LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

LLNL-JRNL-850052

Non-Uniform Active Learning For Gaussian Process models

K. R. Quinlan, J. Movva, B. Perfect

June 8, 2023

Statistical Analysis and Data Mining: The ASA Data Science Journal

Disclaimer

This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.

Non-Uniform Active Learning For Gaussian Process Models with Applications to Trajectory Informed Aerodynamic Databases

Kevin R. Quinlan*†, Jagadeesh Movva‡ and Brad Perfect §

Lawrence Livermore National Laboratory, Livermore, CA, 94550, USA

Abstract

The ability to non-uniformly weight the input space is desirable for many applications, and has been explored for space-filling approaches. Increased interests in linking models, such as in a digital twinning framework, increases the need for sampling emulators where they are most likely to be evaluated. In particular, we apply non-uniform sampling methods for the construction of aerodynamic databases. This paper combines non-uniform weighting with active learning for Gaussian Processes (GPs) to develop a closed form solution to a non-uniform active learning criterion. We accomplish this by utilizing a kernel density estimator as the weight function. We demonstrate the need and efficacy of this approach with an atmospheric entry example that accounts for both model uncertainty as well as the practical state space of the vehicle, as determined by forward modeling within the active learning loop.

1 Introduction

An important aspect of building surrogate models of computer simulations is generating simulation results in an efficient manner. Space-filling designs and active learning approaches have been shown to be highly effective (see Santner et al. (2003); Gramacy (2020) for summaries). However, as mentioned in Bowman and Woods (2013), many applications do not have equal interest in all parts of the input space. The need for non-uniform approaches is particularly apparent for linked computer simulations (Kyzyurova et al., 2018). As a motivating example, consider the problem of constructing a database of aerodynamic coefficients (hereafter referred to as an aero-database). Aero-databases are used in flight simulation and give information on 6 degree of freedom aerodynamic coefficients (e.g., lift and drag) of a vehicle under different flight conditions (e.g. Mach and Altitude). This database functions as a rapidly queryable surrogate for the otherwise computationally expensive process of obtaining full computational fluid dynamic (CFD) solutions. The number of timesteps required to adequately model atmospheric entry involves possibly thousands of evaluations for a single second of real time which makes computing full CFD solutions within the forward modeling loop intractable. Thus, surrogate models are needed to provide predictions over the range of flight parameters. The typical approach of aero-database generation is to uniformly sample a full factorial parameter space of relevant variables (e.g. angle of attack, mach number) with CFD over some anticipated range (Burchett et al., 2022). Applying approaches from the computer experiments literature have been shown to achieve considerable computational time savings over traditional gridded designs for flight applications (Quinlan et al., 2021). Here, we demonstrate an additional benefit of a Gaussian Process (GP) aero-database: rather than specifying the desired input space and sampling strategy a priori, we can sequentially build the database and utilize interim results from the 6 degree-of-freedom flight simulation to focus the design to more relevant regions. We, therefore, need an approach to generate samples that will increase the accuracy of the surrogate in regions used by the forward models rather than improving database accuracy for infeasible flight conditions. Our approach in this work is to sample the input space of the first simulator (e.g., CFD solver) subject to a weight function based on the most commonly observed conditions by the second simulator (e.g., trajectory

*Corrsponding Author: quinlan5@llnl.gov

†Applied Statistician, Applied Statistics Group

‡Aerospace Engineer Analyst, Weapons and Complex Integration

§Analyst, Computational Engineering Division

simulator). In addition to the present application, there are many other practical reasons why some parts of the input space may be more valuable, such as having an increased risk or even having differential costs (real or computational) associated with the experiments. Although our focus in this paper is on the case of two linked simulators the methodology is more generally applicable.

The overall goal of this work is to demonstrate that combining information from the interim surrogate model with a relevant weight function can reduce the number of required samples compared to other state-of-the-art methods while still representing important features of a more densely sampled database. Constructing space-filling designs with non-uniform weighting has been studied in other works such as Bowman and Woods (2013), Joseph et al. (2015a, 2019), and Lu et al. (2021), but space-filling approaches do not directly incorporate information provided by the fitted model. The idea of incorporating a weight function into a sequential design criterion for GPs was originally suggested in Sacks et al. (1989a). It has also been used in other works such as Picheny et al. (2010) for the problem of contour estimation. However, these approaches leverage some approximation of the full criterion. We extend work in Binois et al. (2019) to develop a closed form expression for a non-uniform integrated mean square prediction error (NU-IMSPE) criterion, which minimizes the weighted average model prediction variance. We note that the difference between our Non-Uniform IMSPE and the weighted IMSPE in Leatherman et al. (2018) is that they were considering weighting different potential lengthscale hyperparameters rather than sampling on some observed weight function. The active learning (or sequential design) described in this work comes from utilizing the NU-IMSPE on the current GP fit to inform the location of additional samples for the simulator under study.

In this paper we derive the closed form NU-IMSPE and provide the expressions for a Kernel Density Estimator (KDE) weight function. We also demonstrate the efficacy of this approach using a simulation study and a case study involving Lawrence Livermore National Lab's STREAM geometry. Section 2 describes the relevant background on GP modeling as well as covering some relevant design of experiments methods. In Section 3, we derive closed form solution for calculating NU-IMSPE and describe the optimization algorithm. Following that, we will demonstrate the effectiveness of this approach in Section 4 with an atmospheric entry simulation example. Section 5 will provide commentary and conclusions.

2 Background Information

2.1 Non-Uniform Space-Filling Designs

Here, we briefly describe design of experiments when a non-uniform weight function is applied over an input space. Non-uniform weighting in space-filling designs was introduced in Bowman and Woods (2013). A generalized framework for non-uniform space filling designs is detailed in Lu et al. (2021). They describe a maximin style distance criterion Johnson et al. (1990) applied with an associated weight function. For a design of size n, the optimized design D_{Mm} takes the form

$$D_{Mm}(n) = \max_{D_n \subset \chi^p} \min_{\mathbf{x}_i, \mathbf{x}_j \in D_n} d^w(x_i, x_j), \tag{1}$$

with the distance criterion,

$$d^w(x_i, x_j) = \sqrt{w(x_i)w(x_j)d(x_i, x_j)^2}. \tag{2}$$

The criterion maximizes the minimum distance between points subject to a given weight function. In addition to the criterion (1), Lu et al. (2021) also introduces the idea of a Maximum Weight Ratio (MWR) to allow the practitioner to have control over the degree of non-uniformity. Note, that for our application the weight function incorporates relevant sources of uncertainty and so will be used directly with no modification when applying the NUSF. The specific distance function above is a special case of the generalized energy criterion for a Minimum Energy Design (MED) as described in Joseph et al. (2015a, 2019); however, note that the NUSF can take more general forms that are not special cases of the MED. In this work, we apply a Maximum Projection style distance Joseph et al. (2015b), which leads exactly to the criterion used for an MED. Figure 1 shows an example NUSF design, where the banana function

$$f(x) = \exp\left(-\frac{1}{2}\frac{x_1^2}{100} - \frac{1}{2}(x_2 + 0.03x_1^2 - 3)^2\right)$$

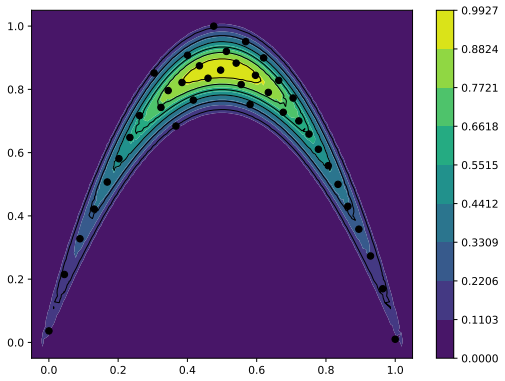

Figure 1: Example Minimum Energy Design.

as defined in Haario et al. (1999) is the weight function.

NUSF methods are valuable for generating an initial design that will be supplemented later and for cases where only a single set of points is desired. Sequential design strategies can benefit from interim information from previously-collected data and yield better overall results for the same simulation budget. Lu and Anderson-Cook (2021) extended the NUSF approach to a sequential design context and such space-filling approaches are advantageous in that they have no implicit model assumptions and can be applied generally. However, active learning approaches such as those described in the following sections should provide better performance by utilizing the current fitted model when assumptions are reasonably met (e.g. stationarity for a GP and well estimated model parameters).

2.2 Gaussian Process Models and Active Learning

Space-filling approaches, as seen in the previous section, may be used sequentially but do not by themselves contain a mechanism for incorporating information from the fitted model to generate subsequent samples. The process by which this step is included is referred to as active learning; here, we review the basic formulation for a rectangular region en route to the derivation of a non-uniform variation.

A Gaussian Process (GP) model is commonly used for computer experiments due to its flexibility. It was first popularized for computer experiment literature by Sacks et al. (1989b). GPs assume that random variables are jointly Gaussian and therefore can be fully defined by mean and covariance functions. Let $Y(x) \sim GP(\mu(x), C(x, x')$ be a Gaussian Process. In this work we assume the mean function can be set to zero, and let the covariance take the form,

$$C(x, x') = \tau^2 K(x, x') + \theta I_{x=x'}. \tag{3}$$

Where τ^2 is a scaling or variance hyperparameter, θ is the nugget parameter, and $I_{x=x'}$ in indicator that $x = x'$. Although we typically assume computer experiments are deterministic the introduction of the nugget parameter is still useful (Gramacy and Lee, 2012). For example, the resulting data from CFD solvers comes from the optimization of complex problems and will typically have some small amount of convergence error. By incorporating a nugget into our GP we can smooth out these artifacts. There are many potential choices for $K(x, x')$, but consider the Matèrn 5/2,

$$K(x, x') = \left(1 + \frac{\sqrt{5}d}{\nu} + \frac{5d^2}{3\nu^2}\right) \exp\left(-\frac{\sqrt{5}d}{\nu}\right), \tag{4}$$

where ν is the lengthscale parameter and $d = |x - x'|$, as exemplary. The kernel function controls the rate of decay for correlation of neighboring points through the lengthscale parameter. We consider anisotropic models so that each input parameter will have an associated lengthscale.

The prediction variance from a fitted GP takes the form

$$\hat{\sigma}^2(x^*) = \theta + \tau^2[1 - k(x^*)K^{-1}k(x^*)], \tag{5}$$

and we can utilize the prediction variance as the basis of our active learning criterion. Note, that by fixing the estimated hyperparameters the prediction variance can be calculated for a new design. Active learning takes advantage of this property to better inform future placement of design points. Specifically, we can minimize the model's overall prediction variance and this is the Integrated Mean Square Prediction Error (IMSPE) criterion. Sacks et al. (1989a) originally introduced the idea of using gaussian process models to inform the design in a computer experiments context. In fact, they even mention that a weight function could be applied to this criterion. This approach to sampling is also sometimes referred to as Active Learning Cohn (ALC) (Cohn, 1993). The basic criteria can be defined as

$$\text{IMSPE}(X_N) = \int_{x \in D} \check{\sigma}_N^2(x) dx \tag{6}$$

where D is the design space ($[0,1]^k$ for our purposes) and X_N is the current design. $\check{\sigma}_N^2(x)$ is the prediction variance with the nugget parameter removed. In our case, this is a constant value so optimizing $\check{\sigma}_N^2(x)$ is equivalent to optimizing $\sigma_N^2(x)$. This criteria is integrating the model prediction variance over the input space of interest. The overall goal is to minimize this quantity for IMSPE(X_{N+1}). Binois et al. (2019) developed a general closed form solution for the IMSPE criterion that also works for heteroscedastic computer simulations. Previous approaches generally approximated the criterion using Monte Carlo integration or substituted another quantity such as the maximum variance. These approaches cover the uniform weighting case, and we will extend these methods to a non-uniform setting.

3 Non-Uniform Active Learning for Gaussian Processes

We combine advances in closed form IMSPE calculations with a given weight function to sample non-uniformly. We provide an example with a KDE weighting which can be extended to more general weight functions. The KDE is a natural choice for our motivating example and applications where a known functional form of the weight function does not exist. This would be most common in computer simulations when the weight function comes from propagating uncertainty through to a forward model.

3.1 Non-Uniform IMSPE

This section extends the closed form calculation of the IMSPE to a weighted version of the calculation, specifically

$$\text{NU-IMSPE}(X_N) = I_{w,N} = \int_{x \in D} w(x)\check{\sigma}^2(x) dx \tag{7}$$

over the range $D = [0,1]^k$. With $\check{\sigma}^2(x)$ the prediction variance with the nugget parameter removed. First we show a general representation for an arbitrary weight function, and can thus find what conditions are required for a closed form solution. In Binois et al. (2019), they developed a closed form expression for calculating the IMSPE criterion by re-characterizing the IMSPE criterion in terms of solvable integrals. The proof here for the non-uniform case follows similar steps as the presence of the weight function does not fundamentally change the framework of the solution.

Lemma 1. *The Non-Uniform IMSPE criterion* $\int_{x \in D} w(x)\check{\sigma}^2(x)dx$ *can be re-expressed as* $\tau^2 E[w(X)] - tr(K_N^{-1}V_N)$ *with* $V_N = \int_{x \in D} w(x)k(x_i, x)k(x_j, x)$

Proof. By definition,

$$\int w(x)\check{\sigma}^2(x)dx = (E\left[w(X)k(X,X)\right] - E\left[w(X)k_N(X)^T K_N^{-1} k_N(X)\right]). \tag{8}$$

The first term simplifies due to the properties of the kernel so that

$$E\left[w(X)k(X,X)\right] = \tau^2 E\left[w(X)\right]. \tag{9}$$

Let $z = \sqrt{w(x)}k(x)$. Using the identity $E(z^t K_N^{-1} z) = tr(K_N^{-1}M) + m^T K_N^{-1} m$ where m is the mean of z and M is the covariance of z leads to (Petersen et al., 2008),

$$E\left[w(X)k_N(X)^T K_N^{-1} k_N(X)\right]) = tr(K_N^{-1}M + m^T K_N^{-1} m) = tr(K_N^{-1}(M + mm^t)) = tr(K_N^{-1} V_N). \tag{10}$$

Taking properties of expectation gives that

$$V_N = \int_{x\in D} w(x)k(x_i,x)k(x_j,x)dx. \tag{11}$$

□

Based on this result a closed form exists for the NU-IMSPE if there is a closed form solution for V_N and for $E[w(X)]$. For the case where $w(x)$ is represented by a kernel density estimator the value V_N becomes

$$V_N = \sum_k \int_{x\in D} k(x_i,x)k(x_j,x)k_w(x_k,x). \tag{12}$$

Where k_w is the KDE value. In the appendix we provide the form of V_N for the Gaussian and Matèrn 5/2 kernels with a Gaussian kernel for the KDE. Note, that our interest is not in the actual value of the NU-IMSPE, but rather in the improvement from additional samples, that is $I_{w,N+1} - I_{w,N}$. When considering this quantity for optimization the term $E[w(x)]$ is not needed for the sequential update formula. For the uniform case, the total calculation is $O(N^2)$ due to the update formula from Binois et al. (2019) and these same speed-ups are directly applicable for the non-uniform case by substituting in the V_N matrix. We write out the updated formula with the appropriate changes for the non uniform case here. Let $h(x) = -\sigma_n^2(x)^{-1} K_N^{-1} k_N(x)$, then the update formula is written as

$$I_{w,N_1}(\mathbf{x}) - I_{w,N}(x) = -\sigma_n^2(x)h(x)^T V_N h(x) + 2v(x)^T h(x) + \sigma_n^2(x)^{-1} v(x,x)). \tag{13}$$

Let M be the total number of samples for the KDE. Then the calculation of V_N and $v(x,x)$ each require M calculations resulting in a total computation time of $O(MN^2)$. Therefore it is advantageous computationally if one can limit the number of samples included in the KDE representation of the weight function. We have found that it is beneficial to generate a representative downsample of the original data using methods such as Mak and Joseph (2018); Vakayil and Joseph (2022) when generating the KDE weight function. Finally, we note that the approach above can be further extended to a non-uniform analog of the batch sequential framework as described in Zhang et al. (2020).

To optimize the design an L-BFGS-B algorithm was utilized as implemented in the Python package scipy (Virtanen et al., 2020). Multiple random starts were generated to ensure better solutions as this methodology will converge to a local optima. Initial points for each random restart are generated by obtaining samples directly from the KDE weight function. More global optimization algorithms such as simulated annealing (Kirkpatrick et al., 1983) could provide better solutions, but would greatly increase the computational cost. It would also be possible to improve the optimization speed by deriving gradient information for the sequential update criterion but we leave this for future work.

4 Examples

4.1 OTL Circuit with Banana weight function

NU-IMSPE is first demonstrated on a canonical set of inputs using the output-transformerless push-pull circuit problem (Ben-Ari and Steinberg, 2007). This can also be found on the Simon Fraiser Virtual Library of Simulation Experiments.

$$V_m(x) = \frac{(V_{b1} + 0.74)\beta(R_{c2} + 9)}{\beta(R_{c2} + 9) + R_f} + \frac{11.35R_f}{\beta(R_{c2} + 9) + R_f} + \frac{0.74R_f\beta(R_{c2} + 9)}{(\beta(R_{c2} + 9) + R_f)R_{c1}}, \text{with}$$
$$V_{b1} = 12R_{b2}R_{b1} + R_{b2}$$

The OTL circuit problem has two highly sensitive parameters, two slightly sensitive parameters, and two relatively insensitive parameters. The weight function is the banana function mentioned in Section 2.1 in Figure 1 on the two most sensitive parameters.

This weight function was chosen since it restricts the input space, but still spans most of the viable region with positive probability. Therefore, we could not reduce the surrounding rectangular region even if we knew the high probability regions a-priori. The choice of the OTL circuit is also useful as a stand in for our aero-database problem as not all flight coefficients will be sensitive to the same flight parameters.

For this example, we consider a comparison between 5 methods: a random sample as a baseline, a space-filling approach, a non-uniform space-filling approach, IMSPE, and NU-IMSPE sampling. This example problem spans six dimensions, so we utilize a Maximum Projection (MaxPro) optimized Latin Hypercube Design (LHD) as implemented in the **MaxPro** R package (Ba and Joseph, 2018) as a space-filling design. For the non-uniform space-filling design we refer to this as the NUSF: MED to indicate the approach and design type. Both of these space-filling approaches should do a better job of filling the projected active space (only four of the six parameters are active) compared to alternative distance metrics and thus give the best performance among other space-filling choices. Additionally, a new design is generated at each unique number of samples rather than continually adding one or multiple points at a time and this should out-perform a sequential space-filling approach. Finally, the IMSPE sampling is implemented using the **hetGP** R package (Binois and Gramacy, 2021).

Each evaluation starts with a 25 sample space-filling design (MaxPro for IMSPE and MED for NU-IMSPE) and add samples are added sequentially until there are 100 total. Each method is repeated 50 times to account for variability in the initial design and optimization of each design method. All methods use a GP with a Gaussian kernel. Note, that for the NUSF: MED approach uses the exact weight function, but a KDE approximation is applied for the NU-IMSPE sampling. The weight function is generated by first obtaining a large sample from the weight function and this was downsampled to the 200 most representative points using data twinning (Vakayil and Joseph, 2022). The comparison thus ensures the best case scenario for other methods and a somewhat realistic approach for our methodology. All methods are evaluated based on the weighted root mean square error (wRMSE) defined as,

$$\sqrt{\sum_i \left(w_i(Y_i - \hat{Y}_i)^2\right)/N}, \text{with} \sum_i w_i = 1, \tag{14}$$

and evaluated over a 10,000 sample LHD. This should give a measure of accuracy weighted by the importance of the input space.

The plot on the left in Figure 2 shows median performance of each method in terms of wRMSE with an increasing number of samples. The plot on the right shows the range of outcomes for each method at 100 total samples. Unsurprisingly, random sampling performs worse than all alternatives on average. The weighed methods generally outperform unweighted methods, although the difference in performance between the MED and the IMSPE approaches narrows as the sample size increases. The NU-IMSPE active learning approach with an MED start performs best overall compared to the alternatives. This should be expected since this is the only method which incorporates both model uncertainty and weights by the region of interest. The plot on the right does show an interesting result that the MED has a larger variation in performance.

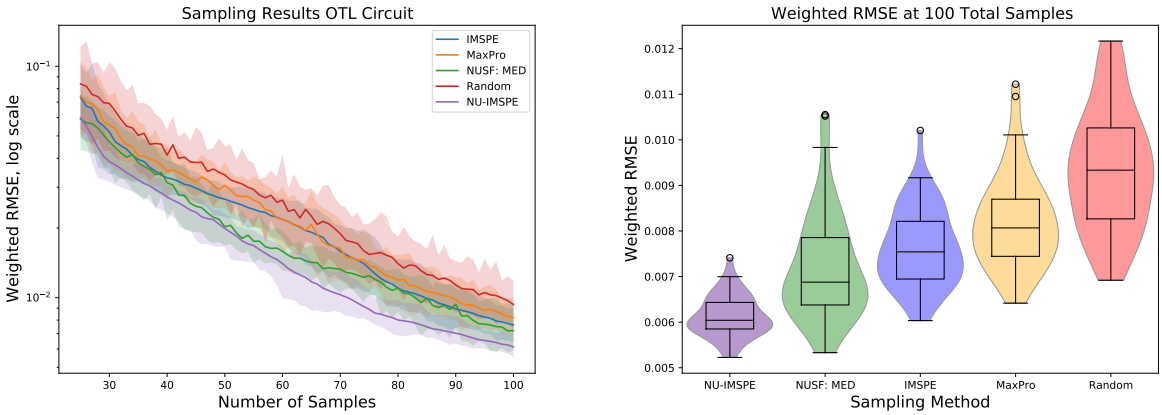

Figure 2: Results from OTL circuit sampling simulation study. The shaded bounds on the left contain the middle 95% of results for 50 random restarts.

4.2 STREAM Atmospheric Entry Modeling Example

This section applies the above techniques to a canonical atmospheric entry problem using LLNL's STREAM geometry. STREAM is a generic axi-symmetric sphere cone geometry. NASA's CBAERO (Kinney, 2007) was used to generate solutions and is advantageous for the simulation study as it is fast to generate data (at most 5 minutes per sample) compared to higher fidelity codes which can take many hours to days. The use of a low fidelity code mimics how NU-IMSPE could be used to assist in the analysis of a new geometry. The trajectory simulator takes tabularized force and moment coefficients (i.e., the aero database), tabularized atmospheric conditions that normalize those coefficients, and other initial conditions. A 4th order Runge Kutta scheme advances the rotational and translational equations of motion for a rigid body, generating a time history of the vehicle's trajectory.

Figure 3 displays the proposed application of the NU-IMSPE approach for a sampling workflow for aerodynamic database problems. Note, that the following approach is one way to construct the weight function, but in general the NU-IMSPE could be applied with an arbitrarily generated KDE weight function. The first computer code could be any CFD solver and the second computer code is a trajectory simulator. The trajectory simulator starts from some initial conditions and updates at discrete time steps using aero coefficients from the database. We propagate uncertainty from the aero-database through to the resulting trajectories by using GP draws and running the trajectory simulator multiple times. The values of the database used for each trajectory simulator (e.g. (Mach,Angle of Attack) combinations) are tracked during the simulation and these values are used to construct the KDE. This weight function captures the most commonly observed input combinations used by the trajectory simulator, and the NU-IMSPE sampling targets areas that will reduce the aero coefficient uncertainty subject to this weighting. Note that this design loop is initialized with a space-filling design which allows us to directly compare the sampling methods. It would be possible to use an initial weight function based on an analyst's prior knowledge using an NUSF design and this would possibly lead to even larger improvements. Alternatively, initial studies with a low-fidelity aerodynamics model could be used to quickly identify regions where higher fidelity methods should be targeted, but we leave this for future work.

For the sampling study, a reference set of 1,600 samples was generated over a rectangular range that envelopes those flown in feasible trajectories. The samples scaled to the range $[0, 1]^2$ are shown in Figure 4 and the resulting surfaces are shown in Figure 5. To generate the weight function for the non-uniform sampling a GP surrogate was fit over the large reference database and propagated forward into the flight simulator with samples over a range of potential initial conditions (Altitude, Speed, etc.). A KDE on the resulting trajectory data is used as the 'true' reference weight function and for computing the wRMSE. Also, note that this reference weight function was applied to the non-uniform sampling approaches to save computational effort and compare performance of the sampling methods directly. We will additionally run the full sampling

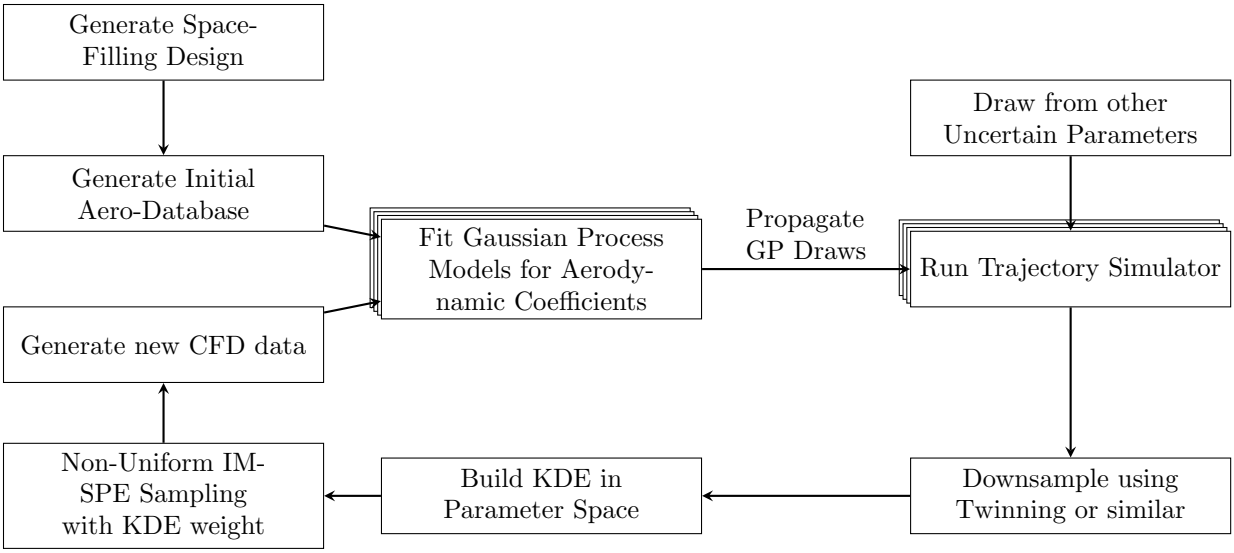

Figure 3: Diagram of the proposed sampling workflow.

loop generating new weight functions to inform the sampling at each iteration for the non-uniform IMSPE approach for comparison.

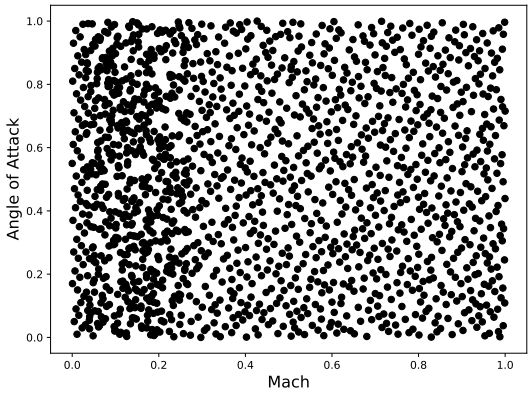

Figure 4: Plot of generated CBAERO samples.

The same methods from Section 4.1 above were applied with a few key difference in this study. When choosing an optimal design point the closest point in the reference dataset that has not already been sampled was used. Additionally, both the coefficient of Lift and Drag are of interest rather than a single output. For methods that are active learning based we used a simple average of the criteria values for each output. An extension of this could estimate and incorporate sensitivity of trajectory simulation into the weight function, but we leave fine-tuning the weight function when resolving multiple quantities of interest for future work. The output trajectory data to resolve the full 6DOF vehicle motion was overly dense and so data twinning (Vakayil and Joseph, 2022) was used to obtain a highly representative subsample. Finally, the Matérn 5/2 kernel was used in place of the Gaussian in section 4.1.

The results in Figure 6 show that the non-uniform methods show even larger improvement over uniform when compared to the toy example. The median performance at the full 50 samples is equivalent to the median NU-IMSPE at close to only 15 total samples. This is partially due to the available data spanning

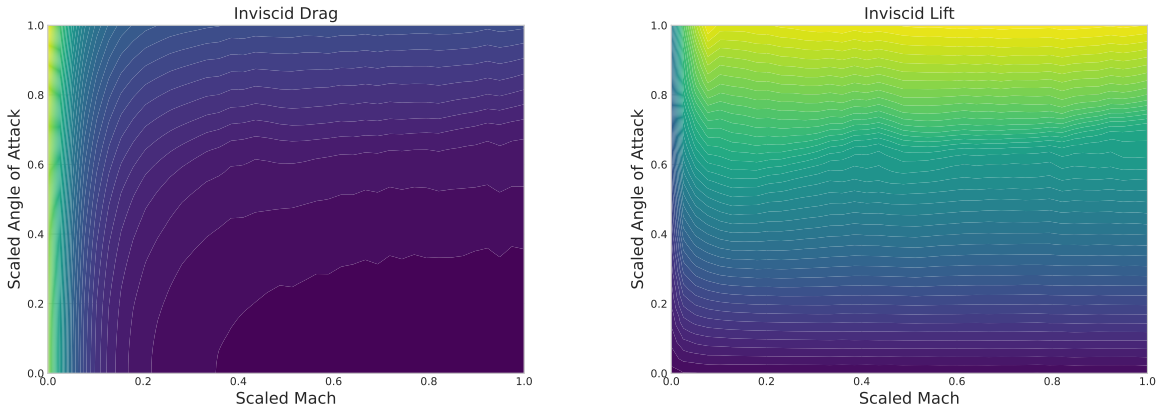

Figure 5: Surfaces for the coefficients of Drag and Lift for STREAM using design in Figure 4.

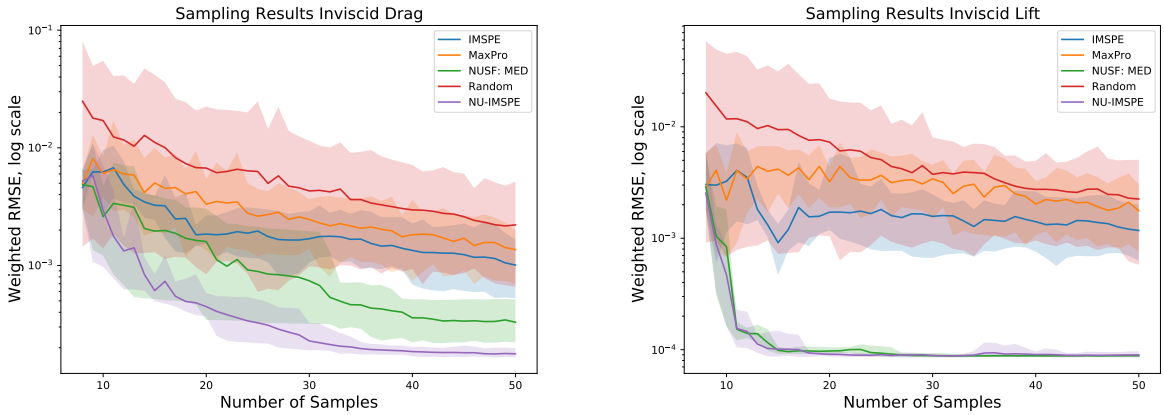

Figure 6: Results from the STREAM sampling study using 50 random restarts for each approach. The shaded bounds contain the middle 95% of results.

a much larger input space than necessary. Building a space-filling design over a smaller range would lead to better performance for non-weighted methods, but we note that the initial space was obtained from aerodynamicists; an overly large space is not uncommon, particularly for a candidate geometry. We therefore consider the speedups that were obtained to be fairly representative of what might be expected if these methods are applied to a new geometry. This gives greater support to adaptively building the aero-database. For Lift, the MED and NU-IMSPE sampling have highly similar performance. However, the NU-IMSPE achieves a much smaller error for the coefficient of Drag and is thus preferable in an overall sense.

Further comparison in Figure 7 shows the Continuous Rank Probability Score (CRPS) and log-likelihood of the models for the design choices. The CRPS is measured as $\sigma(-\pi^{-\frac{1}{2}} + 2\phi(z) + z(2\Phi(z) - 1))$ with ϕ and Φ the standard normal pdf and cdf respectively and $z = \frac{y-\mu}{\sigma}$ where y is a vector of the the observed values, μ is the predicted means and σ is the predicted standard deviations. CRPS is used to measure both the fit and uncertainty quantification of the model (Gneiting and Raftery, 2007). The closer the value is to 0 the better the score for CRPS. For this comparison we take a weighted average of the CRPS values (wCRPS) using the full CBAERO dataset as a reference. Figure 7 shows the performance for 50 random restarts. The wCRPS values maintain the same trends as in Figure 6 reflecting the improvement in the accuracy and also demonstrates that the NU-IMSPE and MED approaches lead to comparable accuracy in their uncertainty quantification. Additionally, we can observe the log-likelihood of the competing sampling approaches. Models

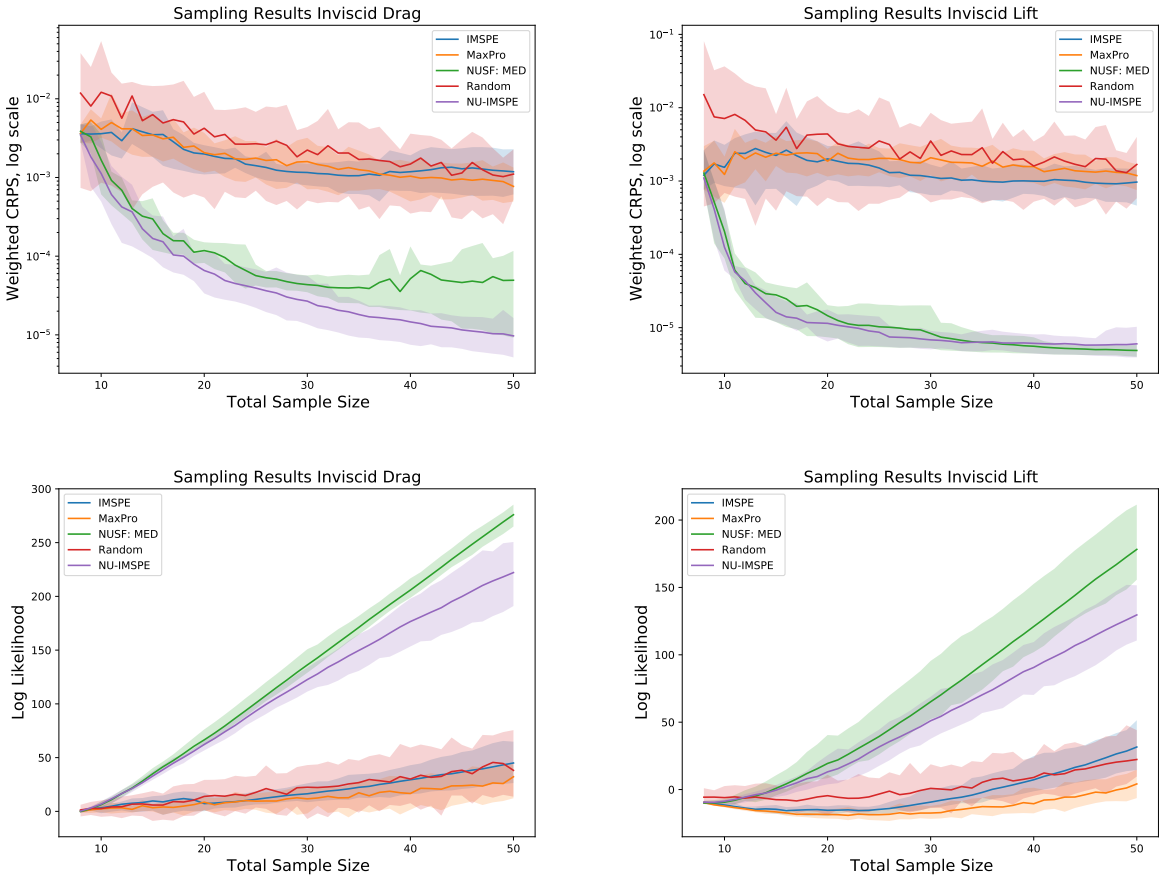

Figure 7: Additional results from the STREAM sampling study using 50 random restarts for each approach. The shaded bounds contain the middle 95% of results. (Top) Continuous Rank Probability Scores for drag and lift models. (Bottom) Log Likelihood for drag and lift models.

with the non-uniform sampling have a higher likelihood compared to the uniform approaches. Interestingly, the MED space-filling approach has a higher model likelihood, but this still leads to a poorer fit according to the other metrics. These results are likely a reflection of the input space explored. The MED approach tries to space-fill according to the weight function distribution, but this leads to less exploration of lower weighted regions of the input space. Some lower weighted areas contain non-stationary features (particularly lower Mach values) which would lower the overall likelihood of the model.

Next we compare the NU-IMSPE with a known weight function as displayed in Figure 6 to running the full loop in Figure 3 (referred to as estimated weight). For each sampling loop, 250 trajectories are generated over three uncertain parameters on the initial conditions. When generating trajectories for the weight function, we add one caveat that non-physical trajectories were removed from the weighting as draws from an initial coarsely sampled databases can sometimes lead to unrealistic behavior.

Overall, the results in Figure 8 show that running the full cycle as in Figure 3 results in highly similar improvements over other methods which is encouraging for applying this approach for future studies. The results are also interesting in that it appears that we actually performed slightly worse on average using the true weight function for Drag. However, the differences overall are relatively small and indicate that the trajectory simulation has mostly converged early in the process. The required accuracy for an aero-database for a given trajectory simulation is problem dependent and will be explored in future work. The differences in performance between the two approaches appear to be due to the wider weight function when emulating the

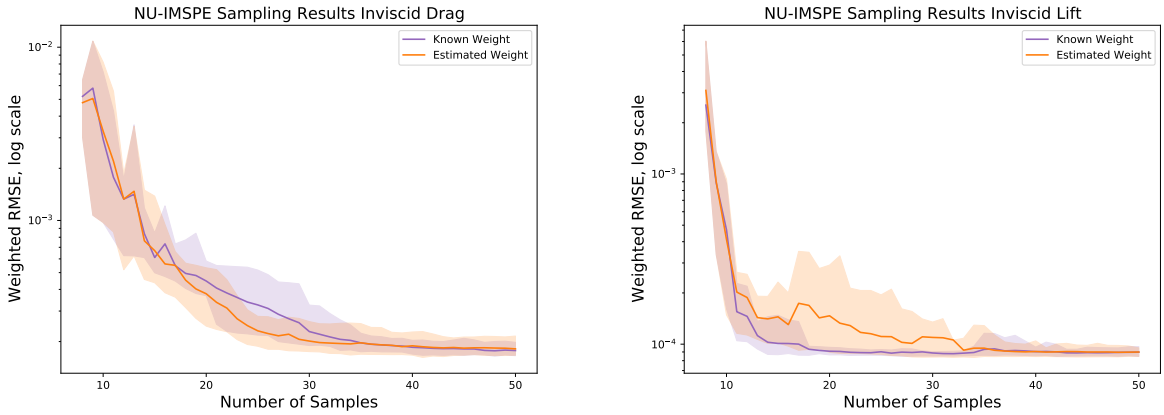

Figure 8: Results from the STREAM sampling study using 50 random restarts for each approach. The known weight is the same as the result from Figure 6 and the estimated weight uses interim trajectory information to generate the weight at each iteration of the sampling loop.

real process putting a heavier relative weight on lower Mach values than the known weight function. Lower Mach values have some slight non-stationarity as seen in Figure 5, and this is included earlier in the process when estimating the weight function. When emulating the true process, a slightly smaller lengthscale is estimated earlier which increased the error for Lift, and happened to give some improvement to the estimate for Drag. Finally, note that for this study, we simply added one sample at a time to make a better comparison to the study with a fixed weight function; however, in practice, it is likely we would add batches of samples as multiple CFD values could be computed in parallel.

5 Conclusions

In this work we have derived closed form solutions for a Non-Uniform IMSPE active learning criterion and demonstrated its effectiveness on two example problems. The NU-IMSPE approach showed improvements over a non-uniform space filling approach and both methods greatly outperformed uniformly weighted methods. The benefit of sampling non-uniformly shows great promise in speeding up turn around times for new vehicle studies.

For future work, we will extend and refine the sampling approach detailed here. There are typically uncertain parameters such as the initial conditions, or the discrepancies of the CFD model solutions that can be estimated when test data is available. By incorporating this information into the active learning loop we can more efficiently replicate a given scenario. A natural extension is to demonstrate this approach in a multi-fidelity setting. The techniques from Le Gratiet and Cannamela (2015) are directly applicable here; however the incorporation of a weight function introduces some new challenges. For example, adding data from a new high fidelity simulator may alter the weight function. Using a maximum weight ratio to limit the strength of the weight function such as in Lu and Anderson-Cook (2021) would be a promising approach to guide new samples. There may also be some expert knowledge on typical discrepancies of CFD solvers that could also be incorporated into the weight function. Finally, a common situation in aerodynamic modeling is building a trimmed aero-dynamic database. This problem is closely related to a contour estimation problem, and the closed form techniques described in this work could be extended to develop a closed form solution to the criterion in Picheny et al. (2010).

Acknowledgments

We would like to thank the reviewer and editor for their helpful comments which have improved this work. This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. LLNL-JRNL-850052-DRAFT

Appendix

5.1 V_N for Gaussian Kernel

We provide the closed form for the Gaussian kernel for the GP as well as a Gaussian kernel for the KDE

$$A = \frac{4l^2x_k^2 - 4x_jl^2x_k - 4x_il^2x_k + 2h^2x_j^2 - 2x_ix_jh^2 + 2h^2x_i^2 + 2l^2x_j^2 + 2l^2x_i^2}{4l^2(2h^2 + l^2)}$$

$$B = \frac{2l^2x_k + 2(x_i + x_j)h^2}{4l^2h^2\sqrt{\frac{2h^2+l^2}{2l^2h^2}}}$$

$$C = \frac{(x_k - 1)l^2 + (x_i + x_j - 2)h^2}{2l^2h^2\sqrt{\frac{2h^2+l^2}{/}2l^2h^2}}$$

$$V_N(x_i, x_j, x_k) = \sqrt{\pi}\frac{e^{-A}\text{erf}(B) - \text{erf}(C)}{2\sqrt{\frac{2h^2+l^2}{2h^2l^2}}}$$

5.2 V_N for Matèrn 5/2 Kernel

Due to the absolute value, the integrand is broken into three cases and assumes $0 < x_i < x_j < 1$ W.L.O.G. Thus we need the case where $0 < x_i$, $x_i < x_j$, and $x_j < 1$. The lower integral and upper integral share the same form due to symmetry (take the integral from 0 to $1 - x_j$). For simplicity let $c = \frac{\nu}{\sqrt{5}}$ and $d = 2h^2$ with h the lengthscale for the KDE and ν the lengthscale for the GP. The integral was too big to plug directly into solvers such as SymPy, so we first collect in terms of each polynomial of x_k and integrate each one separately. Finally, we leave a computational note that there is an ERV_{diff} term and using a simple call of the `erf` from Python's math package will result in a value of 0 for small differences, using the `erfc` and swapping the order provides the precision needed to evaluate this accurately.

For $0 < x_i$:

$C_s = \frac{1}{9c^4}$
$C_0 = ((x_i^2 + 3x_ic + 3c^2)(x_j^2 + 3x_jc + 3c^2))$
$C_1 = -(2x_i^2x_j + 3x_i^2c + 2x_ix_j^2 + 12x_ix_jc + 15x_ic^2 + 3x_j^2c + 15x_jc^2 + 18c^3)$
$C_2 = (x_i^2 + 4x_ix_j + 9x_ic + x_j^2 + 9x_jc + 15c^2)$
$C_3 = 2(-x_i - x_j - 3c)$
$C_4 = 1$
$ERV_{diff} = \text{erf}\,\frac{cx_k+d}{c\sqrt{d}} - \text{erf}\,\frac{-x_ic+cx_k+d}{c\sqrt{d}}$
$D_0 = -(\sqrt{\pi}\sqrt{d}\exp((c(2x_k - x_j - x_i) + d)/c^2)(ERV_{diff}))/2$ $f_a = (1/(2c))\sqrt{d}\exp(-(x_i^2c + x_id + x_jd + cx_k^2)/(cd))$
$f_b = \sqrt{\pi}(cx_k + d)\exp((x_i^2 + (cx_k + d)^2/c^2)/(d))(ERV_{diff})$
$f_c = c\sqrt{d}(\exp(x_i^2/d) - \exp(2x_i((1/c) + (x_k/d))))$
$D_1 = f_a(f_b + f_c)$
$s_a = (1/(4c^2))\exp(-(x_id + x_jd + cx_k^2)/(cd))$
$s_b = \sqrt{\pi}\sqrt{d}\exp((cx_k + d)^2/(c^2d))(c^2(d + 2x_k^2) + 4cdx_k + 2d^2)ERV_{diff}$
$s_c = 2cd(cx_k + d) - 2cd\exp((x_i(2(cx_k + d) - x_ic))/(cd))(c(x_i + x_k) + d)$
$D_2 = s_a(s_b + s_c)$
$t_a = 1/(4c^3)\exp(-(x_id + x_jd + cx_k^2)/(cd))$
$t_b = \sqrt{\pi}\sqrt{d}\exp((cx_k + d)^2/(c^2d))(c^3(3dx_k + 2x_k^3) + 3c^2d(d + 2x_k^2) + 6cd^2x_k + 2d^3)ERV_{diff}$

$t_c = -2cd\exp((x_i(2(cx_k+d)-x_ic))/(cd))(c^2(x_i^2+x_ix_k+d+x_k^2)+cd(x_i+2x_k)+d^2)+2cd(c^2(d+x_k^2)+2cdx_k+d^2)$
$D_3 = ta(tb+tc)$
$r_a = 1/(8c^4)\exp(-(x_id+x_jd+cx_k^2)/(cd))$
$r_b = \sqrt{\pi}\sqrt{d}\exp((cx_k+d)^2/(c^2d))(c^4(3d^2+12dx_k^2+4x_k^4)+8c^3dx_k(3d+2x_k^2)+12c^2d^2(d+2x_k^2)+16cd^3x_k+4d^4)ERV_{diff}$
$r_c = 2cd(c^3(5dx_k+2x_k^3)+c^2d(5d+6x_k^2)+6cd^2x_k+2d^3)$
$r_d = -2cd\exp((x_i(2(cx_k+d)-x_ic))/(cd))(c^2d(2x_i^2+4x_ix_k+5d+6x_k^2)+c^3(2(x_i^3+x_i^2x_k+x_ix_k^2+x_k^3)+d(3x_i+5x_k))+2cd^2(x_i+3x_k)+2d^3)$
$D_4 = r_a(r_b+r_c+r_d)$
$V_{N,\text{low}}(x_i,x_j,x_k) = C_s\sum_{q=0}^4 C_qD_q$

For $x_i < x_j$:
$C_s = \frac{1}{9c^4}$
$C_0 = (x_i^2-3x_ic+3c^2)(x_j^2+3x_jc+3c^2)$
$C_1 = (-2x_i^2x_j-3x_i^2c-2x_ix_j^2+3x_ic^2+3x_j^2c+3x_jc^2)$
$C_2 = (x_i^2+4x_ix_j+3x_ic+x_j^2-3x_jc-3c^2)$
$C_3 = 2(-x_i-x_j)$
$C_4 = 1$
$ERV_{diff} = \text{erf}\,\frac{r-x_i}{\sqrt{d}} - \text{erf}\,\frac{r-x_j}{\sqrt{d}}$
$D_0 = 0.5\sqrt{\pi}\sqrt{d}\exp((x_i-x_j)/c)ERV_{diff}$
$f_a = 0.5\sqrt{d}\exp((x_i-x_j)/c)$
$f_b = \sqrt{\pi}x_k(ERV_{diff})$
$f_c = \sqrt{d}(\exp(-(x_i-x_k)^2/d)-\exp(-(x_j-x_k)^2/d))$
$D_1 = f_a(f_b+f_c)$
$s_a = 0.25\sqrt{d}\exp((x_i-x_j)/c)$
$s_b = \sqrt{\pi}(d+2x_k^2)(ERV_{diff})$
$s_c = 2\sqrt{d}((x_i+x_k)\exp(-(x_i-x_k)^2/d)-(x_j+x_k)\exp(-(x_j-x_k)^2/d))$
$D_2 = s_a(s_b+s_c)$
$t_a = 0.25\sqrt{d}\exp((x_i-x_j)/c)$
$t_b = \sqrt{\pi}x_k(3d+2x_k^2)(ERV_{diff})$
$t_c = 2\sqrt{d}(\exp(-(x_i-x_k)^2/d)(x_i^2+x_ix_k+d+x_k^2)-\exp(-(x_j-x_k)^2/d)(x_j^2+x_jx_k+d+x_k^2))$
$D_3 = t_a(t_b+t_c)$
$r_a = (1/8.0)\sqrt{d}\exp((x_i-x_j)/c)$
$r_b = \sqrt{\pi}(3d^2+12dx_k^2+4x_k^4)(ERV_{diff})$
$r_c = 2.0\sqrt{d}(\exp(-(x_i-x_k)^2/d)(2(x_i+x_k)(x_i^2+x_k^2)+d(3x_i+5x_k))-\exp(-(x_j-x_k)^2/d)(2(x_j+x_k)(x_j^2+x_k^2)+d(3x_j+5x_k)))$
$D_4 = r_a(r_b+r_c)$
$V_{N,\text{mid}}(x_i,x_j,x_k) = C_s\sum_{q=0}^4 C_qD_q$

References

Ba, S. and Joseph, V. R. (2018). *MaxPro: Maximum Projection Designs*. R package version 4.1-2.

Ben-Ari, E. N. and Steinberg, D. M. (2007). Modeling data from computer experiments: an empirical comparison of kriging with mars and projection pursuit regression. *Quality Engineering*, 19(4):327–338.

Binois, M. and Gramacy, R. B. (2021). hetgp: Heteroskedastic gaussian process modeling and sequential design in r.

Binois, M., Huang, J., Gramacy, R. B., and Ludkovski, M. (2019). Replication or exploration? sequential design for stochastic simulation experiments. *Technometrics*, 61(1):7–23.

Bowman, V. E. and Woods, D. C. (2013). Weighted space-filling designs. *Journal of Simulation*, 7(4):249–263.

Burchett, B. T., Paul, J. L., Vasile, J. D., and Bryson, J. (2022). A high fidelity roll dependent aerodynamic model for a long range, high speed missile. In *AIAA SCITECH 2022 Forum*, page 0420.

Cohn, D. (1993). Neural network exploration using optimal experiment design. *Advances in neural information processing systems*, 6.

Gneiting, T. and Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378.

Gramacy, R. B. (2020). *Surrogates: Gaussian process modeling, design, and optimization for the applied sciences*. Chapman and Hall/CRC.

Gramacy, R. B. and Lee, H. K. (2012). Cases for the nugget in modeling computer experiments. *Statistics and Computing*, 22(3):713–722.

Haario, H., Saksman, E., and Tamminen, J. (1999). Adaptive proposal distribution for random walk metropolis algorithm. *Computational statistics*, 14(3):375–395.

Johnson, M. E., Moore, L. M., and Ylvisaker, D. (1990). Minimax and maximin distance designs. *Journal of statistical planning and inference*, 26(2):131–148.

Joseph, V. R., Dasgupta, T., Tuo, R., and Wu, C. J. (2015a). Sequential exploration of complex surfaces using minimum energy designs. *Technometrics*, 57(1):64–74.

Joseph, V. R., Gul, E., and Ba, S. (2015b). Maximum projection designs for computer experiments. *Biometrika*, 102(2):371–380.

Joseph, V. R., Wang, D., Gu, L., Lyu, S., and Tuo, R. (2019). Deterministic sampling of expensive posteriors using minimum energy designs. *Technometrics*.

Kinney, D. (2007). Aerothermal anchoring of cbaero using high fidelity cfd. In *45th AIAA Aerospace Sciences Meeting and Exhibit*, page 608.

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimization by simulated annealing. *science*, 220(4598):671–680.

Kyzyurova, K. N., Berger, J. O., and Wolpert, R. L. (2018). Coupling computer models through linking their statistical emulators. *SIAM/ASA Journal on Uncertainty Quantification*, 6(3):1151–1171.

Le Gratiet, L. and Cannamela, C. (2015). Cokriging-based sequential design strategies using fast cross-validation techniques for multi-fidelity computer codes. *Technometrics*, 57(3):418–427.

Leatherman, E. R., Santner, T. J., and Dean, A. M. (2018). Computer experiment designs for accurate prediction. *Statistics and Computing*, 28(4):739–751.

Lu, L. and Anderson-Cook, C. M. (2021). Strategies for sequential design of experiments and augmentation. *Quality and Reliability Engineering International*, 37(5):1740–1757.

Lu, L., Anderson-Cook, C. M., and Ahmed, T. (2021). Non-uniform space filling (nusf) designs. *Journal of Quality Technology*, 53(3):309–330.

Mak, S. and Joseph, V. R. (2018). Support points. *The Annals of Statistics*, 46(6A):2562–2592.

Petersen, K. B., Pedersen, M. S., et al. (2008). The matrix cookbook. *Technical University of Denmark*, 7(15):510.

Picheny, V., Ginsbourger, D., Roustant, O., Haftka, R. T., and Kim, N.-H. (2010). Adaptive designs of experiments for accurate approximation of a target region.

Quinlan, K. R., Movva, J., Stein, E. V., and Kupresanin, A. (2021). Leveraging multi-fidelity aerodynamic databasing to efficiently represent a hypersonic design space. In *ASCEND 2021*, page 4245.

Sacks, J., Schiller, S. B., and Welch, W. J. (1989a). Designs for computer experiments. *Technometrics*, 31(1):41–47.

Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989b). Design and analysis of computer experiments. *Statistical science*, pages 409–423.

Santner, T. J., Williams, B. J., Notz, W. I., and Williams, B. J. (2003). *The design and analysis of computer experiments*, volume 1. Springer.

Vakayil, A. and Joseph, V. R. (2022). Data twinning. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 15(5):598–610.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

Zhang, B., Gramacy, R. B., Johnson, L., Rose, K. A., and Smith, E. (2020). Batch-sequential design and heteroskedastic surrogate modeling for delta smelt conservation. *arXiv preprint arXiv:2010.06515*.