = 1, 2$	$f_{\mathrm{HIM}}(\mathbf{x}) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$
Branin [79]	$x_d \in [-5, 10]$, $d = 1, 2$	$f_{\mathrm{BRN}}(\mathbf{x}) = \left(x_2 - \dfrac{5.1}{4\pi^2}x_1^2 + \dfrac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \dfrac{1}{8\pi}\right)\cos(x_1) + 10$
DropWave [80]	$x_d \in [-0.6, 0.9]$, $d = 1, 2$	$f_{\mathrm{DRP}}(\mathbf{x}) = -\left(1 + \cos\left(12\sqrt{x_1^2 + x_2^2}\right)\right)\left(0.5(x_1^2 + x_2^2) + 2\right)^{-1}$
Ishigami [81]	$x_d \in [-\pi, \pi]$, $d = 1, 2, 3$	$f_{\mathrm{ISH}}(\mathbf{x}) = \sin(x_1) + 7\sin^2(x_2) + 0.1x_3^4\sin(x_1)$
Hartmann [82]	$x_d \in [0, 1]$, $d = 1, \dots, 6$	$f_{\mathrm{HRT}}(\mathbf{x}) = -\sum_{i=1}^{4}\boldsymbol{\alpha}_i \exp\left(-\sum_{j=1}^{6}\mathbf{A}_{ij}(x_j - \mathbf{P}_{ij})^2\right),$ where $\boldsymbol{\alpha} = (1, 1.2, 3, 3.2)^T,$ $\mathbf{A} = \begin{pmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 7 \\ 17 & 8 & 0.05 & 10 & 0.01 & 14 \end{pmatrix}, \mathbf{P} = 10^{-4}\begin{pmatrix} 1312 & 1696 & 5569 & 124 & 8283 & 5886 \\ 2329 & 4135 & 8307 & 3736 & 1004 & 9991 \\ 2348 & 1451 & 3522 & 2883 & 3047 & 6650 \\ 4047 & 8828 & 8732 & 5743 & 1091 & 381 \end{pmatrix}$
Rosenbrock [83]	$x_d \in [-5, 5]$, $d = 1, \dots, n$	$f_{\mathrm{ROS}}(\mathbf{x}) = -\sum_{i=1}^{n-1}\left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right]$
Ackley [84]	$x_d \in [-5, 5]$, $d = 1, \dots, n$	$f_{\mathrm{ACL}}(\mathbf{x}) = -20\exp\left(-0.2\sqrt{n^{-1}\sum_{i=1}^{n}x_i^2}\right) - \exp\left(n^{-1}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + \exp(1)$
Michalewicz [85]	$x_d \in [0, \pi]$, $d = 1, \dots, n$	$f_{\mathrm{MIC}}(\mathbf{x}) = -\sum_{i=1}^{n}\left[\sin(x_i)\sin^{10}\left(\dfrac{ix_i^2}{\pi}\right)\right]$
Schwefel [86]	$x_d \in [-5, 5]$, $d = 1, \dots, n$	$f_{\mathrm{SCH}}(\mathbf{x}) = 418.9829n - \sum_{i=1}^{n}x_i\sin\left(\sqrt{x_i}\right)$
StyblinskiTang [87]	$x_d \in [-5, 5]$, $d = 1, \dots, n$	$f_{\mathrm{STY}}(\mathbf{x}) = 0.5\sum_{i=1}^{n}\left(x_i^4 - 16x_i^2 + 5x_i\right)$

‘even=last’ has to be passed at the function call together with a vector with s evenly spaced values on $[0, 1]$ to yield the same results as explained in Eq. (14), including start and end points.

Appendix C. Benchmark functions

Table C.5 gives an overview of the 15 different benchmark functions used throughout this study. The last five functions can be adapted to different dimensions.

References

[1] G.E. Karniadakis, I.G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, et al., Physics-informed machine learning, Nat. Rev. Phys. 3 (6) (2021) 422–440, http://dx.doi.org/10.1038/s42254-021-00314-5.

[2] L. Lu, P. Jin, G. Pang, Z. Zhang, G.E. Karniadakis, Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators, Nat. Mach. Intell. 3 (3) (2021) 218–229, http://dx.doi.org/10.1038/s42256-021-00302-5.

[3] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, et al., Fourier neural operator for parametric partial differential equations, 2021, arXiv:2010.08895.

[4] A. Forrester, A. Sobester, A. Keane, Engineering Design Via Surrogate Modelling: A Practical Guide, John Wiley & Sons, 2008.

[5] C.E. Rasmussen, C.K.I. Williams, Gaussian Processes for Machine Learning, in: Adaptive Computation and Machine Learning, MIT Press, Cambridge, 2006.

[6] G. Kopsiaftis, E. Protopapadakis, A. Voulodimos, N. Doulamis, A. Mantoglou, Gaussian process regression tuned by Bayesian optimization for seawater intrusion prediction, Comput. Intell. Neurosci. 2019 (2019) http://dx.doi.org/10.1155/2019/2859429.

[7] D. Sterling, T. Sterling, Y. Zhang, H. Chen, Welding parameter optimization based on Gaussian process regression Bayesian optimization algorithm, in: 2015 IEEE International Conference on Automation Science and Engineering, CASE, IEEE, Gothenburg, Sweden, 2015, pp. 1490–1496, http://dx.doi.org/10.1109/CoASE.2015.7294310.

[8] A. Damianou, N.D. Lawrence, Deep Gaussian Processes, in: C.M. Carvalho, P. Ravikumar (Eds.), Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics, in: Proceedings of Machine Learning Research, vol. 31, PMLR, Scottsdale, Arizona, USA, 2013, pp. 207–215.

[9] M. Titsias, N.D. Lawrence, Bayesian gaussian process latent variable model, in: Y.W. Teh, M. Titterington (Eds.), Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, in: Proceedings of Machine Learning Research, vol. 9, PMLR, Chia Laguna Resort, Sardinia, Italy, 2010, pp. 844–851.

[10] J. Hensman, N. Fusi, N.D. Lawrence, Gaussian Processes for Big Data, in: Twenty-Ninth Conference on Uncertainty in Artificial Intelligence, Bellevue, Washington, USA, 2013, pp. 282–290, http://dx.doi.org/10.48550/ARXIV.1309.6835.

[11] K.A. Wang, G. Pleiss, J.R. Gardner, S. Tyree, K.Q. Weinberger, et al., Exact Gaussian Processes on a million data points, in: 33nd Conference on Neural Information Processing Systems, Vancouver, Canada, 2019, pp. 14648–14659.

[12] K. Cremanns, D. Roos, Deep Gaussian covariance network, 2017, arXiv:1710.06202.

[13] K. Cremanns, Probabilistic Machine Learning for Pattern Recognition and Design Exploration (Ph.D. thesis), RWTH Aachen University, 2021.

[14] Y. Gal, Z. Ghahramani, Dropout as a Bayesian approximation: Representing model uncertainty in deep learning, in: M.F. Balcan, K.Q. Weinberger (Eds.), Proceedings of the 33rd International Conference on Machine Learning, in: Proceedings of Machine Learning Research, vol. 48, PMLR, New York, New York, USA, 2016, pp. 1050–1059.

[15] D.J.C. MacKay, A practical Bayesian framework for backpropagation networks, Neural Comput. 4 (3) (1992) 448–472, http://dx.doi.org/10.1162/neco.1992.4.3.448.

[16] J. Lampinen, A. Vehtari, Bayesian approach for neural networks— review and case studies, Neural Netw. 14 (3) (2001) 257–274, http://dx.doi.org/10.1016/S0893-6080(00)00098-8.

[17] D.M. Titterington, Bayesian methods for neural networks and related models, Statist. Sci. 19 (1) (2004) http://dx.doi.org/10.1214/088342304000000099.

[18] R.M. Neal, Bayesian Learning for Neural Networks, in: Lecture Notes in Statistics, Springer New York, New York, 1996.

[19] B. Lakshminarayanan, A. Pritzel, C. Blundell, Simple and scalable predictive uncertainty estimation using deep ensembles, in: 31nd Conference on Neural Information Processing Systems, Long Beach, California, USA, 2017, pp. 6405–6416.

[20] K. Chua, R. Calandra, R. McAllister, S. Levine, Deep reinforcement learning in a handful of trials using probabilistic dynamics models, in: 32nd Conference on Neural Information Processing Systems, Montréal, Canada, 2018, pp. 4759–4770.

[21] J. Sacks, W.J. Welch, T.J. Mitchell, H.P. Wynn, Design and analysis of computer experiments, Statist. Sci. 4 (4) (1989) http://dx.doi.org/10.1214/ss/1177012413.

[22] M.D. McKay, R.J. Beckman, W.J. Conover, A comparison of three methods for selecting values of input variables in the analysis of output from a computer code, Technometrics 21 (2) (1979) 239, http://dx.doi.org/10.2307/1268522, arXiv:1268522.

[23] D. Roos, Latin hypercube sampling based on adaptive orthogonal decomposition, in: Proceedings of the VII European Congress on Computational Methods in Applied Sciences and Engineering, ECCOMAS Congress 2016, Institute of Structural Analysis and Antiseismic Research School of Civil Engineering National Technical University of Athens (NTUA) Greece, Crete Island, Greece, 2016, pp. 3333–3343, http://dx.doi.org/10.7712/100016.2038.7644.

[24] D. Huntington, C. Lyrintzis, Improvements to and limitations of Latin hypercube sampling, Probab. Eng. Mech. 13 (4) (1998) 245–253, http://dx.doi.org/10.1016/S0266-8920(97)00013-1.

[25] J. Mockus, Application of Bayesian approach to numerical methods of global and stochastic optimization, J. Global Optim. 4 (4) (1994) 347–365, http://dx.doi.org/10.1007/BF01099263.

[26] D.R. Jones, M. Schonlau, W.J. Welch, Efficient global optimization of expensive black-box functions, J. Global Optim. 13 (4) (1998) 455–492, http://dx.doi.org/10.1023/A:1008306431147.

[27] D.G. Krige, A Statistical Approach to Some Basic Mine Valuation Problems on the Witwatersrand, Chemical, Metallurgical and Mining Society of South Africa, 1951.

[28] K. Crombecq, D. Gorissen, D. Deschrijver, T. Dhaene, A novel hybrid sequential design strategy for global surrogate modeling of computer experiments, SIAM J. Sci. Comput. 33 (4) (2011) 1948–1974, http://dx.doi.org/10.1137/090761811.

[29] H. Liu, S. Xu, Y. Ma, X. Chen, X. Wang, An adaptive Bayesian sequential sampling approach for global metamodeling, J. Mech. Des. 138 (1) (2015) http://dx.doi.org/10.1115/1.4031905.

[30] H. Liu, J. Cai, Y.-S. Ong, An adaptive sampling approach for Kriging metamodeling by maximizing expected prediction error, Comput. Chem. Eng. 106 (2017) 171–182, http://dx.doi.org/10.1016/j.compchemeng.2017.05.025.

[31] A.P. Kyprioti, J. Zhang, A.A. Taflanidis, Adaptive design of experiments for global Kriging metamodeling through cross-validation information, Struct. Multidiscip. Optim. 62 (3) (2020) 1135–1157, http://dx.doi.org/10.1007/s00158-020-02543-1.

[32] C.Q. Lam, Sequential Adaptive Design in Computer Experiments for Response Surface Model Fit (Ph.D. thesis), The Ohio State University, 2008.

[33] X. Chen, Y. Zhang, W. Zhou, W. Yao, An effective gradient and geometry enhanced sequential sampling approach for Kriging modeling, Struct. Multidiscip. Optim. 64 (6) (2021) 3423–3438, http://dx.doi.org/10.1007/s00158-021-03016-9.

[34] K. Fang, Y. Zhou, P. Ma, An adaptive sequential experiment design method for model validation, Chin. J. Aeronaut. 33 (6) (2019) 1661–1672, http://dx.doi.org/10.1016/j.cja.2019.12.026.

[35] S. Mo, D. Lu, X. Shi, G. Zhang, M. Ye, et al., A Taylor expansion-based adaptive design strategy for global surrogate modeling with applications in groundwater modeling, Water Resour. Res. 53 (12) (2017) http://dx.doi.org/10.1002/2017WR021622.

[36] J. Eason, S. Cremaschi, Adaptive sequential sampling for surrogate model generation with artificial neural networks, Comput. Chem. Eng. 68 (2014) 220–232, http://dx.doi.org/10.1016/j.compchemeng.2014.05.021.

[37] H. Liu, Y.-S. Ong, J. Cai, A survey of adaptive sampling for global metamodeling in support of simulation-based complex engineering design, Struct. Multidiscip. Optim. 57 (1) (2018) 393–416, http://dx.doi.org/10.1007/s00158-017-1739-8.

[38] J.N. Fuhg, A. Fau, U. Nackenhorst, State-of-the-art and comparative review of adaptive sampling methods for Kriging, Arch. Comput. Methods Eng. 28 (4) (2021) 2689–2747, http://dx.doi.org/10.1007/s11831-020-09474-6.

[39] C. Wu, M. Zhu, Q. Tan, Y. Kartha, L. Lu, A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks, Comput. Methods Appl. Mech. Engrg. 403 (2023) 115671, http://dx.doi.org/10.1016/j.cma.2022.115671.

[40] K. Tang, X. Wan, C. Yang, DAS-PINNs: A deep adaptive sampling method for solving high-dimensional partial differential equations, J. Comput. Phys. 476 (2023) 111868, http://dx.doi.org/10.1016/j.jcp.2022.111868.

[41] Y. Gu, H. Yang, C. Zhou, SelectNet: Self-paced learning for high-dimensional partial differential equations, J. Comput. Phys. 441 (2021) 110444, http://dx.doi.org/10.1016/j.jcp.2021.110444.

[42] A.M. Kupresanin, G. Johannesson, Comparison of Sequential Designs of Computer Experiments in High Dimensions, Technical Report LLNL-TR-491692, Lawrence Livermore National Lab. (LLNL), United States, 2011.

[43] T.J. Mackman, C.B. Allen, M. Ghoreyshi, K.J. Badcock, Comparison of adaptive sampling methods for generation of surrogate aerodynamic models, AIAA J. 51 (4) (2013) 797–808, http://dx.doi.org/10.2514/1.J051607.

[44] M.L. Stein, Interpolation of Spatial Data, in: Springer Series in Statistics, Springer, New York, 1999, http://dx.doi.org/10.1007/978-1-4612-1494-6.

[45] B. Matérn, Spatial Variation, second ed., in: Lecture Notes in Statistics, Springer, New York, 1986.

[46] M. Abramowitz, I.A. Stegun (Eds.), Handbook of Mathematical Functions: With Formulas, Graphs, and Mathematical Tables, tenth ed., Applied Mathematics, (no. 55) National Bureau of Standards, 1972.

[47] J. Hensman, A.G. Matthews, M. Filippone, Z. Ghahramani, MCMC for variationally sparse Gaussian Processes, in: Advances in Neural Information Processing Systems, Vol. 28, Curran Associates, Inc., 2015, p. 9.

[48] R. Kohavi, A study of cross-validation and bootstrap for accuracy estimation and model selection, in: Ijcai, Vol. 14, 1995.

[49] S. Sundararajan, S.S. Keerthi, Predictive approaches for choosing hyperparameters in gaussian processes, Neural Comput. 13 (5) (2001) 1103–1118, http://dx.doi.org/10.1162/08997660151134343.

[50] H.-G. Beyer, The Theory of Evolution Strategies, Springer, Berlin, Heidelberg, 2001.

[51] J.L. Loeppky, J. Sacks, W.J. Welch, Choosing the sample size of a computer experiment: A practical guide, Technometrics 51 (4) (2009) 366–376, http://dx.doi.org/10.1198/TECH.2009.08040.

[52] W. Sewall, Correlation and causation, J. Agric. Res. 20 (3) (1921) 557–585.

[53] D. Chicco, M.J. Warrens, G. Jurman, The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation, PeerJ Comput. Sci. 7 (2021) http://dx.doi.org/10.7717/peerj-cs.623.

[54] S. Venkateshan, P. Swaminathan, Numerical integration, in: Computational Methods in Engineering, Elsevier, 2014, pp. 317–373, http://dx.doi.org/10.1016/B978-0-12-416702-5.50009-0.

[55] D.T. Lee, B.J. Schachter, Two algorithms for constructing a Delaunay triangulation, Int. J. Comput. Inf. Sci. 9 (3) (1980) 219–242, http://dx.doi.org/10.1007/BF00977785.

[56] D.R. Burt, C.E. Rasmussen, M. van der Wilk, Convergence of sparse variational inference in Gaussian Processes Regression, J. Mach. Learn. Res. 21 (131) (2020) 1–63.

[57] X. Wang, Y. Jin, S. Schmitt, M. Olhofer, Recent advances in Bayesian optimization, ACM Comput. Surv. (2023) http://dx.doi.org/10.1145/3582078.

[58] A. Saltelli, M. Ratto, T. Andres, F. Campolongo, J. Cariboni, et al., Global sensitivity analysis, in: The Primer, first ed., Wiley, 2007, http://dx.doi.org/10.1002/9780470725184.

[59] Z. Wang, F. Hutter, M. Zoghi, D. Matheson, N. de Freitas, Bayesian optimization in a billion dimensions via random embeddings, 2016, http://dx.doi.org/10.48550/arXiv.1301.1942.

[60] J. Shlens, A tutorial on principal component analysis, 2014, http://dx.doi.org/10.48550/arXiv.1404.1100, arXiv:1404.1100.

[61] B. Schölkopf, A. Smola, K.-R. Müller, Kernel principal component analysis, in: W. Gerstner, A. Germond, M. Hasler, J.-D. Nicoud (Eds.), Artificial Neural Networks, ICANN'97, in: Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 1997, pp. 583–588, http://dx.doi.org/10.1007/BFb0020217.

[62] D.E. Rumelhart, J.L. McClelland, Learning internal representations by error propagation, in: Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations, MIT Press, 1987, pp. 318–362.

[63] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: International Conference for Learning Representations, 2015, http://dx.doi.org/10.48550/arXiv.1412.6980.

[64] W. Dahmen, A. Reusken, Numerik für Ingenieure und Naturwissenschaftler, second ed., in: Springer-Lehrbuch, Springer, Berlin Heidelberg, 2008.

[65] L.V. Jospin, W. Buntine, F. Boussaid, H. Laga, M. Bennamoun, Hands-on Bayesian neural networks – a tutorial for deep learning users, 2022, arXiv:2007.06823.

[66] G. Van Rossum, F.L. Drake Jr., Python Reference Manual, Centrum voor Wiskunde en Informatica Amsterdam, 1995.

[67] H. Tim, K. Manoj, N. Holger, L. Gilles, S. Iaroslav, Scikit-Optimize/Scikit-Optimize, Zenodo, 2021, http://dx.doi.org/10.5281/ZENODO.5565057.

[68] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, et al., Scikit-learn: Machine learning in Python, J. Mach. Learn. Res. 12 (2011) 2825–2830.

[69] P. Virtanen, R. Gommers, T.E. Oliphant, M. Haberland, T. Reddy, et al., SciPy 1.0: Fundamental algorithms for scientific computing in Python, Nature Methods 17 (2020) 261–272, http://dx.doi.org/10.1038/s41592-019-0686-2.

[70] R.H. Byrd, P. Lu, J. Nocedal, C. Zhu, A limited memory algorithm for bound constrained optimization, SIAM J. Sci. Comput. 16 (1995) 1190–1208, http://dx.doi.org/10.1137/0916069.

[71] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, et al., TensorFlow: Large-scale machine learning on heterogeneous systems, 2015.

[72] J. Hensman, N. Fusi, R. Andrade, N. Durrande, A. Saul, et al., GPy: A Gaussian process framework in Python, 2014.

[73] F. Biscani, D. Izzo, A parallel global multiobjective framework for optimization: Pagmo, J. Open Source Softw. 5 (53) (2020) 2338, http://dx.doi.org/10.21105/joss.02338.

[74] C. Bogoclu, D. Roos, T. Nestorović, Local Latin hypercube refinement for multi-objective design uncertainty optimization, Appl. Soft Comput. 112 (2021) http://dx.doi.org/10.1016/j.asoc.2021.107807.

[75] R.B. Gramacy, H.K.H. Lee, Cases for the nugget in modeling computer experiments, 2010, arXiv:1007.4580.

[76] A. Ajdari, H. Mahlooji, An adaptive exploration-exploitation algorithm for constructing metamodels in random simulation using a novel sequential experimental design, Comm. Statist. Simulation Comput. 43 (5) (2014) 947–968, http://dx.doi.org/10.1080/03610918.2012.720743.

[77] S.K. Mishra, Some new test functions for global optimization and performance of repulsive particle swarm method, SSRN Electr. J. (2006) http://dx.doi.org/10.2139/ssrn.926132.

[78] D.M. Himmelblau, Applied Nonlinear Programming, McGraw-Hill, New York, 1972.

[79] F.H. Branin, Widely convergent method for finding multiple solutions of simultaneous nonlinear equations, IBM J. Res. Dev. 16 (5) (1972) 504–522, http://dx.doi.org/10.1147/rd.165.0504.

[80] J. Contreras, I. Amaya, R. Correa, An improved variant of the conventional harmony search algorithm, Appl. Math. Comput. 227 (2014) 821–830, http://dx.doi.org/10.1016/j.amc.2013.11.050.

[81] T. Ishigami, T. Homma, An importance quantification technique in uncertainty analysis for computer models, in: First International Symposium on Uncertainty Modeling and Analysis, IEEE Comput. Soc. Press, College Park, MD, USA, 1991, pp. 398–403, http://dx.doi.org/10.1109/ISUMA.1990.151285.

[82] J.K. Hartman, Some experiments in global optimization, Nav. Res. Logist. Q. 20 (3) (1973) 569–576, http://dx.doi.org/10.1002/nav.3800200316.

[83] H.H. Rosenbrock, An automatic method for finding the greatest or least value of a function, Comput. J. 3 (3) (1960) 175–184, http://dx.doi.org/10.1093/comjnl/3.3.175.

[84] D.H. Ackley, A Connectionist Machine for Genetic Hillclimbing, in: The Kluwer International Series in Engineering and Computer Science, (no. SECS 28) Kluwer Academic Publishers, Boston, 1987.

[85] Z. Michalewicz, Genetic Algorithms + Data Structures=Evolution Programs, 1992.

[86] H. Schwefel, Numerical Optimization of Computer Models, John Wiley Sons, 1981.

[87] M. Styblinski, T. Tang, Experiments in nonconvex optimization: Stochastic approximation with function smoothing and simulated annealing, Neural Netw. 3 (4) (1990) 467–483.