Empirical Assessment of Deep Gaussian Process Surrogate Models for Engineering Problems

Dushhyanth Rajaram, Tejas G Puranik, S Ashwin Renganathan, Woongje Sung, Olivia Pinon Fischer, Dimitri Mavris, Arun Ramamurthy

Empirical Assessment of Deep Gaussian Process Surrogate Models for Engineering Problems

Dushhyanth Rajaram*, Tejas G. Puranik†
Aerospace Systems Design Laboratory, Georgia Institute of Technology, Atlanta, GA, 30332, U.S.A.

S. Ashwin Renganathan‡
Argonne National Laboratory, Lemont, IL, 60439, U.S.A.

WoongJe Sung†, Olivia Pinon Fischer§, Dimitri N. Mavris¶
Aerospace Systems Design Laboratory, Georgia Institute of Technology, Atlanta, GA, 30332, U.S.A.

Arun Ramamurthy‖
Siemens Corporate Technology, Princeton, NJ, 08540, U.S.A.

In recent years, multi-layered hierarchical compositions of the well-known and widely used Gaussian process models called deep Gaussian processes are finding use in the approximation of black-box functions. In this paper, the performance of deep Gaussian process models is empirically evaluated and compared against the well-established Gaussian process models with a special emphasis on engineering problems. The work draws conclusions through detailed comparisons in terms of metrics such as computational training cost, data requirement, predictive error, and robustness to the choice of the initial design of experiments. Additionally, the viability and robustness of Deep Gaussian process models for applications on practical engineering problems are analyzed through sensitivity to hyperparameters and scalability with respect to the input space dimensionality respectively. Finally, the models are also compared in an adaptive construction setting, where they are built sequentially by selecting points that maximize posterior variance. Experiments are conducted on canonical test functions with varying input dimensions, an engineering test function, and a practical transonic airfoil test case with a high-dimensional input space. The experiments suggest that deep Gaussian process models outperform traditional Gaussian process models in terms of accuracy at the cost of incurring a significantly higher computational expense for the training procedure. The sensitivity studies indicate that inducing points is the most important hyperparameter that affects deep Gaussian process performance and training time. This work empirically shows that deep Gaussian processes are promising candidates for problems that are known to be nonlinear, high-dimensional, and when limited training data is available.

I. Introduction

For complex engineered systems, the design journey from conceptualization to commercialization is an expensive undertaking. It is not uncommon for the design and manufacture of systems such as aircraft/rotorcraft/spacecraft to take 15 to 30 years and cost billions of U.S. dollars. A majority of the cost is associated with building a prototype and testing it in a physical experimental setup. Under this paradigm, most of the knowledge about the product is gained after testing in a physical experiment, at which point making any design changes costs time and money. Another consequence of such a practice is that it limits the number of alternatives that can be evaluated throughout the design process. The advent and growth of supercomputing have helped mitigate this problem to some extent by enabling simulation of the system's physics using high-fidelity (HF) mathematical models. Such models are typically based on conservation

*Senior Graduate Researcher, Daniel Guggenheim School of Aerospace Engineering, Georgia Institute of Technology, AIAA Student Member
†Research Engineer II, Daniel Guggenheim School of Aerospace Engineering, Georgia Institute of Technology, AIAA Member
‡Postdoctoral Fellow, Mathematics and Computer Science Division, Argonne National Laboratory, AIAA Member
§Senior Research Engineer, Daniel Guggenheim School of Aerospace Engineering, Georgia Institute of Technology, AIAA Senior Member
¶S.P. Langley NIA Distinguished Regents Professor and Director of Aerospace Systems Design Laboratory, Daniel Guggenheim School of Aerospace Engineering, Georgia Institute of Technology, AIAA Fellow.
‖Research Scientist, Siemens Corporate Technology

laws and solve a coupled non-linear system of partial differential equations on a discretized spatio-temporal domain. While HF models may offer a cheaper alternative to physical testing, their use is accompanied by many challenges and limitations that must be addressed. Indeed, it is not uncommon for models that accurately capture complex physics to take several days to solve for some design configurations, even on a supercomputer. Since exploration of the design space may involve several such simulations, their use in an inherently iterative process is impractical. A paradigm shift in the existing design process is necessary. If successful, it would result in cheaper, faster and hence more efficient engineering design cycles. A common solution is to replace the actual model with a computationally cheaper, simplified model called a *surrogate* model. Ideally, a surrogate model trades a small amount of accuracy for significantly larger gains in computational efficiency to enable reliable, real-time decision making.

There have been numerous recent studies that apply surrogate modeling techniques on various aerospace engineering problems. Srivastava and Meade [1] have applied machine learning surrogate models in the form of classification techniques for cavity flow and resonance problem. Kumar and Ghosh [2] have used the random forest and decision trees machine learning models for unsteady aerodynamic modeling. Becker et al. [3] have used gaussian process kriging models for uncertainty analysis of a novel remotely piloted airship.

While the use of surrogate models is now commonplace in engineering circles, research in the area of machine learning is evolving at a rapid pace. New kinds of surrogate models meant to address deficiencies in contemporary models are being proposed at an unprecedented rate. Factors such as large input spaces, sufficient availability of data, and nonlinearity of the function of interest are some of the most important challenges with regard to the construction of surrogates for engineering problems. This paper focuses on empirically evaluating the capacity of a nascent surrogate model called the Deep Gaussian Process (DGP) [4] to tackle the aforementioned challenges. This is achieved by constructing surrogate models for emulating canonical test and engineering functions. In order to compare against a widely accepted surrogate model, this study evaluates the performance of DGPs relative to GPs which are considered as the benchmark. GPs have been commonly used in many prior aerospace applications as the surrogate model of choice. For example, Toal and Keane [5] have utilized co-kriging surrogate models for efficient multi-point aerodynamic design optimization problems. Kwon et al. [6] have used dual-level Kriging surrogate models for the robust design optimization of low-noise, coaxial contrarotating rotor. Tang et al. [7] have used kriging surrogate models for unsteady aerodynamic optimization of airfoils. Note that there is significant value in comparing DGPs with other popular surrogate models such as Deep Neural Networks (DNN), polynomial response surfaces, radial basis functions etc. The goal of this work is a thorough preliminary assessment of DGPs relative to GPs. Comparison with other well-known surrogate models is deferred to the future.

In particular, this work demonstrates the benefit of employing DGPs on the routinely encountered problem of predicting aerodynamic quantities of interest (QoI) such as pressure, lift, drag and moment coefficients that capture the aerodynamic performance of an aircraft. These quantities are related to the operating flight conditions and the aerodynamic shape of the aircraft outer mold line (OML). Most flexible shape parametrizations require a large number of parameters. An efficient surrogate model in this context enables the design of an optimal OML given a set of flight conditions. However, the construction of efficient surrogate models for an aerodynamic QoI faces several challenges. First, the landscape of the response can be highly non-linear with strong localized variability. Second, queries to the HF model to generate training data are expensive and hence constrain the affordable size of the training dataset. Finally, the appropriate design of experiments (DOE) to generate the training data is challenging to determine apriori. These challenges necessitate the need for a surrogate model robust to training dataset size and DOE, in addition to having the flexibility to approximate a wide range of input-output relationships.

DGP models are hierarchical, multi-layered generalizations consisting of compositions of Gaussian processes (GPs). Their structure is similar to that of a deep neural network (DNN). As such, they share the stochastic properties of GPs, while also allowing their compositions across layers of DNNs. The suitability of DGPs for non-stationary responses has been demonstrated by Damianou et al [4, 8] and Vafa [9] with both synthetic and real datasets. There have been recent efforts [10] to approximate non-stationary responses using clustering and local gaussian process regressions. Bui et al. [11] showed that DGPs can outperform competing methods (GPs and Bayesian NNs) on select regression problems. DNNs perform very well when a sufficiently large amount of data are available. However, data in aerospace design are typically generated from expensive experiments and hence are typically limited in their size. Salimbeni and Deisenroth [12] have demonstrated on sample regression and classification problems that adding layers (and thereby complexity) to the DGP model did not result in overfitting even with small and medium sized datasets.

The recent study by Hebbal et al. [13] is the only known application of DGPs to aerospace engineering functions. They use a multi-objective expected improvement criterion for Bayesian optimization of a rocket booster using DGPs. However, the benefits and/or detriments of DGPs over more traditional surrogate models have received limited attention

in the context of their application to engineering problems. This work addresses this gap by methodically comparing the performance of DGPs against regular GPs with a focus on training time, training dataset size requirement, and model accuracy among other metrics. Additionally, the scalability of DGPs in higher dimensions and its sensitivity to hyperparameters are also evaluated using canonical test problems. In this regard, limited work currently exists in the literature to the best of the authors' knowledge. This work is intended as one of the few first efforts to empirically evaluate DGPs as a surrogate model for aerospace design problems.

The findings from this research are directly applicable to problems where an expensive black-box must be approximated for repeated evaluations in the many-query context [5–7]. In particular, for instances where data are computationally expensive to generate and therefore sparse while the underlying function is complex and nonlinear, this work finds that DGPs generalize significantly better than conventional GPs (or Kriging models).

The rest of the paper is organized as follows: The background material and theoretical basis of GP and DGP models are reviewed in section II. The experimental test case and associated parametrization are presented in section III. The results are presented on canonical test functions followed by the airfoil test cases in section IV. The paper finally concludes with a summary of the results and an outlook on future work.

II. Overview of Gaussian Process and Deep Gaussian Process Models

This section provides an overview of the theory underpinning GPs and DGPs and discusses some of the challenges associated with the estimation of DGP model parameters.

A. Gaussian Process Regression

Let the inputs and observations of a physical/computer experiment be $\mathbf{x} \in \mathbb{R}^p$ and $y(\mathbf{x}) \in \mathbb{R}$, respectively, where p is the number of input dimensions. Then a GP model assumes that

$$y(\mathbf{x}) = f(\mathbf{x}) + z(\mathbf{x}) + \epsilon, \tag{1}$$

where $z(\mathbf{x}) \sim \mathcal{GP}(0, \sigma^2 r(\cdot))$ is a *zero-mean* GP with constant variance σ^2 and correlation structure $r(\cdot)$, and ϵ is assumed to be a Gaussian white noise. A parametrized function specifying the correlation structure is referred to as the *kernel* function denoted by $k(\cdot, \cdot; \theta))$, where θ are the parameters; we use $r()$ and $k()$ interchangeably through the rest of the paper. The unknown (deterministic) function $f(\mathbf{x})$ can be specified as a parametrized function but a common approach is to integrate it out to form the marginal likelihood distribution given as

$$p(y|\mathbf{x}) = \int p(y|f(\mathbf{x}), \mathbf{x}) p(f(\mathbf{x})) d\mathbf{f}, \tag{2}$$

The conjugacy property of GP models leads to a closed-form expression for the marginal likelihood which makes it amenable to be estimated with conventional optimization techniques. However, this is not possible in DGP models as will be shown in section II.B. One of the key ingredients of GP models is the choice of the kernel. The kernel function encodes information about the relationship between the function values using closeness between points in the input space. While many options exist for the kernel such as matern, rational quadratic, radial basis function (RBF) etc., this work uses the RBF kernel because it is a common choice for capturing the behavior of well-behaved nonlinear functions. For a thorough introduction to the topic, the reader is referred to the textbook by Rasmussen [14].

B. Deep Gaussian Process Regression

A schematic representation of a DGP model with $L - 1$ hidden layers and 1 hidden unit per layer is shown in Figure 1.

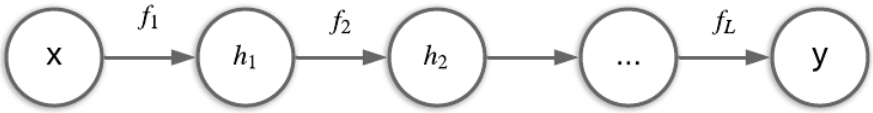


Fig. 1 Schematic of a Deep Gaussian Process (DGP) network.

h_i is the ith hidden layer and is mapped to hidden layer h_{i-1} via a latent function f_i, which is a GP. In other words, each hidden layer is a composition of a GP that maps it to the previous hidden layer establishing the following relationship between the inputs and outputs.

$$y(\mathbf{x}) = f_L\left(f_{L-1}\left(\ldots f_1(\mathbf{x})\right)\right) + \epsilon, \tag{3}$$

Each $f_i \sim \mathcal{GP}(\mu_i, \sigma_i^2 r(\cdot,\cdot;\theta_i))$ with parameters $\{\mu_i, \sigma_i^2, \theta_i\}$ and ϵ is Gaussian white noise. The marginal likelihood for the DGP is obtained by integrating out the hidden layers resulting in

$$p(y|\mathbf{x}) = \int p(y|h_L)p(h_L|h_{L-1})\ldots p(h_1|\mathbf{x})d\mathbf{h}_1\ldots d\mathbf{h}_{L-1} \tag{4}$$

The integral in Equation (4) does not have a closed-form expression similar to the GP model due to the non-linear dependence of the kernels of the probability densities on the hidden layers. Additionally, as the number of hidden layers/units increases, the integral's numerical approximation can quickly become intractable. This is the fundamental challenge associated with the estimation of DGP models compared to regular GP models. Approximate estimation methods for DGP include variational inference (VI) [15] or statistical sampling such as the Markov Chain Monte Carlo (MCMC) method [16].

III. Test Problems

A. Canonical Problems

For initial testing, as discussed in section IV, the Himmelblau and Branin two-dimensional functions are employed whereas the Ackley and Trid functions are used to demonstrate the scalability of DGP models in larger input dimensions. These benchmark functions are commonly available and frequently used to test surrogate model accuracy. This work uses the descriptions and definitions from the Simon Fraser University's Virtual Library of Simulation Experiments*. Details about these test functions are provided in Appendix V. The Output transformerless (OTL) circuit function is used as a candidate engineering canonical problem for testing static DOE cases and adaptive sampling. The OTL circuit function models an output transformerless push-pull circuit. The response V_m is the midpoint voltage. The input consists of six dimensions. The input variables are $[R_{b1}, R_{b2}, R_f, R_{c1}, R_{c2}, \beta]$ and their usual input ranges are given by the following bounds: lower bound = [50, 25, 0.5, 1.2, 0.25, 50], upper bound = [150, 70, 3, 2.5, 1.2, 300]. The analytical form of this function is given by

$$V_m(\mathbf{x}) = \frac{(V_{b1} + 0.74)\ \beta\ (R_{c2}+9)}{\beta(R_{c2}+9) + R_f} + \frac{11.35R_f}{\beta(R_{c2}+9) + R_f} + \frac{0.74R_f\ \beta\ (R_{c2}+9)}{(\beta(R_{c2}+9) + R_f)R_{c1}} \ldots V_{b1} = \frac{12R_{b2}}{R_{b1} + R_{b2}} \tag{5}$$

B. Practical Application Problem

One of the main goals of this work is to construct surrogate models of the relationship between aerodynamic QoI, the operating flight conditions and the OML shape parameters of an aerodynamic object, in this case an airfoil. Doing so first requires parameterizing the airfoil shape, which is non-trivial, specifically due to the existence of sensitive regions such as the rounded leading edge (that could lead to infinite slope) and the sharp trailing edge (which could lead to a discontinuity). Consequently, a parsimonius and smooth parameterization with sufficient flexilibity is preferred.

To test and demonstrate the performance of DGPs for problems with a large number of input design parameters, a 2-D airfoil shape design test problem is chosen. The shape of the airfoil is parameterized by the mean camber position and thickness at 22 longitudinal positions. This information is used to define a total of 42 (the size of the input space) points on an airfoil; 20 upper, 20 lower, leading edge and trailing edge points. The 22 fixed chord-wise coordinates are clustered near both leading and trailing edges. For each airfoil shape, the non-dimensional lift and drag coefficients are calculated using XFOIL [17]. A total of over 600 airfoil shapes are evaluated at Mach number 0.75, Reynolds number 6×10^6 and angle of attack 0^o. The non-dimensional lift and drag coefficients are available to train and tune the surrogate models. The total variation of the airfoil among all the 600 shapes used in this work is shown in Figure 2 with the corresponding variations in lift and drag coefficients. Note that there are other methods which can also be used to parameterize airfoils such as those by Ghosh et al. [18], or class shape transofrmations (CST) [19, 20]. A

*https://www.sfu.ca/~ssurjano/optimization.html

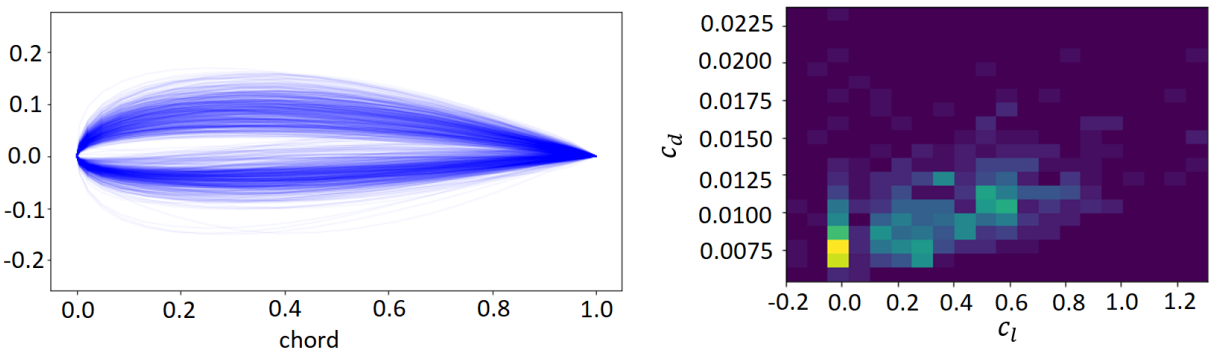


Fig. 2 Variation of airfoil shape (left) and lift/drag coefficients (right) in data set (M = 0.75, Re = 10^6, AoA = 0^o).

similar setup has been used in previous work [21, 22] for fitting a neural network models for predicting airfoil lift coefficient and other properties. Similarly, Liu et al. [23] have used multiresponse surfaces for airfoil design with multiple-output-Gaussian-process-regression model and obtained higher accuracies than single output GP regression for airfoil design problems.

IV. Implementation and Results

This section reports the results obtained from the application of DGP to various canonical problems using a static design of experiments. A comparison of GP and DGP in an adaptive sampling context is provided for the canonical engineering problem and the aerodynamic test case outlined in section III. In each case, the comparison is made between DGP and traditional GP using the same set of training points and the same kernel function. Additionally, the computational cost in the form of training time (wall time) is also compared for both sets of models. Before proceeding, it is worth emphasizing the progression and types of experiments performed in this study (please refer to Figure 3). First, the implementations of both GP and DGP are tested through preliminary experiments on canonical problems (the Himmelblau and Branin functions) in terms of predictive accuracy and training time. Second, the scalability (with increasing input space dimensionality) and robustness (with respect to the choice of hyperparameters) of DGPs are evaluated on the scalable Ackley canonical function and the Himmelblau and Branin test functions respectively. Third, GPs and DGPs are compared with respect to their predictive accuracy and training time using a static DoE for both the OTL circuit problem and the airfoil problem. This is referred to as the static DoE case wherein the idea is to compare performance on a given dataset without the ability to query the underlying function to refine the model. Finally, for both the canonical engineering (OTL circuit) problem and the practical airfoil problem, GPs and DGPs are compared in an adaptive setting. Starting with a model trained on an initial DoE, the design space is sequentially queried by optimizing a criterion that maximizes the information gained about the function. The idea is to test the relative performance under a constraint on the budget of total true function evaluations.

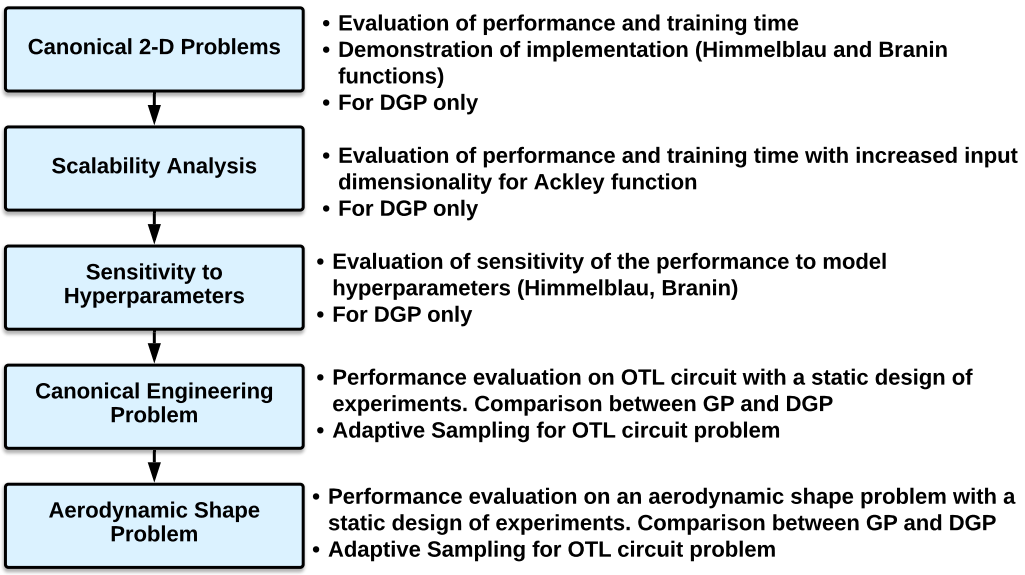


Fig. 3 Progression of experiments for DGP evaluation demonstrated in the paper.

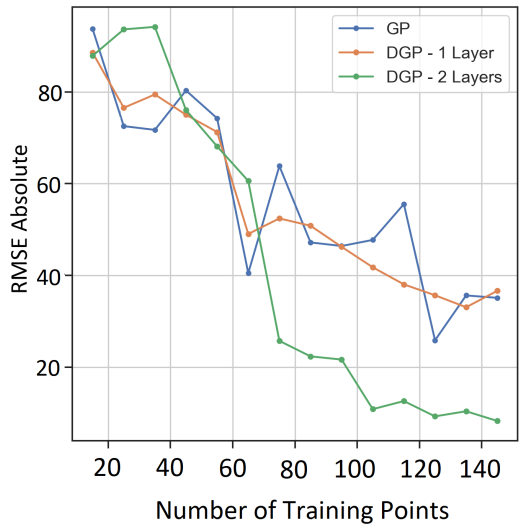
There are various publicly available implementations of DGPs and GPs under active development that are being continuously improved. These libraries allow for models to be trained given input-output pairs and modification of the source code, if needed. For GPs, this work utilizes the GPy library developed by the Sheffield Machine Learning Group[†]. The algorithm implemented in GPy trains the model using a gradient-based optimizer to minimize the negative log-marginal likelihood to obtain an maximum likelihood estimate of the kernel parameters. For DGPs, PyDeepGP library developed by the Sheffield Machine Learning Group is used due to its popularity[‡]. The models are trained using approximate variational inference. This implementation contains three main hyperparameters – the number of hidden layers, the number of hidden units (called latent units) per layer, and the number of inducing points. Unlike DNN models or other models with deep architectures, between 1 and 5 hidden layers are usually sufficient for DGPs. Moreover, beyond this range it was found that the training procedure became unwieldy in terms of memory requirements and computational complexity. The number of latent units also varies from 1 to 5 in previous implementations [4, 9, 11]. The number of inducing points affects the computational cost of the training procedure significantly by reducing the number of points used in the computation of the pairwise covariance function. Since this leads to an approximation of the covariance matrix, the reduction in computational complexity may result in a loss in predicitve accuracy of the model. There is no clear guidance for choosing the appropriate number of inducing points [12]. In practice, it is recommended that the number of inducing points be at least equal to 25% of the number of training points. In general, a higher number of inducing points increases model quality at the cost of additional computational time. It is empirically observed that for functions with a small number of inputs (e.g. 2 inputs), the number of inducing points must be at least 50-75% of the number of training points. A smaller number of inducing points results in poorly converging training optimization runs [4]. Through grid search and trial-and-error on the bounds, the number of inducing points in this work is varied between 0.25 and 2 times the size of the training dataset whereas both the number of layers and latent units is chosen as 1 or 2. Note that a DGP with one hidden layer is different than a GP as it still contains a composition of GPs. A DGP with zero hidden layers is equivalent to a GP. The following sections describe the results obtained from these experiments and their implication on the utility of DGP models.

A. Canonical 2-D Problems

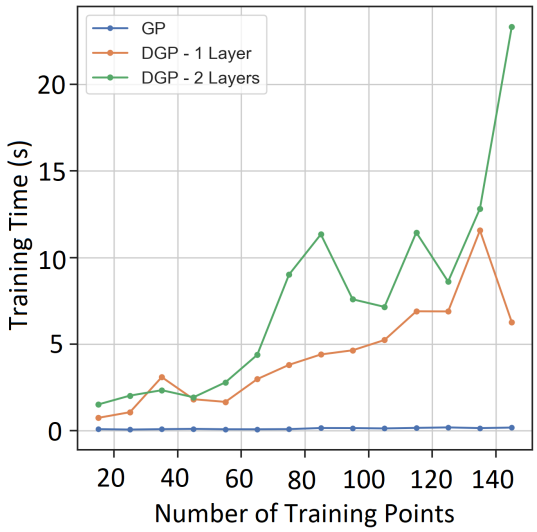
The first set of experiments involves training DGPs on popular regression benchmarks for 2-D functions namely the Himmelblau and the Branin test functions. More details about these functions can be found in Appendix V. For each of these functions, GP and DGP (1-layer and 2-layer variants) models are trained for an increasing number of training points (each obtained using a static latin hypercube design of experiments). In each case, the trained models are tested

[†]https://github.com/SheffieldML/GPy
[‡]https://github.com/SheffieldML/PyDeepGP

on a set of one hundred test points in terms of the absolute root mean square error. The reported errors are averaged over ten training repetitions to account for the inherent stochasticity. The computational cost is measured in terms of wall time for the training procedure.



(a) Variation of the averaged absolute RMS error with increasing number of training points.



(b) Variation of the averaged training time with increasing number of training points.

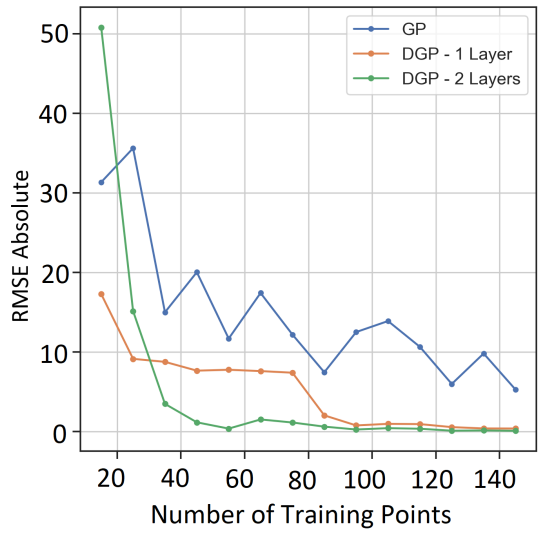
Fig. 4 RMS error and Training time comparisons between GP and DGP for Himmelblau problem.

1. Himmelblau Function

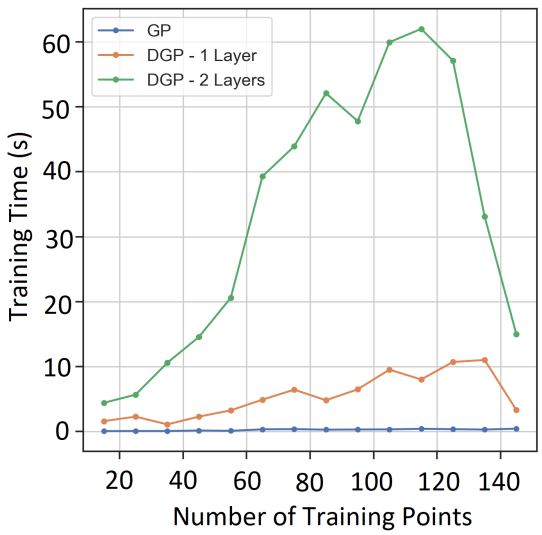
As shown in Figures 4a and 4b, as the number of training points increases, the DGP models with 2 layers perform better than the GP and DGP with 1 layer for the Himmelblau function. The training cost for both variants of DGP models is higher than that of the GPs, which remains relatively constant for this problem even as the number of training points increases considerably. The overall difference in the performance between GP and DGP is marginal (especially with a single hidden layer).

2. Branin Function

For the Branin function, as seen from Figures 5a and 5b, both DGP models exhibit superior averaged absolute RMS error than GP models but perform worse than GP in terms of training time with the 2-layer DGP performing considerably worse than the 1-layer DGP. The Branin function assumes values between 0 and 300 in its domain. Even with largest number of training points, absolute RMS errors ($\approx$10) for the GP models are significantly larger than the errors for DGP models ($\approx$0). As the number of training points increases, both variants of DGP perform very well and are almost indistinguishable from each other indicating diminishing returns in predictive accuracy. The training time however gets progressively worse as the number of training points increases; a general trend that is observed in this study as well as in work published in literature. One can note that, for the Branin function, there is a drop in training time for DGP models beyond 120 training points. A similar behavior is observed only for the 1-layer variant for the Himmelblau function. This could be related to the number of inducing points used in this experiment (120). In later sections, the effects resulting from changing the number of inducing points are investigated.



(a) Variation of the averaged absolute RMS error with increasing number of training points.



(b) Variation of the averaged training time with increasing number of training points.

Fig. 5 RMS error and Training time comparisons between GP and DGP for Branin problem.

It is evident from the results for both of these canonical problems that DGP models indicate promise in terms of their superior predicitve performance over the more traditional GP models, albeit at the cost of higher training time. Therefore, it is important to explore and understand the scalability of DGP models as well as their sensitivity to model hyperparameters in order to use them in the most efficient manner.

B. Scalability Analysis

The next set of experiments measures how DGP models scale with an increasing number of input dimensions. The scalability of GPs has been a subject of prior research (Ghosh et al. [24], Eriksson et al. [25], Wilson et al. [26]). However, there has been limited research on scalability of DGPs. The scalability of the method with an increasing number of input dimensions is an important indicator of its applicability to large, practical engineering design problems. For this purpose, the experiments are performed on a parametrically scalable n-dimensional canonical test function called the Ackley function. The number of dimensions can be increased from 2 to n. For a fixed number of training points (100), DGP models are trained with various hyperparameters (hidden layers, latent units, and inducing points) and an increasing number of dimensions. The measurement of the RMS error is normalized for this particular experiment to enable a fair comparison because the absolute values of the function response increase with the number of input dimensions. The average training wall time is also compared for each case to understand how the training time scales with dimensions.

Figure 6 illustrates the normalized RMS error for the validation points with increasing dimensionality of the Ackley function. One can observe that the normalized error decreases with larger dimensionality across all hyperparameter settings. In other words, the difference between the trends and values of the normalized error for the different hyperparameter settings is insignificant. It is worth noting that the decrease in normalized RMSE as the number of input dimensions increases is also dependent on the function in question. The observed trend with the Ackley function in Figure 6 is a result of the characteristics of its analytical form. As the number of dimensions increases the function values also increase. In the computation of the normalized RMSE, while the difference between the predicted and actual function values increases with the number of inputs, the denominator increases faster. Consequently, the normalized RMSE decreases. This trend is also observed with a GP.

Figure 7 shows that the training time remains reasonably constant for a particular number of inducing points as the number of dimensions increases for the Ackley function. This behavior is expected because for GPs and DGPs, the training cost scales strongly with the number of training points rather than with the number of input dimensions. It is important to note that depending on the kernel, the number of training parameters for the model can increase with the number of input dimensions. However, since this work utilizes the isotropic RBF kernel, which has a shared common length scale for all input dimensions, the number of trainable parameters does not increase as the number

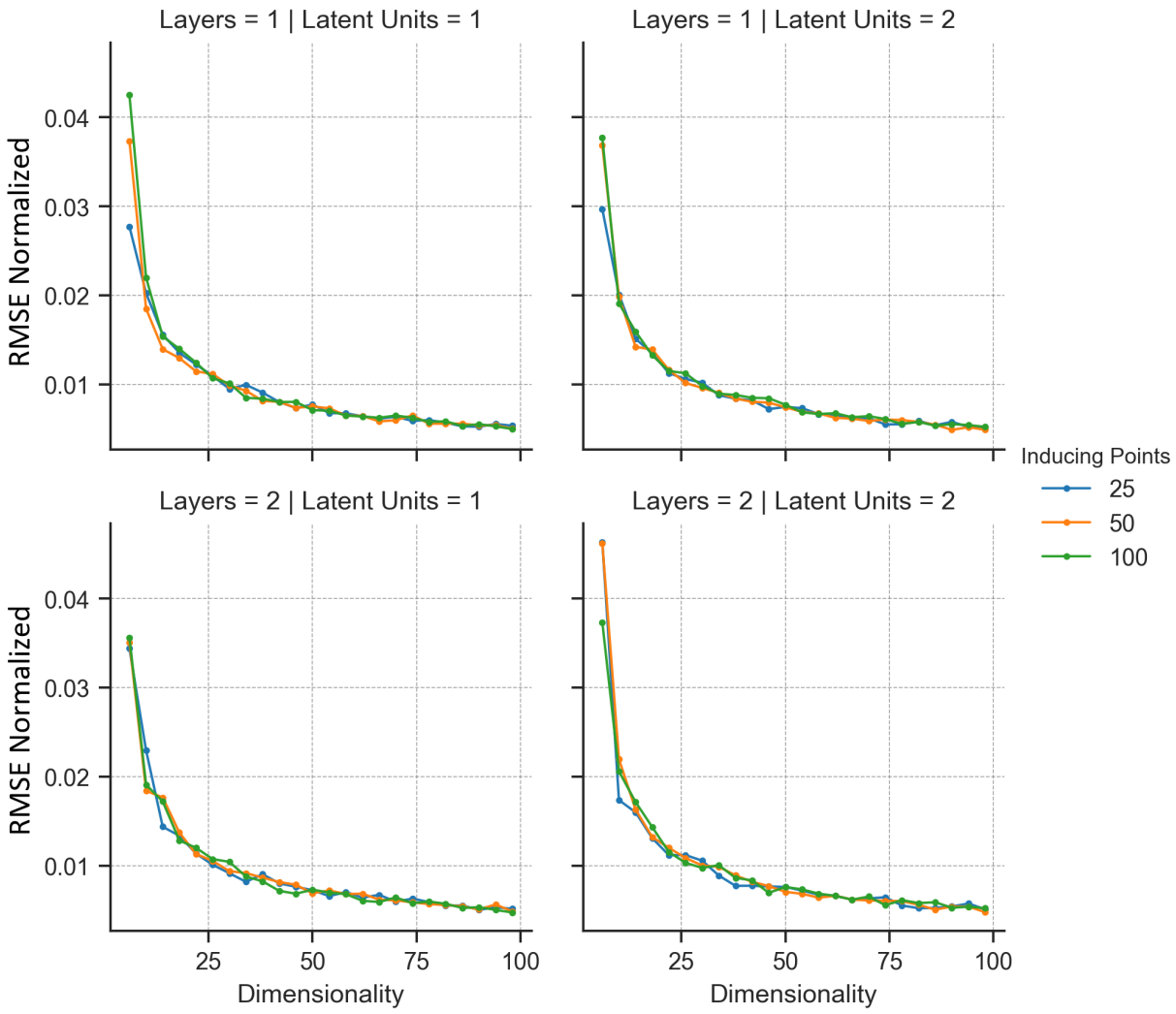


Fig. 6 Root-mean-square validation error for DGP model of Ackley function with increasing dimensionality.

of inputs increases. Thus, for a fixed set of hyperparameters and fixed number of training points, the training time remains relatively constant as the number of input dimensions increase. Reading from top to bottom in Figure 7, one can observe that the training time for DGP models increases slightly between one- and two-layer variants when all other hyperparameters are held constant. As the number of hidden layers increases, the training time is expected to increase because the models become increasingly complex due to the addition of GPs to the function composition that results in a DGP. The effect of the number of inducing points on the training time is similar to that of the relationship between the training time and the number of training points. The computation of pairwise covariances and inversion of the dense covariance matrix are the main contributors to the training cost. The computations for pairwise covariance strongly depend on the number of inducing points considered. Therefore, an increase in the number of inducing points directly reflects in an increase in the training time.

Indeed, an interesting trend observed for the test problem to which DGP has been applied is that higher dimensional problems had a lower normalized error compared to lower dimensional problems; evidence indicating the possibility that DGP is a suitable candidate model for higher dimensional problems. For both functions in this scalability test, it is observed that the training time is reasonably independent of the hyperparameters (other than the number of inducing points, which has a slight effect on the training time).

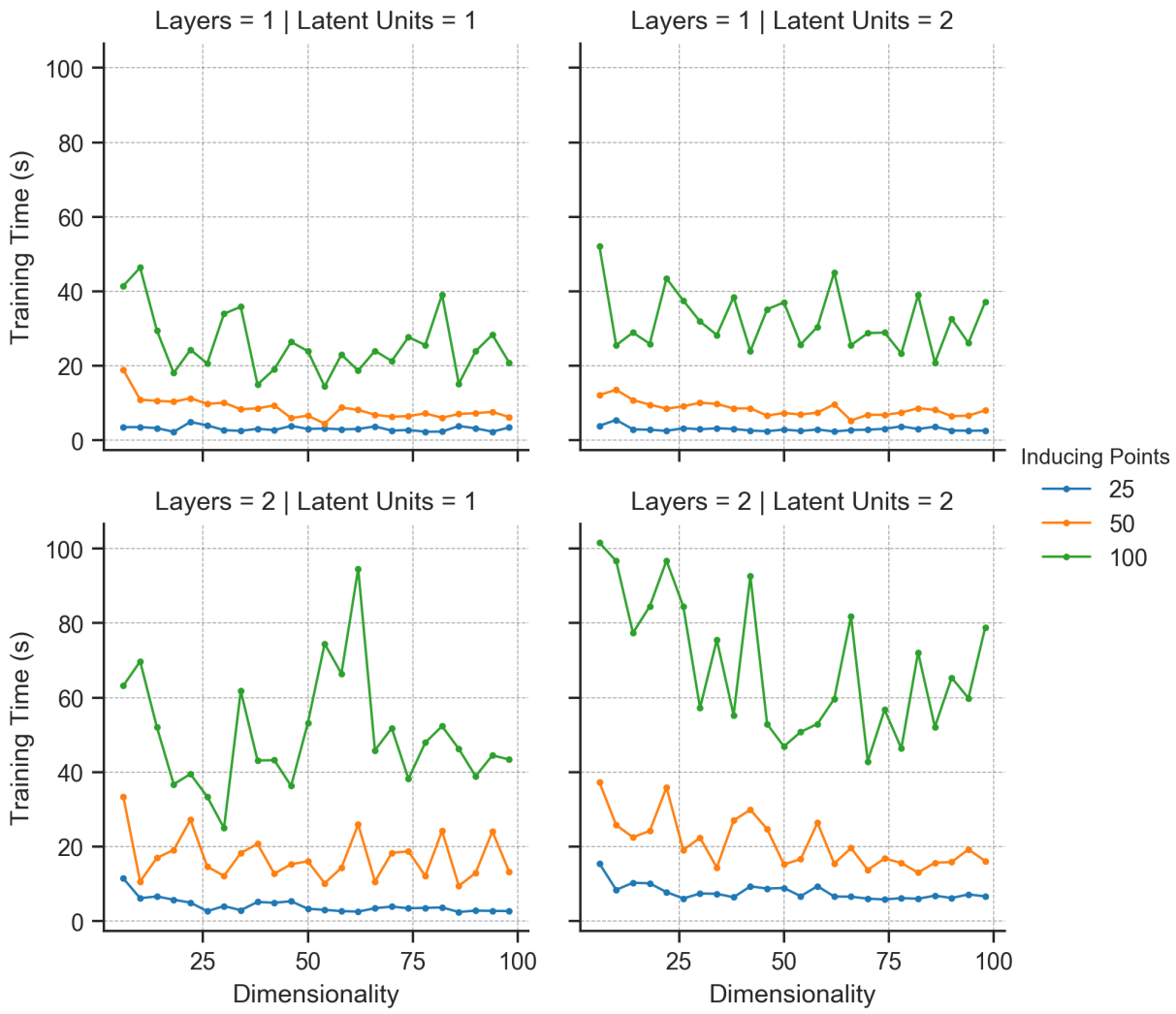


Fig. 7 Scalability of DGP models as tested using the Ackley function with increasing dimensionality.

C. Sensitivity to Hyperparameters

While DGP models are demonstrated to be scalable to higher dimensions, they are sensitive to the choice of hyperparameters. For a DGP model, the main hyperparameters affecting the model are the number of hidden layers, the number of units, and the number of inducing points (used in the inference). The combination of these parameters can have an impact on the accuracy and applicability of the method. Therefore, in this section, a methodical approach is undertaken to quantify the sensitivity of the DGP models with respect to these hyperparameters using the canonical functions presented in the section above. In each case, DGP models are trained for the respective function using two different number of training points (50 and 100). For each of the training point sets, a sweep of the hyperparameter space in the form of a design of experiments (DOE) is conducted along with multiple repetitions at each level. The results obtained from this study are plotted in the Figures 8 and 9. Table 1 outlines the DOE that is executed for the sensitivity analysis. Thus, for each test function, a total of 256 DGP models are trained and validated.

Table 1 Details of the design-of-experiments for hyperparameter sensitivity analysis

Parameter	Options / Levels
Number of Hidden Layers	[1, 2]
Number of Latent Units	[1, 2]
Number of Inducing Points	[75, 125, 175, 225]
Number of Training Points	[50, 100]
Number of Repetitions	8

Figure 8 indicates the variability in the validation error for the Himmelblau function using 100 training points which varies for different hyperparameter settings. Overall, with a single hidden layer, two latent units generally produce better models than one latent unit for all inducing point settings. This trend is interestingly reversed in the two hidden layer models where models with one latent unit outperforms models with two latent units. There is no clear advantage of using more inducing points for this problem within the range of values experimented. It is worth mentioning that the absolute RMS for the best DGP model with 100 training points (which was the model with 2 hidden layers in Figure 4a) is approximately equal to 18 (lower end of the box plots shown in Figure 8). It is generally observed that models with two hidden layers are better suited for the Himmelblau test function.

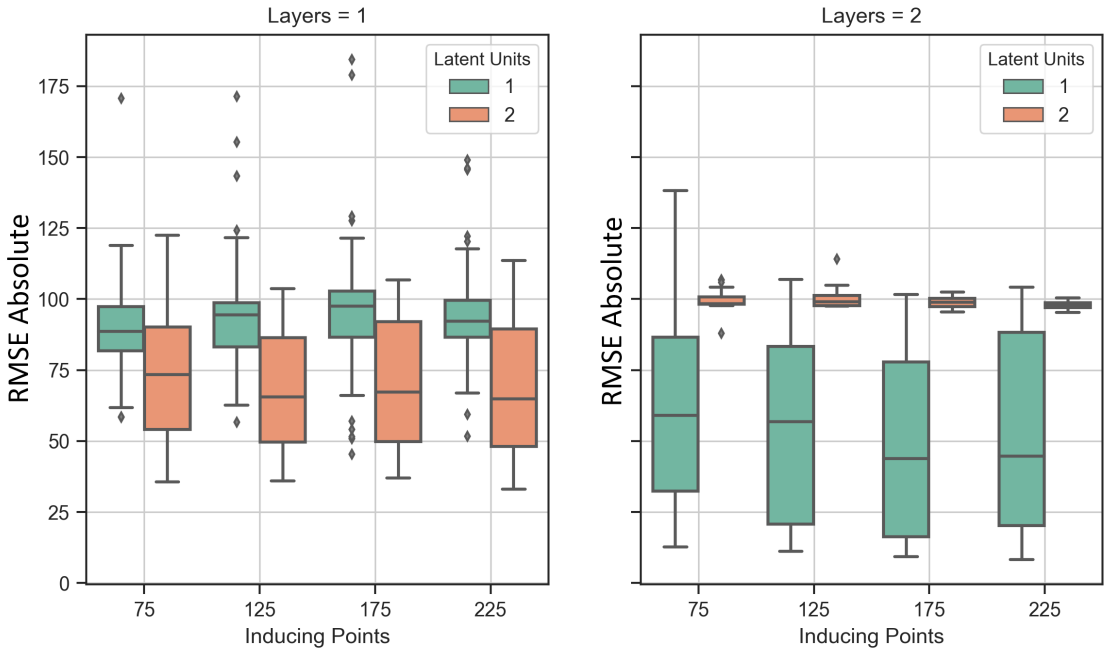


Fig. 8 Sensitivity analysis for the Himmelblau Test function using 100 training points.

For the Branin test function, as observed in Figure 9, the error is slightly higher for all models with a single hidden layer than those with two hidden layers. Models with single latent units, in particular, perform poorly for one hidden layer models. Models with higher number of inducing points generally have a tighter distribution in the error for the two-layer variant but the high number of inducing points does not provide any advantage for this problem. Note that the absolute RMS error for the best DGP model with 100 training points (which was the model with both 1 and 2 hidden layers in Figure 5a) is approximately equal to 0.02 (lower end of the box plots shown in Figure 9).

An important observation made during these sensitivity experiments is that the number of inducing points seems to have the most significant impact on the ability to obtain a good DGP model. If the number of inducing points is insufficient, the model tends to underfit the data regardless of the number of training points. Whereas if number of the inducing points increase beyond a certain threshold, the training time starts increasing rapidly and the performance does not necessarily improve proportionally. Higher dimensional problems generally performed better with higher number of

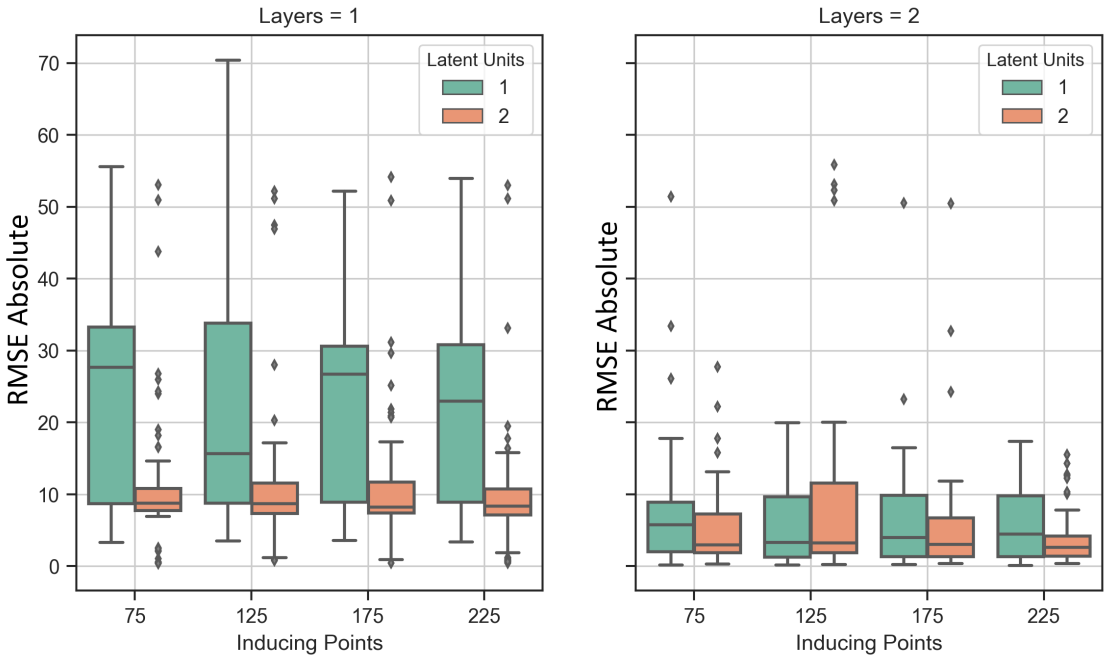


Fig. 9 Sensitivity Analysis for the Branin Test function using 100 training points.

inducing points. No other obvious trends could be observed across the different problems to inform the selection of the optimal number of inducing points to be used.

As mentioned before, the number of hidden layers and latent units was tested only at two values, 1 and 2, based on previous published literature related to DGP. Among the four resulting combinations of hidden layers and latent units ([1, 1], [1, 2], [2, 1], [2, 2]), both extremes ([1, 1], [2, 2]) typically perform relatively worse. For the lower dimensional canonical problem, models with a single latent unit and single hidden layer had an erratic behavior in the validation error performance. The trends in performance of the best model for a given number of training points is similar across all four combinations of hidden layers and latent units. To summarize the hyperparameter sensitivity study, the experiments only indicate that the performance of DGP is non-trivially dependent on the hyperparameter settings. However, the nature and behavior of this dependency is unclear. Examining the changes one hyperparameter at a time does not yield any interesting insight either.

D. Canonical Engineering Problem

In this section, the predictive performance of DGP models is demonstrated through application on a canonical engineering problem called the OTL circuit function. The function models an output transformerless push-pull circuit and consists of one output and six input dimensions. For this problem, multiple DGP and GP models are trained using a set of training points and the root mean square error on a completely different set of one thousand test points is evaluated. The number of training points is successively increased from 25 up to a maximum of 95. For each set of the training points, GP and DGP models are trained multiple times to account for stochasticity in the training process. Additionally, for DGP models, as there are multiple choices available for hyperparameter settings, each DGP model is trained for *eight* hyperparameter settings (full-factorial combinations of $n_{layers} = [1, 2]$, $n_{latent} = [1, 2]$, $n_{inducing} = [75, 150]$).

1. Static DOE for the Canonical Engineering Problem

Figures 10 and 11 show the variation of RMS error and training time required against the number of training points respectively. Because there are multiple models being trained (repetitions for GP, repetitions and hyperparameter choices for DGP), the average error and training time are computed at each training point setting. The shaded region represents the spread of values obtained at a training point setting for different models. The larger spread among the errors for DGP models is evident and expected because, as seen before, the different hyperparameter settings can

significantly affect performance of the model. For each training point setting in this 6-dimensional problem, DGP models unequivocally perform better than GP models throughout. Moreover, the overall trend for both models shows that the performance improves and becomes more consistent as the number of training points increases. In fact, it is relatively more pronounced for DGP than GP.

In terms of training time, it is observed that DGP models have a significantly worse computational performance than GP models for the same number of training points. The average training time does not increase noticeably for DGP for higher number of training points whereas it increases by an order of magnitude for GP models. These results indicate that, for the canonical engineering problem, the cost of DGP models is higher throughout than that of GP models. Table 2 shows the results of the best model obtained at each training point setting for each of the two types of surrogate models (DGP and GP) along with the percentage difference between the two (measured as $\left(\frac{GP-DGP}{GP}\right) \times 100$).

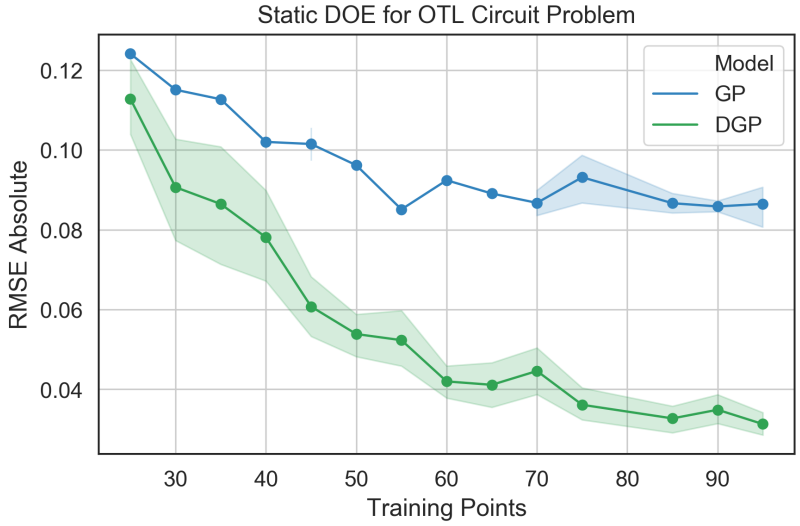


Fig. 10 Average RMS error for each surrogate model with increasing number of training points.

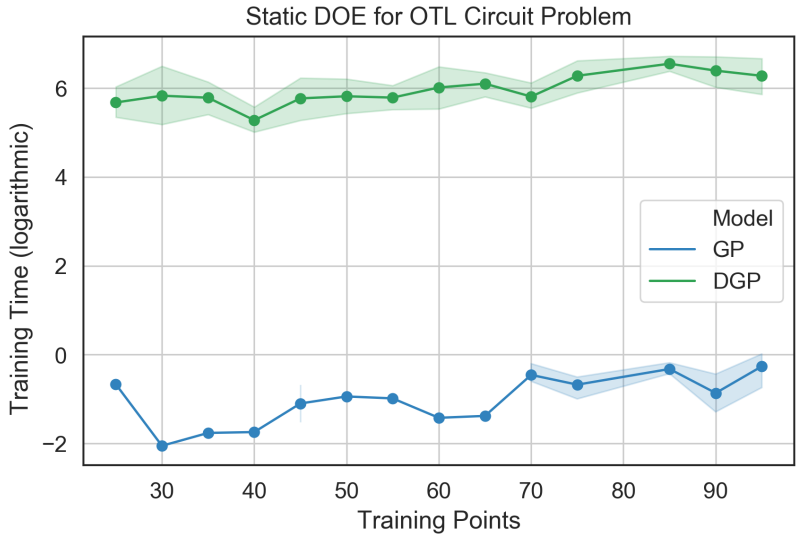


Fig. 11 Average training time required for each surrogate model on a natural log scale with increasing number of training points.

Table 2 Summary of results on OTL circuit problem with static DOE

OTL Circuit Problem						
Training Points	RMSE GP	RMSE DGP	RMSE Difference	Training Time GP (s)	Training Time DGP (s)	Training Time Ratio (GP / DGP)
25	0.124	0.094	24%	0.514	159.2	3×10^{-3}
30	0.115	0.056	51%	0.341	80.06	4×10^{-3}
35	0.112	0.052	53%	0.248	147.7	1×10^{-3}
40	0.102	0.056	45%	0.286	114.1	2×10^{-3}
45	0.097	0.046	52%	0.506	77.88	6×10^{-3}
50	0.096	0.040	58%	0.391	160.3	2×10^{-3}
55	0.085	0.044	48%	0.443	163.5	2×10^{-3}
60	0.092	0.033	64%	0.423	173.6	2×10^{-3}
65	0.089	0.025	72%	0.278	216.6	1×10^{-3}
70	0.083	0.032	61%	0.817	231.7	3×10^{-3}
75	0.086	0.031	64%	0.592	210.6	2×10^{-3}
80	0.082	0.032	61%	0.982	180.3	5×10^{-3}
85	0.083	0.023	72%	0.887	474.7	1×10^{-3}
90	0.084	0.027	68%	0.645	310.7	2×10^{-3}
95	0.088	0.025	72%	1.016	283.9	3×10^{-3}

As is evident from Table 2, the best performing DGP model is better than the best performing GP model but is computationally much more expensive than the corresponding GP model. This presents an interesting trade-off that will eventually determine which model ought to be used in a particular setting.

One of the drawbacks of using a static DOE to train the surrogate models is that information about the training points already available is not intelligently utilized in training newer models. Ideally, several static DOEs need to be constructed and the model must be trained multiple times in order to obtain a model that performs with satisfactory predictive accuracy. This process is computationally expensive and does not necessarily guarantee a good model consistently. One of the solutions for overcoming some of these limitations is through an adaptive sampling strategy that intelligently identifies the set of training points to be sampled.

2. Adaptive Sampling for the Canonical Engineering Problem

While static designs of experiments are well suited for a given set of pre-evaluated designs, adaptive strategies provide a principled feed-back procedure to recommend new experiments serially as new information about the underlying function's landscape becomes available. In this section, the value of DGPs in comparison to GPs is assessed in closed-loop or adaptive scenarios. However, since closed-loop strategies are inherently serial in nature, the inability to parallelize the evaluation of the function to be emulated is a major drawback.

Recommendations for new designs to be evaluated are obtained by optimizing some criterion (defined over the design space) that maximizes the information that can be gained about the function. Several criteria have been proposed and assessed for use in closed-loop settings [27–29]. In this paper, the *maximum variance design* [27] criterion is maximized over the design space in order to guide the search. The criterion capitalizes on the idea that the best location to sample is the one with the largest uncertainty in the prediction. For GPs and DGPs, the *maximum variance design* is obtained by maximizing the posterior variance (conditioned on the observed data). Although a closed-form expression is not available for DGPs (due to the composition of multiple GPs), the approximate variational inference allows for the evaluation of the posterior variance. The expression for posterior variance for GPs is given by

$$s^2(\mathbf{x}) = k(\mathbf{x}^*, \mathbf{x}^*) - \mathbf{k}_*^T M^{-1} \mathbf{k}_*, \quad \text{where} \quad M = K + \sigma_n^2 I \tag{6}$$

In the expression above, $k(\cdot,\cdot)$ is the covariance function, $\mathbf{x}_*$ is the test point where the posterior variance must be evaluated, and K is the matrix containing pairwise covariances computed for the points observed so far.

The performance of adaptive sampling using GP and DGP is demonstrated on the OTL circuit canonical engineering problem and the aerodynamic shape problem. Beginning with a GP/DGP trained on a static latin hypercube design with a pre-specified number of points, the procedure proceeds by maximizing the posterior variance to yield the next design point to be evaluated and added to the training set. The models are then updated with the new observed data point by warm-starting the training algorithm with the parameter values of the current model. This process is repeated until a specified number of training points is added to both the GP and the DGP models. In order to avoid getting trapped in local optima, this work employs the meta-heuristic particle swarm optimization algorithm [30] to optimize the posterior variance. The adaptive sampling strategy highlighted in Figure 12 is first applied to the canonical OTL

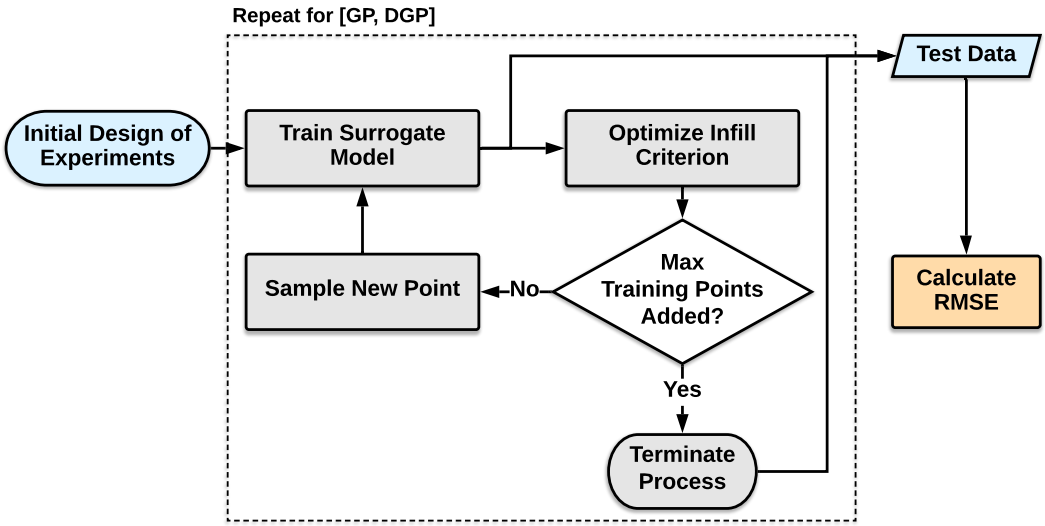


Fig. 12 Adaptive Sampling Methodology.

circuit engineering problem. For this particular problem, an initial candidate model at a particular set of training points is chosen for each of the two surrogate models. In order to have a fair comparison across models, the model structure (hyperparameters, kernel, etc.) is fixed. New points are adaptively added as outlined earlier and the RMS error on the same test set as the static DOE is calculated. The main aim of the adaptive sampling approach is to highlight the possibility of starting from a static model with lower number of training points and sequentially improving it with each additional point. This is especially useful for scenarios when there is a limited budget available for obtaining additional data points. Figure 13 shows the results obtained from applying the adaptive sampling on the OTL circuit problem. In this experiment, a model obtained from the static DOE using 30 initial training points is chosen as a start and 30 more training points are adaptively added while re-training the model after every addition. The progression of the RMS error with each added training point is shown in Figure 13 using the solid green (DGP) and blue (GP) lines. The dotted green (DGP) and blue (GP) lines represent the average RMSE for the initial static models with 30 points. Similarly, the dashed green line (DGP) and the dashed blue line (GP) represent the average RMSE for the static models with 60 training points. These dotted and dashed lines represent average errors with no adaptive sampling (static DOEs). As is evident from Figure 13, the performances of the GP and DGP models steadily improve with successive addition of training points until both models perform better than the static models with the same number of training points. For the static models, the performance represented by dashed lines is obtained following a search across multiple hyperparameters and repetitions requiring significant computational resources. One should note that since the best hyperparameter setting is unknown a priori in static DOE schemes, several models must be trained to search the hyperparameter space. For adaptive sampling, the same model is re-trained with the added points and therefore is computationally much more efficient. In some sense, the freedom in the ability to select sampling points for a given set of hyperparameters reduces the need to search the hyperparameter space.

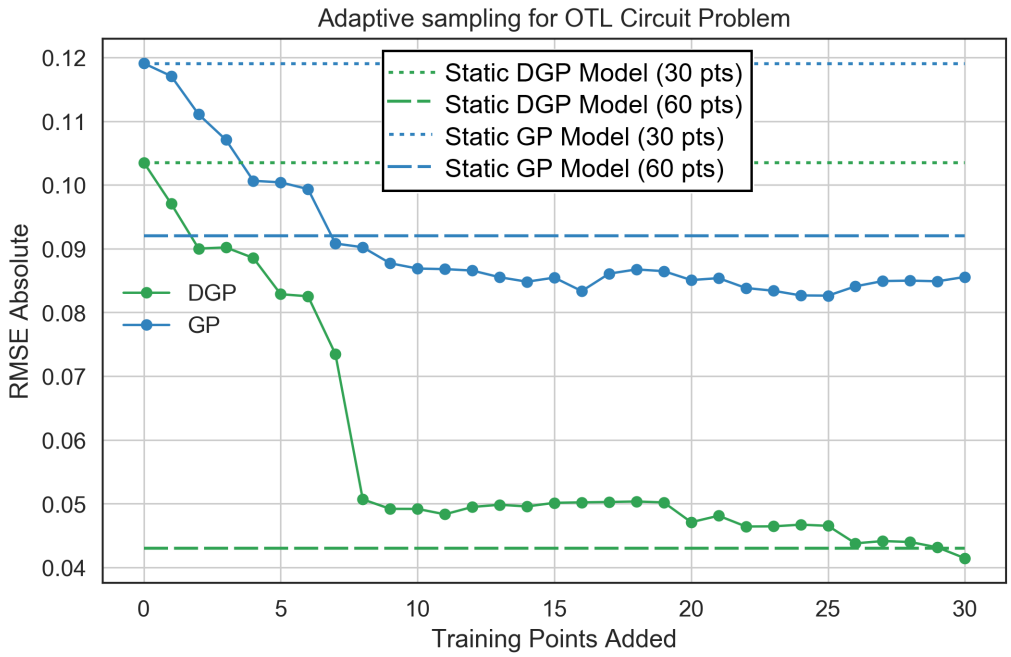


Fig. 13 Adaptive sampling for DGP and GP models using 30 initial training points.

E. Aerodynamic Shape Problem

Using the knowledge from the canonical test problems and engineering problem, DGP models are trained on the airfoil aerodynamics problem described in section III. For this problem, parameterized airfoils are constructed using 42-dimensional design vectors and the flow properties are evaluated using XFoil. For the purpose of demonstration in this paper, the non-dimensional lift coefficient (c_l) and non-dimensional drag coefficient (c_d) of the airfoil are chosen as the output quantities of interest. Both GP and DGP models are successively trained using an increasing number of airfoil shapes in the training set and the error is evaluated on an independent test set of a thousand airfoil shape outputs. Similar to the OTL circuit problem, each DGP model is trained for *eight* hyperparameter settings (full-factorial combinations of n_{layers} = [1, 2], n_{latent} = [1, 2], $n_{inducing}$ = [75, 150]).

1. Static DOE for Airfoil Lift Coefficient Problem

Figures 14 and 15 show respectively, the RMS error and training time required for each of the models at each training point setting for the airfoil lift coefficient problem. One can observe that, the DGP models afford better accuracy but poorer computational performance. Note that for both GP and DGP there is some spread observed in the errors at each training point setting. The performance generally improves as the number of training points increases.

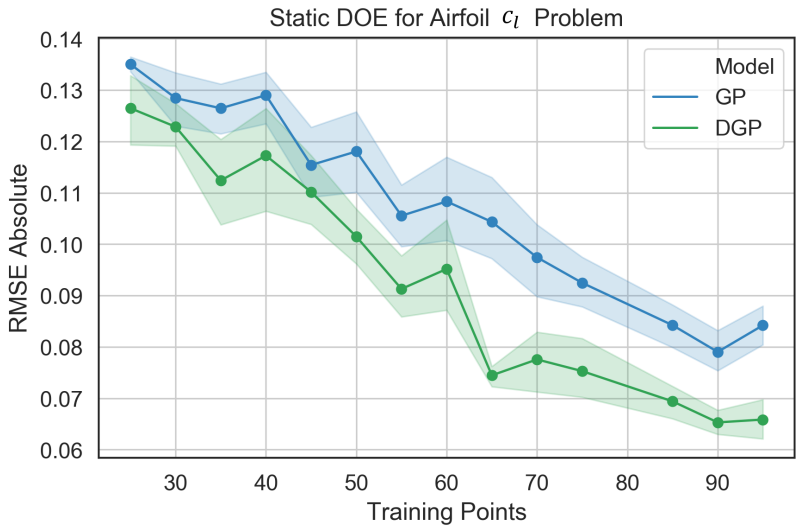


Fig. 14 Average RMS error for each surrogate model with increasing number of training points for the airfoil lift coefficient problem.

The computational cost of the GP and DGP models for this 42-dimensional problem is similar to that of the lower dimensional OTL circuit problem and remains reasonably constant as the number of training points increase. It can also be seen that GP models perform significantly better than DGP models in terms of computational cost.

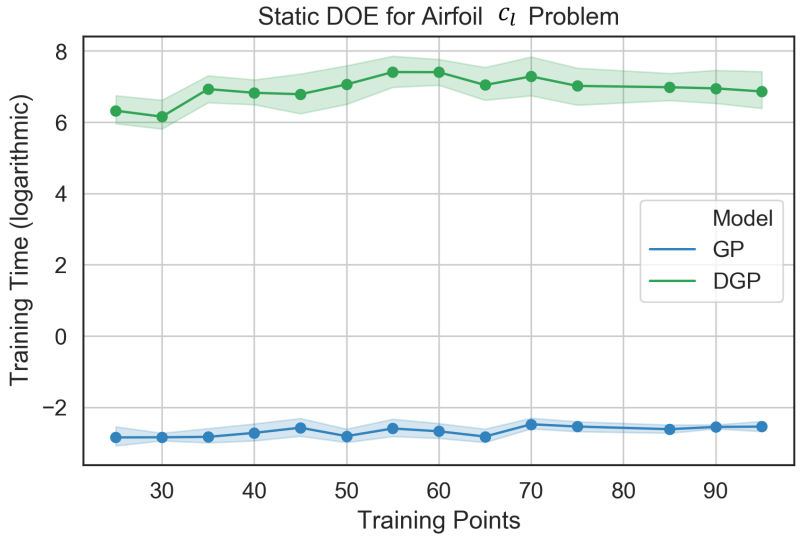


Fig. 15 Average training time for each surrogate model with increasing number of training points for the airfoil lift coefficient problem.

Table 3 shows the results of the best model obtained at each training point setting for each of the two types of surrogate models (DGP and GP) along with the percentage difference between the two (measured as $\left(\frac{GP-DGP}{GP}\right) \times 100$).

Table 3 Summary of results on airfoil lift coefficient problem with static DOE

Airfoil c_l Problem						
Training Points	RMSE GP	RMSE DGP	RMSE Difference	Training Time GP (s)	Training Time DGP (s)	Training Time Ratio (GP / DGP)
25	0.133	0.103	23%	0.132	274.9	4×10^{-4}
30	0.117	0.113	3%	0.064	183.4	3×10^{-4}
35	0.113	0.091	19%	0.129	479.7	2×10^{-4}
40	0.112	0.094	16%	0.136	538.8	2×10^{-4}
45	0.104	0.102	2%	0.104	320.8	3×10^{-4}
50	0.099	0.087	12%	0.102	348.3	2×10^{-4}
55	0.092	0.051	45%	0.146	685.5	2×10^{-4}
60	0.093	0.082	12%	0.111	743.0	1×10^{-4}
65	0.088	0.068	23%	0.112	458.1	2×10^{-4}
70	0.081	0.065	20%	0.147	462.1	3×10^{-4}
75	0.082	0.066	20%	0.117	486.2	2×10^{-4}
80	0.078	0.061	22%	0.072	514.9	1×10^{-4}
85	0.072	0.059	18%	0.098	532.7	1×10^{-4}
90	0.071	0.059	17%	0.091	564.7	1×10^{-4}
95	0.077	0.057	26%	0.102	590.6	1×10^{-4}

As seen from Table 3, there is a noticeable improvement in the overall accuracy and a decline in the computational performance when using DGP over GP for higher dimensional problems.

2. Static DOE for Airfoil Drag Coefficient Problem

The static DOE is tested on the other output quantity of interest in the aerodynamic test case: the non-dimensional drag coefficient. Figures 16 and 17 show the RMS error and training time required for each of the models at each training point setting for the airfoil drag coefficient problem respectively. The results and trends obtained are identical to those from the lift coefficient and OTL circuit problems.

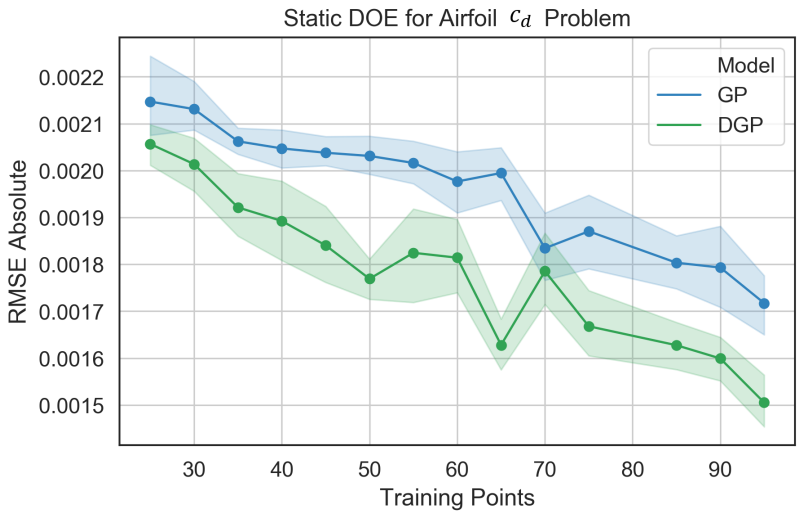


Fig. 16 Average RMS error for each surrogate model with increasing number of training points for the airfoil drag coefficient problem.

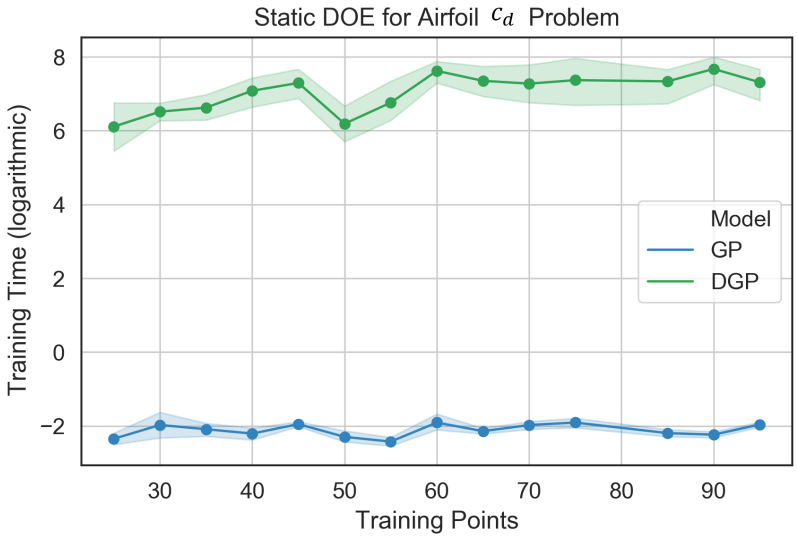


Fig. 17 Average training time for each surrogate model with increasing number of training points for the airfoil drag coefficient problem.

Table 4 shows the results of the best model obtained at each training point setting for each of the two types of surrogate models along with the percentage difference between the two (measured as $\left(\frac{GP-DGP}{GP}\right) \times 100$).

Table 4 Summary of results on airfoil drag coefficient problem with static DOE

Airfoil c_d Problem						
Training Points	RMSE GP	RMSE DGP	RMSE Difference	Training Time GP (s)	Training Time DGP (s)	Training Time Ratio (GP / DGP)
25	0.00198	0.00194	2%	0.133	92.15	1×10^{-3}
30	0.00204	0.00190	7%	0.211	548.2	3×10^{-4}
35	0.00202	0.00176	13%	0.162	387.33	4×10^{-4}
40	0.00192	0.00169	12%	0.133	468.8	2×10^{-4}
45	0.00199	0.00167	16%	0.149	697.5	2×10^{-4}
50	0.00195	0.00168	14%	0.161	359.3	4×10^{-4}
55	0.00193	0.00160	17%	0.121	387.0	3×10^{-4}
60	0.00188	0.00173	8%	0.303	739.9	4×10^{-4}
65	0.00185	0.00150	19%	0.141	603.9	2×10^{-4}
70	0.00170	0.00164	4%	0.167	455.6	3×10^{-4}
75	0.00176	0.00155	12%	0.205	251.8	8×10^{-4}
80	0.00172	0.00154	10%	0.273	714.7	3×10^{-4}
85	0.00168	0.00149	11%	0.141	182.9	7×10^{-4}
90	0.00162	0.00146	10%	0.125	558.2	2×10^{-4}
95	0.00153	0.00142	7%	0.160	766.3	2×10^{-4}

As seen from Table 4, there is a noticeable improvement in the overall accuracy and a decline in the computational performance when using DGP over GP for higher dimensional problems. It is worth noting that the computational time for both GP and DGP is almost the same for 42 dimensional problem as it is for the corresponding GP and DGP model used in the 6 dimensional problem. This observation is consistent with the results of the scalability study presented

is section IV.B. High-dimensional input space functions may demand a large number of evaluations on the original function to build a reliable global metamodel. Hence, the computational time associated with the metamodel building can be prohibitive, especially if there is a high computational cost on the function evaluation. This is where surrogate models that work well on higher dimensional problems such as DGP are valuable as they can provide reliable models with small data. With this understanding, it is important to explore the benefits of adaptive sampling demonstrated on the OTL circuit problem for the airfoil aerodynamic shape problem. The subsequent sections present the results obtained from adaptive sampling for the airfoil lift and drag coefficient problems.

3. Adaptive Sampling for Airfoil Lift Coefficient Problem

Similar to the OTL circuit problem, an initial candidate model with 40 training points is chosen for each of the two surrogate models. To ensure fair comparison across models, the model structure (hyperparameters, kernel, etc.) is fixed and new points are added adaptively. Then, the RMS error on the test set for the static DOE is calculated. Figure 18 shows the results obtained from applying adaptive sampling on the airfoil lift coefficient problem. A total of 30 sampled points are added to the original set of 40 points with the model being re-trained after each addition and calculating the RMS error on the test set.

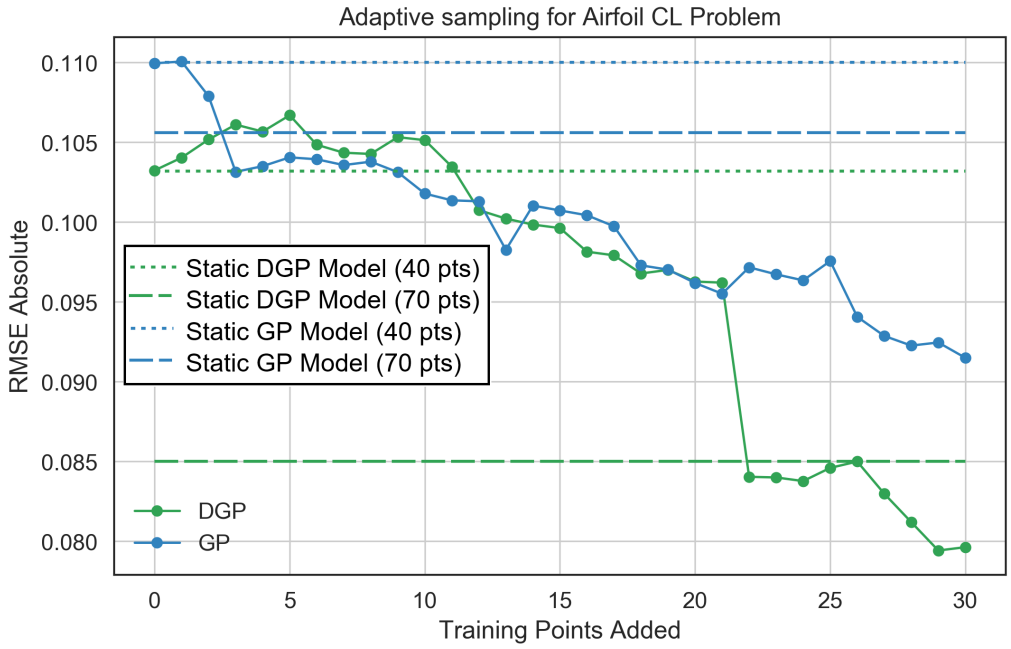


Fig. 18 Adaptive Sampling for Lift Coefficient Problem.

Figure 18 shows that the prediction accuracy of both the GP and DGP models steadily improves when adaptively adding training points. Both adaptively trained models end up with a higher predictive accuracy than the static models with equivalent total number of 70 training points. However, for the static models, the performance of the dashed line is obtained after a search across multiple hyperparameters and repetitions which requires significant computational resources whereas for the adaptive sampling the same model is retrained with the added points. Therefore, it is significantly more computationally efficient. The adaptive sampling DGP models with 70 points are $\sim 6\%$ more accurate in their predictive accuracy than the static DGP model with 70 points despite using a fixed hyperparameter setting, whereas the adaptive sampling GP models are $\sim 12\%$ more accurate in their predictive accuracy than the static GP models for 70 training points. Overall, the computational efficiency of using adaptive sampling over a static DOE coupled with the superior performance of DGP over GP make it a compelling combination especially for problems with a large input space.

4. Adaptive Sampling for Airfoil Drag Coefficient Problem

The experimental setup for the drag coefficient is identical to that of the lift coefficient. Figure 19 shows the results obtained by applying the adaptive sampling on the airfoil drag coefficient problem. A total of 30 adaptive samples are added to the original set of 30 intial training points while retraining the model at each step and calculating the RMS error on the test set.

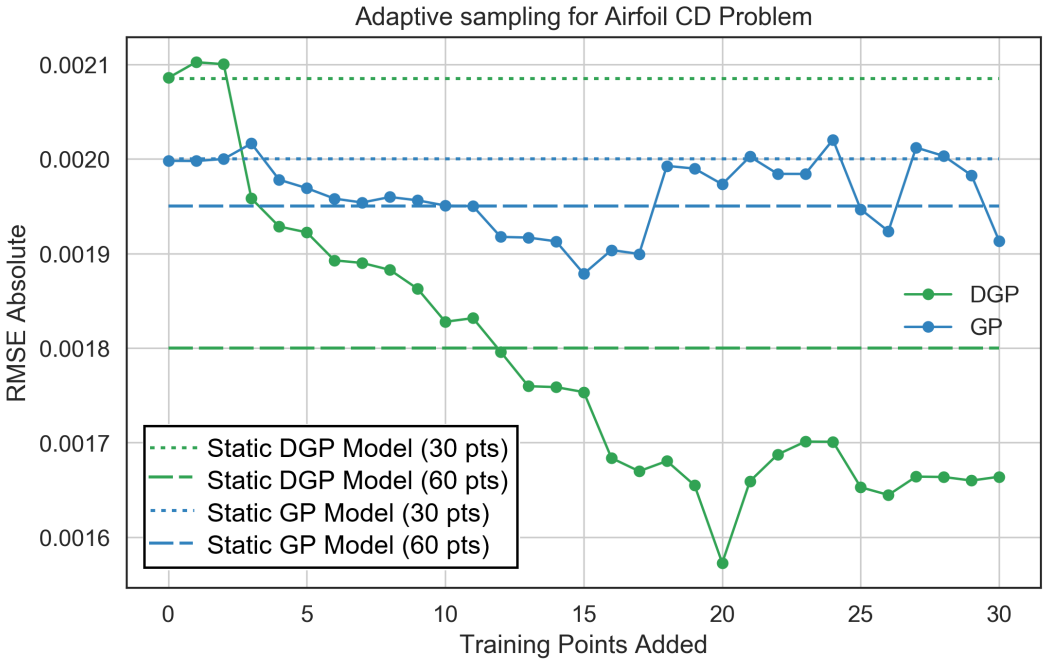


Fig. 19 Adaptive Sampling for Drag Coefficient Problem.

The accuracy of both the GP and DGP models improves with additional training points for the airfoil drag coefficient problem. However, for this problem, note that both the static and adaptive sampling based GP models do not improve much over the range of training points tested. The final adaptive GP model is only $\sim 2.5\%$ better in terms of the RMS error than the final static GP model with 60 points. For the DGP model on the other hand, the static model improves by $\sim 14\%$ from 30 to 60 training points whereas the adaptive sampling DPG model improves by $\sim 21\%$ from the initial 30 training points to the final model with 30 adaptively added points. Interestingly, the adaptive sampling DGP model surpasses the performance of the static GP model with 60 training points after adaptive additions of only 3 training points. It also surpasses the performance of the static DGP model with 60 training points after adaptively adding 11 training points, hence indicating that the DGP model with adaptive sampling offers hope of a significant improvement in the predictive accuracy with a relatively small number of added training points. In other words, the results verify the claims that DGP models indeed perform well under scarcity of data.

V. Conclusion and Future Work

This paper has demonstrated the potential benefits, value, and limitations of DGP as a surrogate model for problems characterized by high input dimensionality, small data, and complex behavior, especially for engineering functions. A systematic application and comparison of DGP with the traditional GP for a series of problems with increasing complexity is demonstrated. Across all problems, DGPs had a significantly accurate model in comparison to GPs. This observation was more pronounced for problems with large input spaces. A scalability study conducted for DGPs showed that across the several problems considered in this work, the models tends to scale well with increasing input dimensions and perform reasonably well for small datasets. A thorough sensitivity study over the various hyperparameters of DGP models was also conducted. The predictive accuracy of DGP models and their computational cost of training is influenced by the models' hyperparameter settings. Finally, the value of adaptive sampling was assessed for both DGP and GP models using a canonical engineering problem and a practical airfoil problem. Based on the results presented

in this work, DGP models seem to offer increased accuracy over traditional GPs for all problems explored at the cost of additional computational time for the training procedure. Therefore, DGPs are particularly suitable for problems with small data sets such as aerodynamic surrogate models where sufficient computational resources are available to train and validate the models but there is a significant incremental cost to get additional data points (either through high-fidelity simulation like computational fluid dynamics or actual experiments). DGPs trained on small data with adaptive sampling were shown to have significant promise for high-dimensional problems. In summary, this empirical study has shown that DGPs are a strong candidate for surrogate modeling of engineering problems. Results have provided clarity regarding the expected trade-off in computational cost to obtain a relatively accurate model using DGPs instead of GPs. Practitioners intending to employ DGPs can borrow findings from this work to decide whether the additional computational complexity is worth incurring for the gains in accuracy. When adaptive sampling is possible, results suggest the use of DGPs because they are superior to GPs and comparable in terms of computational cost when the budget for the number of true function evaluations is fixed. From a practitioner's perspective, it is recommended that the DGPs have at least one or two hidden layers and similar number of latent units. The number of inducing points is recommended to be close to or higher than the number of training points available. Finally, results show that DGPs should be considered for problems with high-dimensional input spaces whereas GPs would suffice for low-dimensional problems.

In future work, the benchmarking will be further extended to include other state-of-the-art surrogate models such as DNNs. The models may also be tested with higher fidelity aerodynamic simulation outputs. Another important avenue of future work that is being investigated is the inclusion of hyperparameter tuning in the adaptive sampling scheme or separate hyperparameter optimization for the surrogate model to improve predictive performance.

Acknowledgements

The authors would like to acknowledge the support of Siemens Corporate Technology for the work performed in this paper. In particular, we would like to thank Sanjeev Srivastava and Wei Xia from Siemens for their valuable technical feedback and support. S.A.Renganathan acknowledges the support by Laboratory Directed Research and Development (LDRD) funding from Argonne National Laboratory, provided by the Director, Office of Science, of the U.S. Department of Energy under Contract No. DE-AC02-06CH11357.

References

[1] Srivastava, A., and Meade, A. J., "Application of Machine Learning Techniques for Classification of Cavity Flow and Resonance," *Journal of Aircraft*, Vol. 51, No. 5, 2014, pp. 1642–1647. doi:10.2514/1.C032282.

[2] Kumar, A., and Ghosh, A. K., "Decision Tree–and Random Forest–Based Novel Unsteady Aerodynamics Modeling Using Flight Data," *Journal of Aircraft*, Vol. 56, No. 1, 2019, pp. 403–409. doi:10.2514/1.C035034.

[3] Becker, W., Worden, K., Battipede, M., and Surace, C., "Uncertainty Analysis of a Dynamic Model of a Novel Remotely Piloted Airship," *Journal of aircraft*, Vol. 48, No. 3, 2011, pp. 1028–1035. doi:10.2514/1.C031207.

[4] Damianou, A., and Lawrence, N., "Deep Gaussian Processes," *Artificial Intelligence and Statistics*, 2013, pp. 207–215. URL http://proceedings.mlr.press/v31/damianou13a.pdf.

[5] Toal, D. J., and Keane, A. J., "Efficient multipoint aerodynamic design optimization via cokriging," *Journal of Aircraft*, Vol. 48, No. 5, 2011, pp. 1685–1695. doi:10.2514/1.C031342.

[6] Kwon, H., Choi, S., Kwon, J.-H., and Lee, D., "Surrogate-Based Robust Optimization and Design to Unsteady Low-Noise Open Rotors," *Journal of Aircraft*, Vol. 53, No. 5, 2016, pp. 1448–1467. doi:10.2514/1.C033109.

[7] Tang, J., Hu, Y., Song, B., and Yang, H., "Unsteady Aerodynamic Optimization of Airfoil for Cycloidal Propellers Based on Surrogate Model," *Journal of Aircraft*, Vol. 54, No. 4, 2017, pp. 1241–1256. doi:10.2514/1.C033649.

[8] Damianou, A., "Deep Gaussian Processes and Variational Propagation of Uncertainty," Ph.d. thesis, University of Sheffield, 2015. doi:10.13140/RG.2.1.1458.8885.

[9] Vafa, K., and Rush, A., "Training and Inference for Deep Gaussian Processes," Ph.D. thesis, Harvard University, 2016.

[10] Zhang, Y., Ghosh, S., Asher, I., Ling, Y., and Wang, L., "Learning Uncertainty using Clustering and Local Gaussian Process Regression," *AIAA Scitech 2019 Forum*, 2019. doi:10.2514/6.2019-1730.

[11] Bui, T. D., Hernández-Lobato, D., Li, Y., Hernández-Lobato, J. M., and Turner, R. E., "Deep Gaussian Processes for Regression using Approximate Expectation Propagation," *Proceedings of the 33rd International Conference on Machine Learning, New York, NY, USA, 2016.*, Vol. 48, New York, NY, USA, 2016, pp. 1472–1481. doi:10.1162/NECO_a_00104.

[12] Salimbeni, H., and Deisenroth, M., "Doubly Stochastic Variational Inference for Deep Gaussian Processes," *31st Conference on Neural Information Processing Systems (NIPS 2017)*, Long Beach, CA, 2017, pp. 4588–4599. URL http://arxiv.org/abs/1705.08933.

[13] Hebbal, A., Brevault, L., Balesdent, M., Talbi, E.-G., and Melab, N., "Multi-objective optimization using Deep Gaussian Processes: Application to Aerospace Vehicle Design," *AIAA SciTech Forum*, San Diego, CA, 2019, pp. 1–19. doi:10.2514/6.2019-1973.

[14] Rasmussen, C. E., "Gaussian processes in machine learning," *Summer School on Machine Learning*, Springer, 2003, pp. 63–71. doi:10.1007/978-3-540-28650-9_4.

[15] Blei, D. M., Kucukelbir, A., and McAuliffe, J. D., "Variational inference: A review for statisticians," *Journal of the American Statistical Association*, Vol. 112, No. 518, 2017, pp. 859–877. doi:10.1080/01621459.2017.1285773.

[16] Andrieu, C., De Freitas, N., Doucet, A., and Jordan, M. I., "An introduction to MCMC for machine learning," *Machine learning*, Vol. 50, No. 1-2, 2003, pp. 5–43. doi:10.1023/A:1020281327116.

[17] Drela, M., "XFOIL: An Analysis and Ddesign System for Low Reynolds Number Airfoils," *Low Reynolds number aerodynamics*, Springer, 1989, pp. 1–12. doi:10.1007/978-3-642-84010-4_1.

[18] Ghosh, S., Ran, H., and Mavris, D. N., "A Generic Airfoil Design Method Based on a Naturally Bounded PARSEC Approach," *32nd AIAA Applied Aerodynamics Conference*, 2014, p. 2010. doi:10.2514/6.2014-2010.

[19] Kulfan, B., and Bussoletti, J., "Fundamental Parameteric Geometry Representations for Aircraft Component Shapes," *11th AIAA/ISSMO multidisciplinary analysis and optimization conference*, 2006, p. 6948. doi:10.2514/1.29958.

[20] Kulfan, B. M., "Universal parametric geometry representation method," *Journal of Aircraft*, Vol. 45, No. 1, 2008, pp. 142–158. doi:10.2514/1.29958.

[21] Sung, W. J., "A neural network construction method for surrogate modeling of physics-based analysis," Ph.D. thesis, Georgia Institute of Technology, 2012. URL http://hdl.handle.net/1853/43721.

[22] Zhang, Y., Sung, W. J., and Mavris, D. N., "Application of convolutional neural network to predict airfoil lift coefficient," *2018 AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2018. doi:10.2514/6.2018-1903.

[23] Liu, X., Zhu, Q., and Lu, H., "Modeling multiresponse surfaces for airfoil design with multiple-output-Gaussian-process regression," *Journal of Aircraft*, Vol. 51, No. 3, 2014, pp. 740–747. doi:10.2514/1.C032465.

[24] Ghosh, S., Asher, I., Kristensen, J., Ling, Y., Ryan, K., and Wang, L., "Bayesian Multi-Source Modeling with Legacy Data," *2018 AIAA Non-Deterministic Approaches Conference*, 2018. doi:10.2514/6.2018-1663.

[25] Eriksson, D., Dong, K., Lee, E., Bindel, D., and Wilson, A. G., "Scaling Gaussian process regression with derivatives," *Advances in Neural Information Processing Systems*, 2018, pp. 6867–6877. URL http://papers.nips.cc/paper/7919-scaling-gaussian-process-regression-with-derivatives.pdf.

[26] Wilson, A. G., Dann, C., and Nickisch, H., "Thoughts on massively scalable Gaussian processes," *arXiv preprint arXiv:1511.01870*, 2015. URL https://arxiv.org/pdf/1511.01870.pdf.

[27] Garbo, A., and German, B., "Comparison of adaptive design space exploration methods applied to S-duct CFD simulation," *57th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2016. doi:10.2514/6.2016-0416.

[28] Ghosh, S., Kristensen, J., Zhang, Y., Subber, W., and Wang, L., "A Strategy for Adaptive Sampling of Multi-Fidelity Gaussian Processes to Reduce Predictive Uncertainty," *ASME 2019 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, American Society of Mechanical Engineers Digital Collection, 2019. doi:10.1115/DETC2019-98418.

[29] Rajaram, D., and Pant, R., "An improved methodology for airfoil shape optimization using surrogate based design optimization," *Engineering Optimization*, Vol. IV, 2015, p. 147–152.

[30] Kennedy, J., and Eberhart, R., "Particle swarm optimization," *Proceedings of ICNN'95-International Conference on Neural Networks*, Vol. 4, IEEE, 1995, pp. 1942–1948. doi:10.1109/ICNN.1995.488968.

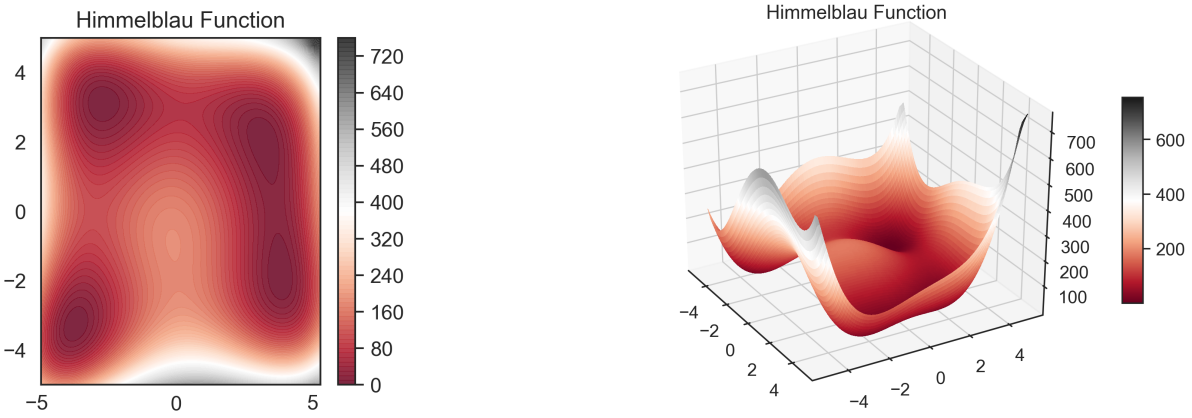
Appendix: Canonical Problems

In this section, additional details about the canonical functions used in this work are provided.

Himmelblau Function

The Himmelblau function is a commonly used analytical function of two dimensions that is continuous and non-convex. The function is usually defined on an input domain of [-5, 5] for both input dimensions. The function has four local minima located at [3, 2], [-2.8051, 3.2832], [-3.7793, -3.2832], and [3.5485, -1.8481]. The function contours and three dimensional surface plot are shown in Figures 20a and 20b respectively. The analytical form of this function is given by the following equation:

$$f(x, y) = (x^2 + y - 11)^2 + (x + y^2 - 7)^2 \tag{7}$$



(a) Contours of the Himmelblau function in its domain.

(b) Surface plot of the Himmelblau function in its domain.

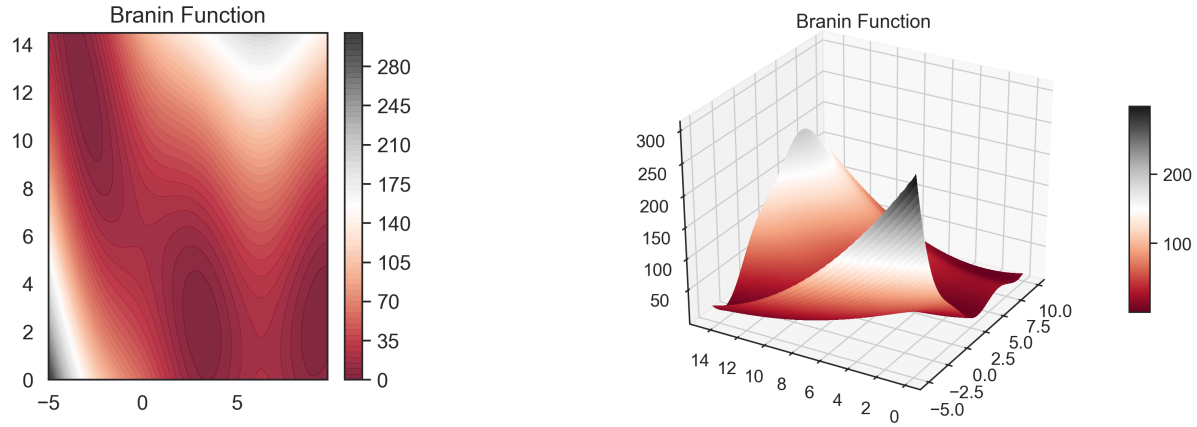
Fig. 20 Contours and surface plot of the Himmelblau function in its typical domain of definition.

Branin Function

The Branin or Branin-Hoo function is another commonly used analytical function of two dimensions that is continuous and non-convex. This function is usually evaluated on the square x_1 = [-5, 10], x_2 = [0, 15]. The function has three global minima located at [$-\pi$, 12.2750], [π, 2.2750], and [9.4248, 2.4750]. The function contours and three dimensional surface plot are shown in Figures 21a and 21b respectively. The analytical form of this function is given by the following equation:

$$f(x, y) = a(x_2 - bx_1^2 + cx_1 - r)^2 + s(1 - t)\cos(x_1) + s \tag{8}$$

The recommended values of the parameters are: a = 1, b = $5.1/(4\pi^2)$, c = $5/\pi$, r = 6, s = 10 and t = $1/(8\pi)$.



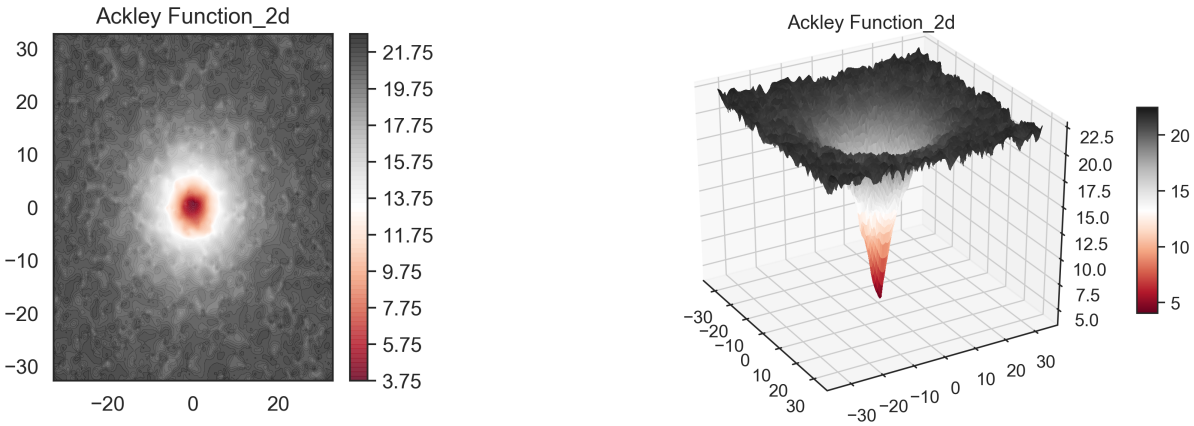
(a) Contours of the Branin function in its domain. **(b) Surface plot of the Branin function in its domain.**

Fig. 21 Contours and surface plot of the Branin function in its typical domain of definition.

Ackley Function

The Ackley function is widely used for testing scalability of optimization algorithms. In its two-dimensional form, it is characterized by a nearly flat outer region, and a large hole at the centre. The functional form can be used to create higher dimensional analytical problems. The function is usually evaluated on the hypercube x_i = [-32.768, 32.768]. It has a global minimum at [0, 0] and numerous local minima. The function contours and three dimensional surface plot of the two dimensional variant of this function are shown in Figures 22a and 22b respectively. The analytical form of this function is given by the following equation:

$$f(\mathbf{x}) = -a \exp\left(-b\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(cx_i)\right) + a + \exp(1) \tag{9}$$



(a) Contours of the 2-d Ackley function in its domain **(b) Surface plot of the 2-d Ackley function in its domain**

Fig. 22 Contours and surface plot of the 2-d Ackley function in its typical domain of definition