Efficient Sampling for Surrogate Modelling

1st Anthony Truelove
Pacific Regional Institute for Marine Energy Discovery (PRIMED)
Institute for Integrated Energy Systems (IESVic) - University of Victoria
Victoria, Canada
wtruelove@uvic.ca

Abstract*—Surrogate modelling has taken hold in a variety of domains in recent years, with model-based design optimization being a particularly fruitful application. That said, while the notions of "sampling efficiency" and "surrogate efficiency" have appeared in the body of surrogate modelling literature, a precise and general metric for these notions has not yet appeared. Therefore, this work addresses two questions: (1) Is there a general metric for surrogate efficiency?, and (2) Is there a clear "winner", with respect to surrogate efficiency, in terms of sampling scheme?. To address these questions, a general expression for surrogate efficiency is first proposed. Then, a set of numerical experiments is designed which seeks to test the proposed efficiency expression in application to instances of surrogate modelling for some optimization benchmark problems. In total, four benchmark problems (Rastrigin, Rosenbrock, Griewank, and Styblinski-Tang), two sampling schemes (simple random and latin hypercube), and five levels of dimensionality (2, 3, 4, 5, 6) are considered, together with a neural network as the surrogate. The results obtained from the numerical experiments suggest that the proposed efficiency expression is both general and useful as a performance metric (and so the answer to question 1 is yes). For instance, results suggest that for the $D = 4$ dimensional Rastrigin function defined on $[-5, 5]^4$, surrogate efficiency is maximized by a sample size N such that $\sqrt[D]{N} = 4$ (i.e., $N = 256$). Corresponding results for Rosenbrock, Griewank, and Styblinski-Tang are $\sqrt[D]{N} = 6$, $\sqrt[D]{N} = 11$, and $\sqrt[D]{N} = 5$ respectively. That said, the results also show that varying the sampling scheme between either simple random or latin hypercube has little to no impact on the resulting surrogate efficiencies (and so the answer to question 2 is no).

***Index Terms*—sampling, optimization, surrogate modelling**

I. Introduction

Consider an optimization problem in which the objective function is computationally expensive to evaluate, so much so that applying an optimization algorithm to the objective directly is intractable. For example, in model-based design optimization, the objective function is often high-resolution modelling and simulation software that can be used to assess the performance of candidate designs. However, when individual model runs take anywhere from minutes to days to complete, these models do not scale well in the context of optimization (where algorithms often evaluate the objective thousands of times [or more!] in search of optima). This then motivates the concept of a *surrogate model*: an approximation of a more expensive model that seeks to minimize computational expense while retaining a maximum of model fidelity [1].

Surrogate modelling has taken hold in a variety of domains in recent years. For example, [2] reviews the application of surrogate modelling to sustainable building design. An example is provided in [3] of applying surrogate modelling to the optimal design of combustion systems, while an example is provided in [4] of applying surrogate modelling to the optimal design of wind turbines. Yet another example is provided in [5] of applying surrogate modelling to the optimization of policy and decision-making in the domain of economics. Compound these examples with the utility of machine learning models as surrogates (as mentioned in all of [2], [3], [5]) and it seems that surrogate modelling will remain a useful technique (and hence a relevant topic) for the foreseeable future.

Of course, in order to construct a surrogate model for any given use case, some amount of data is required (which implies sampling the problem space via the computationally expensive objective function). As such, there exists the notion of surrogate utility versus surrogate cost (i.e., a *surrogate efficiency*), and this begs two questions:

1) Is there a general metric for surrogate efficiency?

2) Is there a clear "winner", with respect to surrogate efficiency, in terms of sampling scheme?

This work aims to address both questions.

II. Methodology

A. General Surrogate Efficiency Metric: Concepts

The concept of "sampling efficiency" in the context of surrogate modelling is mentioned (but never precisely defined) in all of [6]–[8]. Similarly searching the literature for references to "surrogate efficiency" yields a single relevant result [9] (but again, it is not precisely defined). Therefore, a novel and general surrogate efficiency metric η_{SM} is proposed as part of this work. The key concepts that influenced the proposition are summarized in the following sub-sections.

1) Logic: Any measure of surrogate efficiency should express the trade-off between surrogate utility (i.e., how accurate/precise is the surrogate?) and surrogate cost (i.e., what is the computational expense to build the surrogate in the first place?). This logic is sketched out in (1).

$$\eta_{\text{SM}} \sim \frac{\text{surrogate utility}}{\text{surrogate cost}} \tag{1}$$

2) Surrogate Utility: If surrogate utility is essentially a measure of surrogate accuracy and precision, then one might say that utility is inversely proportional to error (2).

$$\text{surrogate utility} \propto \frac{1}{\text{surrogate error}} \tag{2}$$

For the sake of generality, a normalized error metric is desirable. To that end, the damped absolute percentage error (d-APE) metric is selected in this work. As per [10], d-APE can be expressed as

$$\text{d-APE} = \begin{cases} \left|\frac{\widehat{y}-y}{T}\right| & \text{if } |y| \leq T \\ \\ \left|\frac{\widehat{y}-y}{y}\right| & \text{otherwise} \end{cases} \tag{3}$$

where $\widehat{y}$ is a value predicted by the surrogate, y is the corresponding "true value", and $T \neq 0$ is some domain-specific threshold. Finally, for any use case (i.e., any set of $\{(\widehat{y}_i,\ y_i)\}$), surrogate error can be expressed as

$$\text{surrogate error} = \mu_{\text{d-APE}} + \text{IQR}_{\text{d-APE}} \tag{4}$$

where $\mu_{\text{d-APE}} \geq 0$ is the mean of the d-APE values (i.e., a measure of surrogate accuracy) and $\text{IQR}_{\text{d-APE}} \geq 0$ is the interquartile range of the d-APE values (i.e., a measure of surrogate precision).

3) Surrogate Cost: If one assumes that the cost of building a surrogate is dominated by the computational expense of sampling the objective function, then it follows that surrogate cost is proportional to the number of samples. This logic is sketched out in (5)

$$\text{surrogate cost} \propto N \tag{5}$$

where $N > 0$ is the number of samples (i.e., the number of objective function calls).

B. General Surrogate Efficiency Metric: Proposition

Given the preceding concepts, the following expression for a general surrogate efficiency metric is proposed:

$$\eta_{\text{SM}} = \exp\left[-\sqrt[D]{N}(\mu_{\text{d-APE}} + \text{IQR}_{\text{d-APE}})\right] \tag{6}$$

where $D > 0$ is the problem dimensionality (i.e., number of objective function inputs). *This addresses question 1.*

Observe that the expression proposed in (6) has the following desirable properties:

- $\eta_{\text{SM}} \in [0, 1]$ for any values of D, N, $\mu_{\text{d-APE}}$, and $\text{IQR}_{\text{d-APE}}$. That is, (6) behaves like an efficiency metric.
- For any $\sqrt[D]{N} > 0$, $\eta_{\text{SM}} \to 1$ as $\mu_{\text{d-APE}} + \text{IQR}_{\text{d-APE}} \to 0$. That is, increasing surrogate utility is rewarded.
- For any $\mu_{\text{d-APE}} + \text{IQR}_{\text{d-APE}} > 0$, $\eta_{\text{SM}} \to 0$ as $\sqrt[D]{N} \to \infty$. That is, increasing surrogate cost is penalized.

In particular, note the form of $\sqrt[D]{N}$. The intent of this form is to adjust for the so-called "curse of dimensionality". That is, surrogate models for higher dimensional problems are simply more expensive to construct (more samples needed), and so the efficiency of these models should not be punished just because they are harder problems.

C. Sampling Schemes

For the purpose of this work, two common sampling schemes are considered; namely

1) Simple random sampling.
2) Latin hypercube sampling.

The efficacy of these sampling schemes is compared by analyzing how choice effects surrogate efficiency. *This serves to address question 2.*

D. Design of Experiments

With the aim of testing (6) and addressing question 2, a series of numerical experiments is undertaken. The key components of the experimental design are summarized in the following sub-sections. For implementation details and experimental reproduction, refer to the corresponding GitHub repository: https://github.com/gears1763-2/CIVE503_final_project.

1) Benchmark Problems: For the purpose of this work, a set of benchmark optimization problems is selected to serve as proxies for a computationally expensive objective function. This approach is motivated by the following:

- The benchmarks are actually cheap to compute, so investigating higher dimensionalities and larger sample sizes is tractable. Indeed, the selected benchmark problems are defined for any $D \geq 2$ and are of the form

$$\begin{aligned} y\ :\ \mathbb{R}^D &\to \mathbb{R} \\ \vec{x} = \begin{bmatrix} x_1 & x_2 & \cdots & x_D \end{bmatrix}^{\mathsf{T}} &\mapsto y(\vec{x}) \end{aligned}$$

- While the benchmarks are simple to express and can be solved exactly using classical methods, they nonetheless challenge optimization algorithms and numerical modelling techniques. As such, these benchmarks arguably represent an "upper bound", in terms of difficulty, on the problems that one might apply surrogate modelling to in practice.

The benchmarks selected in this work are

- The Rastrigin function:

$$y(\vec{x}) = AD + \sum_{i=1}^{D} \left[x_i^2 - A\cos(2\pi x_i)\right] \tag{7}$$

where $A = 10$.

- The Rosenbrock function:

$$y(\vec{x}) = \sum_{i=1}^{D-1} \left[A(x_{i+1} - x_i^2)^2 + (1 - x_i)^2\right] \tag{8}$$

where $A = 100$.

- The Griewank function:

$$y(\vec{x}) = 1 + \frac{1}{A}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) \tag{9}$$

where $A = 4000$.

- The Styblinski–Tang function:

$$y(\vec{x}) = \frac{1}{2}\sum_{i=1}^{D}\left[x_i^4 - Ax_i^2 + Bx_i\right] \tag{10}$$

where $A = 16$ and $B = 5$.

The benchmarks are illustrated (for the case $D = 2$ and $\vec{x} \in [-5, 5]^D$) in Figures 1 - 4. Note that throughout this work, all benchmark problems are restricted to the domain $[-5, 5]^D$.

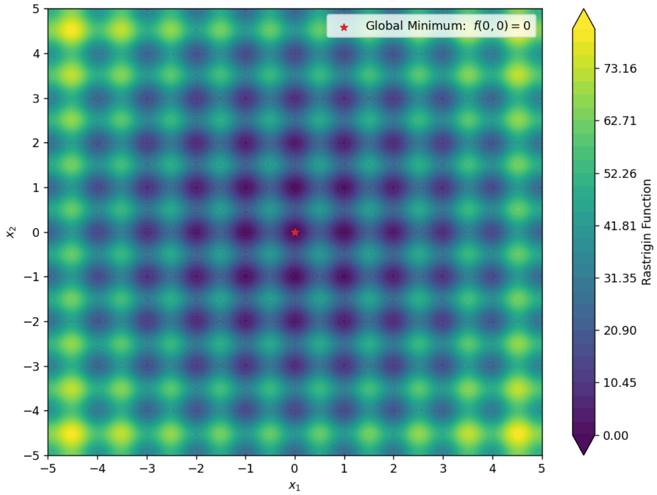

Fig. 1. The Rastrigin function.

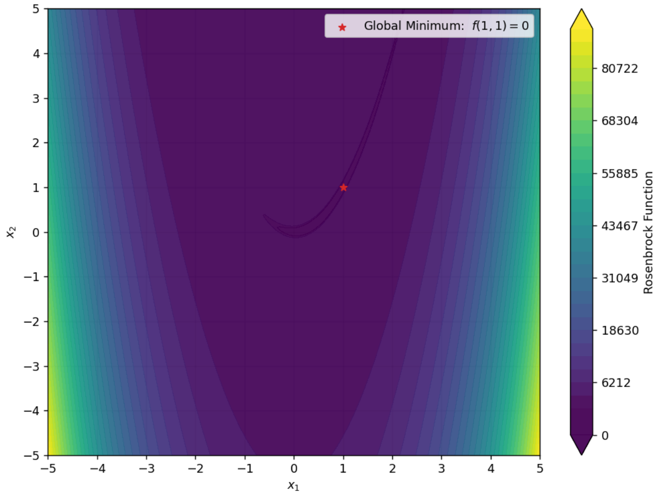

Fig. 2. The Rosenbrock function.

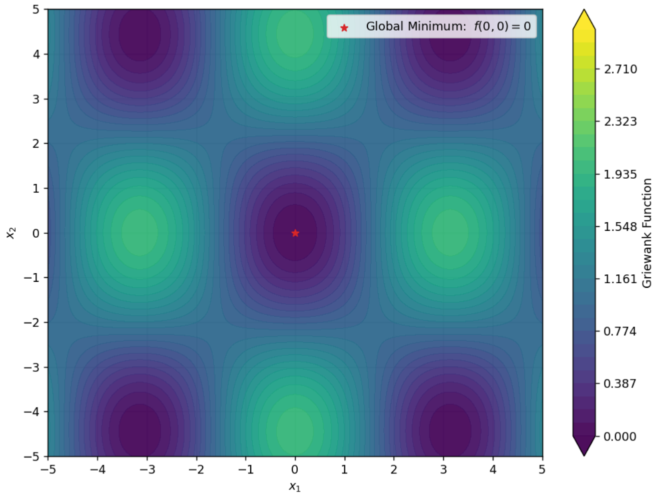

Fig. 3. The Griewank function.

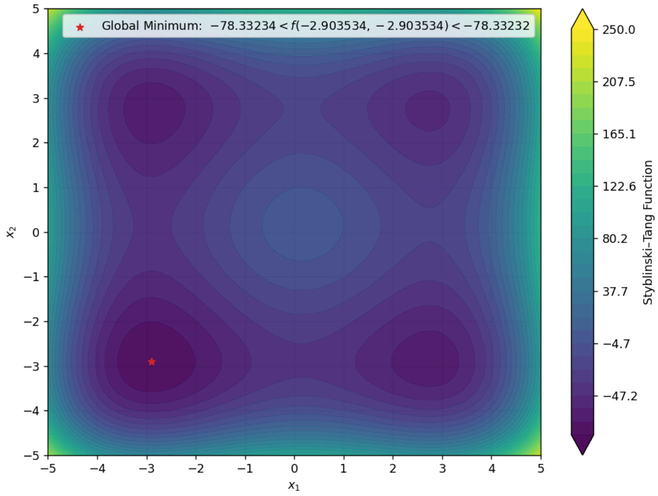

Fig. 4. The Styblinski–Tang function.

2) Surrogate Model: Given the ubiquity of machine learning surrogate models in the reviewed literature, this work adopts the same approach. In particular, a multilayer perceptron (MLP) with hidden layer architecture

$$\left(\underbrace{100, 100, \cdots, 100}_{D \text{ times}}\right)$$

that is, D fully-connected layers of 100 neurons each, is adopted. Model training, validation, and testing are carried out using the functionalities of scikit-learn and PyTorch; for details, refer to the GitHub repo. That said, note that the loss function used in the course of training, validation, and testing is (4).

3) Monte Carlo: In order to test (6) and address question 2, a full factorial Monte Carlo approach is taken. This approach can be summarized in the following pseudocode:

```
for benchmark_problem in benchmark_list:
  for sampling_scheme in scheme_list:
    for D in [2, 3, 4, 5, 6]:
      for N in [
       3D, 4D, 5D, ..., 10D,
       3^D, 4^D, 5^D, ..., 10^D
      ] up to max of 10^5:
        for trial = 1 ... 50:
```

```
sample objective, get data;

train/test split data;

normalize data;

train surrogate using
  validation/early-stopping;

test surrogate;

compute surrogate efficiency
  using test data;

log results;
```

That is, for every combination of benchmark problem, sampling scheme, dimensionality, and number of samples, do 50 Monte Carlo trials of sample (some randomness here), split (some randomness here), train (some randomness here), and test. By way of this approach, a results table with up to $4 \times 2 \times 5 \times 16 \times 50 = 32,000$ rows is generated. Note that the “`up to max of 10^5`” limit placed on number of samples (for the sake of runtime) is why the results table contains *up to* 32,000 rows; the actual number obtained will be less than this.

III. Results

The described full factorial Monte Carlo approach was carried out, and 30,800 rows of data were ultimately obtained. The scripts, data, and all visualizations are available on the GitHub repo. Figures 5 - 12 show a selection of results, namely the $\mathbb{R}^4 \rightarrow \mathbb{R}$ cases for each benchmark and sampling scheme.

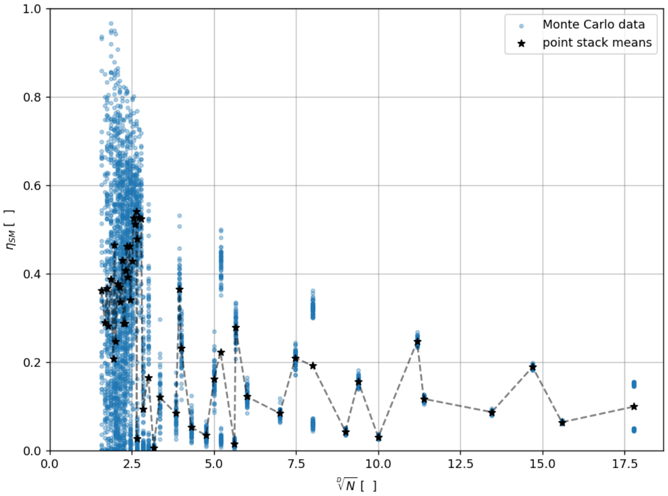

Fig. 5. Results for the Rastrigin function ($\mathbb{R}^4 \rightarrow \mathbb{R}$) under simple random sampling.

IV. Discussion

An inspection of Figures 5 - 12 reveals a few expected results. First, there is a high degree of variance within the Monte Carlo trials of a given sample size for small samples, with the variance tending to decrease as sample size increases. There are also noticeable differences between the simple random sampling results and the latin hypercube results for small samples. Beyond a certain sample size, however, the results appear similar for both sampling schemes.

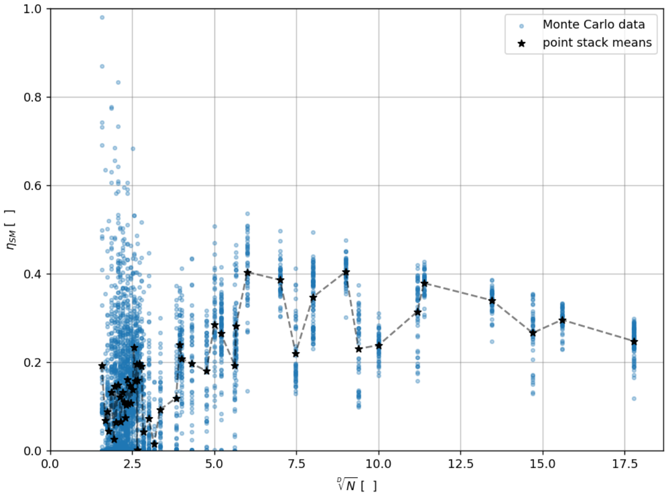

Fig. 6. Results for the Rosenbrock function ($\mathbb{R}^4 \rightarrow \mathbb{R}$) under simple random sampling.

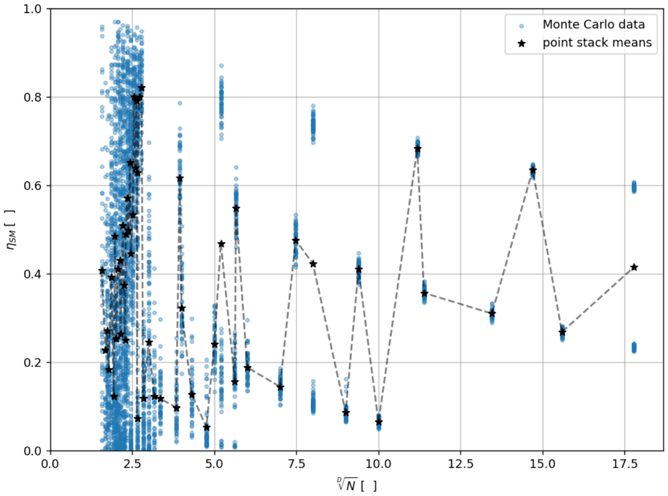

Fig. 7. Results for the Griewank function ($\mathbb{R}^4 \rightarrow \mathbb{R}$) under simple random sampling.

An inspection of Figures 5 - 12 also reveals a few interesting results. If one ignores the high variance region associated with small sample sizes, the surrogate efficiency for Rastrigin seems to attain a maximum around $\sqrt[D]{N} = 4$. Similarly, for Rosenbrock it appears to be around $\sqrt[D]{N} = 6$, and for Griewank it appears to be around $\sqrt[D]{N} = 11$. For Styblinski-Tang, it appears to be around $\sqrt[D]{N} = 4$ or 5, but the surrogate efficiency for this model also appears to be quite low across the range of sample sizes considered (thus suggesting that the

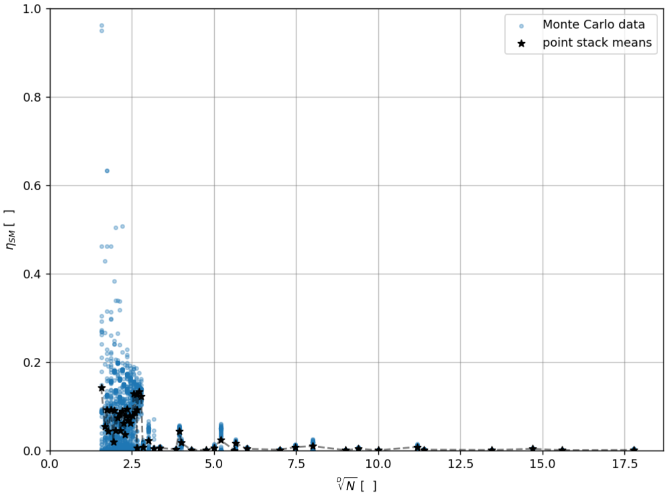

Fig. 8. Results for the Styblinski-Tang function ($\mathbb{R}^4 \to \mathbb{R}$) under simple random sampling.

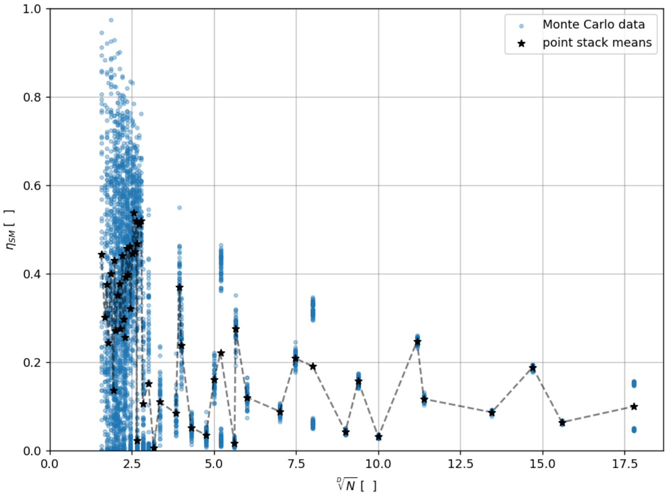

Fig. 9. Results for the Rastrigin function ($\mathbb{R}^4 \to \mathbb{R}$) under latin hypercube sampling.

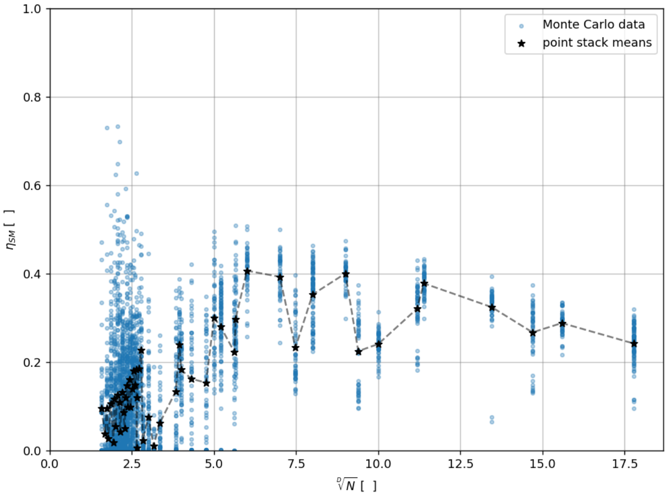

Fig. 10. Results for the Rosenbrock function ($\mathbb{R}^4 \to \mathbb{R}$) under latin hypercube sampling.

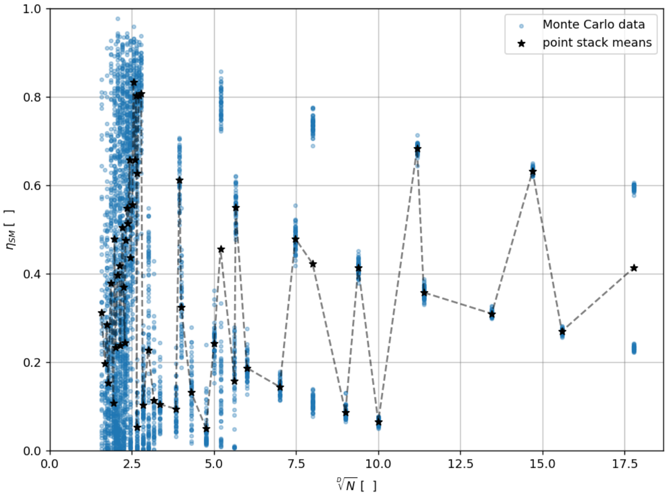

Fig. 11. Results for the Griewank function ($\mathbb{R}^4 \to \mathbb{R}$) under latin hypercube sampling.

choice of surrogate model architecture was a poor one for the Styblinski-Tang use case).

V. Conclusion

Recall the questions that this work set out to address; namely

1) Is there a general metric for surrogate efficiency?
2) Is there a clear “winner”, with respect to surrogate efficiency, in terms of sampling scheme?

Given the interesting results obtained, the proposed surrogate efficiency metric appears to behave in an ideal way. That is to say, it seems to work well over the set of benchmark problems chosen (i.e., it is general), and in each case it suggested an optimal sample size for a given benchmark, dimensionality, and sampling scheme (i.e., it is useful as a performance metric). Therefore, the answer to question 1 is yes (and the expression proposed in (6) seems to serve as a general metric).

As for question 2, the answer is most likely no. For smaller sample sizes, the point stack means tend to exhibit less variance in the latin hypercube case than in the simple random sampling case, but the actual surrogate efficiency values obtained are very similar in both cases. In addition, as was previously observed, past a certain sampling size the sampling scheme had no discernible impact upon surrogate efficiency.

VI. Future Work

In reviewing the methodology and results of this work, it seems that the choice of $\sqrt[D]{N}$ in (6), while motivated by the consensus that problems increase in difficulty as their

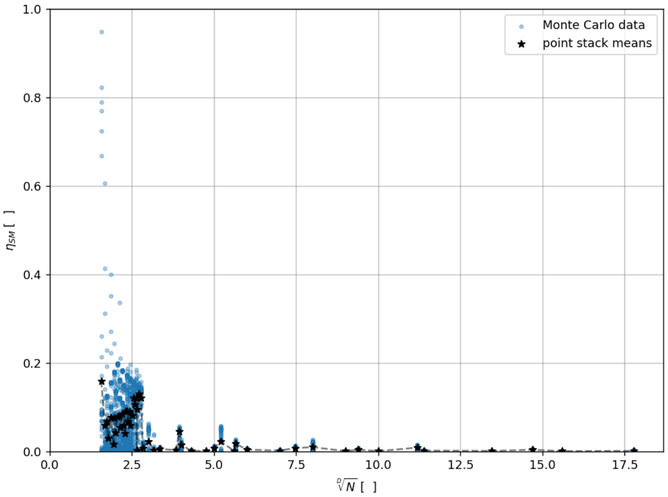

Fig. 12. Results for the Styblinski-Tang function ($\mathbb{R}^4 \rightarrow \mathbb{R}$) under latin hypercube sampling.

dimensionality increases (the so-called "curse of dimensionality"), may have overestimated the degree to which difficulty increases in the number of dimensions. Therefore, rather than assuming that every problem difficulty scales like $\mathcal{O}(2^D)$, it would be more general to allow for problems to scale in various ways such as, for example, $\mathcal{O}(D^2)$ or $\mathcal{O}(D\log_2(D))$. This then suggests alternate expressions for surrogate efficiency such as

$$\eta_{\text{SM}} = \exp\left[-\frac{N}{D^2}\left(\mu_{\text{d-APE}} + \text{IQR}_{\text{d-APE}}\right)\right] \tag{11}$$

or

$$\eta_{\text{SM}} = \begin{cases} \exp\left[-N\left(\mu_{\text{d-APE}} + \text{IQR}_{\text{d-APE}}\right)\right] & \text{if } D = 1 \\ \exp\left[-\frac{N}{D\log_2(D)}\left(\mu_{\text{d-APE}} + \text{IQR}_{\text{d-APE}}\right)\right] & \text{otherwise} \end{cases} \tag{12}$$

This then implies the following future work: experiment with modifying (6) and updating the full factorial Monte Carlo design to suit. For instance, consider changing the sampling size approach to something like

```
for N in [
    3D, 4D, ..., 20D,
    3Dlog2(D), 4Dlog2(D), ..., 20Dlog2(D),
    3D^2, 4D^2,  ..., 20D^2
]
```

References

[1] A. I. J. Forrester, A. Sóbester, and A. J. Keane, *Engineering Design via Surrogate Modelling: A Practical Guide*. John Wiley & Sons, Ltd, 2008, ISBN: 978-0-47-077080-1.

[2] P. Westermann and R. Evins, "Surrogate modelling for sustainable building design - a review," *Energy & Buildings*, vol. 198, 2019, DOI: 10.1016/j.enbuild.2019.05.057.

[3] K. Liu, K. Luo, Y. Cheng, A. Liu, H. Li, J. Fan, and S. Balachandar, "Surrogate modeling of parameterized multi-dimensional premixed combustion with physics-informed neural networks for rapid exploration of design space," *Combustion and Flame*, vol. 258, 2023, DOI: 10.1016/j.combustflame.2023.113094.

[4] R. Haghi and C. Crawford, "Surrogate models for the blade element momentum aerodynamic model using non-intrusive polynomial chaos expansions," *Wind Energy Science*, 2022, DOI: 10.5194/wes-7-1289-2022.

[5] S. van der Hoog, "Surrogate modelling in (and of) agent-based models: A prospectus," *Computational Economics*, vol. 53, 2018, DOI: 10.1007/s10614-018-9802-0.

[6] W. Gong and Q. Duan, "An adaptive surrogate modeling-based sampling strategy for parameter optimization and distribution estimation (asmo-pode)," *Environmental Modelling & Software*, vol. 95, 2017, DOI: 10.1016/j.envsoft.2017.05.005.

[7] P. Westermann and R. Evins, "Adaptive sampling for building simulation surrogate model derivation using the lola-voronoi algorithm," *Proceedings of the 16th IBPSA Conference*, 2019, DOI: 10.26868/25222708.2019.211232.

[8] J. Yin, W. Lu, X. Xin, and L. Zhang, "Application of monte carlo sampling and latin hypercube sampling methods in pumping schedule design during establishing surrogate model," *Proceedings of International Symposium on Water Resource and Environmental Protection*, 2011, DOI: .

[9] T. Casper, U. Römer, and S. Schöps, "Determination of bond wire failure probabilities in microelectronic packages," *Proceedings of 22nd International Workshop on Thermal Investigations of ICs and Systems*, DOI: 10.1109/THERMINIC.2016.7748645.

[10] D. Rulff and R. Evins, "Systematic refinement of surrogate modelling procedure for useful application to building energy problems," *Journal of Building Performance Simulation*, 2024, DOI: 10.1080/19401493.2024.2440418.