SAND2020-3677R

PREPRINT

Weighted greedy-optimal design of computer experiments for kernel-based and Gaussian process model emulation and calibration

H. Harbrecht[1], J.D. Jakeman[2], and P. Zaspel[3]

[1]Department of Mathematics and Computer Science, University of Basel, Basel, Switzerland.
[2]Optimization and Uncertainty Quantification, Sandia National Laboratories, Albuquerque, NM, 87123.
[3]Computer Science and Electrical Engineering, Jacobs University Bremen, Bremen, Germany.

ABSTRACT

This article is concerned with the approximation of high-dimensional functions by kernel-based methods. Motivated by uncertainty quantification, which often necessitates the construction of approximations that are accurate with respect to a probability density function of random variables, we aim at minimizing the approximation error with respect to a weighted L^p-norm. We present a greedy procedure for designing computer experiments based upon a weighted modification of the pivoted Cholesky factorization. The method successively generates nested samples with the goal of minimizing error in regions of high probability. Numerical experiments validate that this new importance sampling strategy is superior to other sampling approaches, especially when used with non-product probability density functions. We also show how to use the proposed algorithm to efficiently generate surrogates for inferring unknown model parameters from data.

ARTICLE INFO

Correspondence:
J.D. jakeman
jdjakem@sandia.gov

Keywords:
Uncertainty quantification, radial basis function, modeling, simulation, surrogate, Gaussian process, experimental design

1 Introduction

This article considers the approximation of a function $u : \Gamma \to \mathbb{R}$ using kernel based interpolation. We are particularly interested in the case where u belongs to a (weighted) Hilbert function space $\mathcal{V}_\omega$ and the argument to the function is a finite-dimensional random variable $\boldsymbol{y}$ with associated probability density function (PDF) $\omega : \Gamma \to \mathbb{R}$ that introduces the weighting to $\mathcal{V}_\omega$. Therefore, we focus on constructing approximations $\Pi_V(u)$ of u, with respect to the finite-dimensional subspace $V \subset \mathcal{V}_\omega$, which have a small approximation error

$$\epsilon_\omega(u, V, p) := \left(\int_\Gamma |u(\boldsymbol{y}) - (\Pi_V u)(\boldsymbol{y})|^p \omega(\boldsymbol{y}) \mathrm{d}\boldsymbol{y} \right)^{1/p}. \tag{1}$$

In other words, we aim at minimizing the approximation error with respect to the ω-weighted $L^p(\Gamma)$-norm, that is

$$\|u\|_{L^p_\omega(\Gamma)} := \left(\int_\Gamma |u(\boldsymbol{y})|^p \omega(\boldsymbol{y}) \mathrm{d}\boldsymbol{y} \right)^{1/p}.$$

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Such function approximation problems often show up in applications of uncertainty quantification (UQ) and Bayesian inference. In these applications u usually corresponds to the parameter-to-solution map of a partial differential equation and the approximation is constructed to reduce the number of computationally expensive simulations (i.e. point evaluations of u).

Numerous techniques have been developed to build such approximations [22]. Within the computational science and engineering community, some of the most widely adopted methods for approximating models are those based on generalized polynomial chaos expansions [9, 40], sparse grid approximation [24, 39], Gaussian process models [28] and low-rank tensor decompositions [12, 25]. These methods can be very efficient when building approximations to functions of independent random variables. However, there is a dearth of algorithmic options for when the variables are dependent.

The objective of this article is to propose a methodology to efficiently generate nested and (greedy-)optimal samples for a given probability density ω function, which requires a small number of evaluations of u to build an approximation with a pre-specified accuracy measured in the L^p_ω norm. Accurate approximations can be built without tailoring the sampling and approximation strategy to the PDF ω. However such approaches, called *domination methods* [2, 14], are suboptimal and require larger number of evaluations of u than methods that consider ω [15]. Instead of building an approximation which minimizes the error measured by the L^p_ω norm, *domination methods* build an approximation that minimizes the error measured by another norm L^p_g, where g is a simpler PDF. Typically g is simply a constant, i.e. the PDF of independent bounded uniform variables. The use of the simpler incorrect PDF g allows the use of existing unweighted approximation methods, but comes at a cost [2, 14, 40].

The following lemma characterizes the accuracy in a ω-weighted norm given an approximation that is accurate in the g-weighted norm.

Theorem 1.1 (Strong convergence [2]). *Let $g : \hat{\Gamma} \to \mathbb{R}$ and $\omega : \Gamma \to \mathbb{R}$ denote two densities which satisfy*

$$\delta = 1 - \int_{\Gamma \cap \hat{\Gamma}} \omega(\boldsymbol{y}) \mathrm{d}\boldsymbol{y}$$

Now assume that the error of the approximation $\Pi_g(u)$ of u satisfies

$$\epsilon := \|u - \Pi_g u\|_{L^p_g(\Gamma)}, \quad p \geq 1, \tag{2}$$

and that u is bounded with $C_u = \|u\|_{L^\infty(\Gamma)}$ Then, there holds

$$\|u - \Pi_g u\|_{L^p_\omega(\Gamma)} \leq C_r^{1/p}\epsilon + C_u \delta^{1/p}, \quad \textit{provided} \quad C_r := \max_{\boldsymbol{y} \in \Gamma \cup \hat{\Gamma}} \frac{\omega(\boldsymbol{y})}{g(\boldsymbol{y})} < \infty. \tag{3}$$

This theorem states, that tailoring an approximation to a PDF different from ω degrades approximation accuracy by a constant C_r. The constant C_r is characterized as the maximum deviation of g from the original PDF ω. If Γ is unbounded, we can still apply approximation methods for bounded domains $\hat{\Gamma}$. However, this induces an additional increase in the error which is proportional to the probability δ (measured with respect to ω) of $\boldsymbol{y}$ falling outside the bounded domain.

In meshfree kernel-based approximation, u is often evaluated at samples drawn from quasi-random number sequences. However, these sequences were primarily designed for unweighted function approximation, i.e. they are good samples for $\mathcal{V}_g$ with g being a PDF of a uniformly distributed random variable. Theorem 1.1 suggests that we can significantly reduce the approximation error if we instead use samples optimized for the PDF of interest ω.

In this article we present a ω-aware approximation technique based on nested greedy-optimal sampling using a weighted version of the so-called *power function*, i.e. the worst case error functional in kernel-based approximation. The power function has been used to construct efficient sampling schemes for unweighted approximation [30]. Our new contribution is to employ a weighting in the sample allocation process, which adapts the samples to the probability measure ω. We will show that this approach outperforms approximations constructed using classical power function based sampling and quasi Monte Carlo sampling. We also compare our approach with methods based upon Quasi Monte Carlo sequences and integrated variance (IVAR) experimental design [11] used by the Gaussian process community.

When ω is a tensor-product measure, our approach produces approximations with accuracy comparable to approximations built using samples obtained from inverse transform sampling [6, 16] applied to quasi

Monte Carlo sequences. However, the latter approach can only be applied easily to tensor-product measures, while our approach is also applicable to non-product measures, for example that occur in Bayesian inference problems. We also demonstrate that our approach produces approximations with similar accuracy to those constructed using IVAR. However, the latter is much more computationally intensive and difficult to implement. Our approach only requires a simple linear algebra task (i.e. the pivoted Cholesky factorization), while IVAR requires an expensive non-convex optimization.

The rest of this article is organized as follows. In Section 2, we provide an overview of kernel-based function approximation. Then, in Section 3, we introduce a new approach for greedily generating samples for kernel based approximation based upon a weighted modification of pivoted Cholesky factorization. We especially discuss the similarities and differences between our new approach and existing sampling strategies used for kernel based approximation. Finally, in Section 4, we provide numerical results that highlight the strengths and performance of our new approach.

2 Kernel-based function approximation

This section provides a summary of radial basis function approximation using scattered data. Our exposition is similar to the discussion on function approximation in finite-dimensional *reproducing kernel Hilbert spaces* found in [35].

2.1 Reproducing kernel Hilbert spaces

Definition 1 (Reproducing kernel Hilbert space). *A reproducing kernel $\mathcal{K}$ for a general Hilbert space $\mathcal{V}$ with inner product $(\cdot,\cdot)_{\mathcal{V}}$ is a function $\mathcal{K}:\Gamma\times\Gamma\to\mathbb{R}$ such that*

1. *$\mathcal{K}(\cdot,\boldsymbol{y})\in\mathcal{V}$ for all $\boldsymbol{y}\in\Gamma$,*
2. *$u(\boldsymbol{y})=(u,\mathcal{K}(\cdot,\boldsymbol{y}))_{\mathcal{V}}$ for all $u\in\mathcal{V}$ and all $\boldsymbol{y}\in\Gamma$.*

A Hilbert space $\mathcal{V}$ with reproducing kernel $\mathcal{K}:\Gamma\times\Gamma\to\mathbb{R}$ is called reproducing kernel Hilbert space (RKHS).

A continuous kernel $\mathcal{K}:\Gamma\times\Gamma\to\mathbb{R}$ is denoted *positive semi-definite* on $\Gamma\subseteq\mathbb{R}^d$, if we have

$$\sum_{j=1}^{N}\sum_{j'=1}^{N}\alpha_j\alpha_{j'}\mathcal{K}(\boldsymbol{y}_j,\boldsymbol{y}_{j'})\geq 0 \tag{4}$$

for all $N\in\mathbb{N}$, all pairwise distinct $X=\{\boldsymbol{y}_1,\dots,\boldsymbol{y}_N\}\subset\Gamma$, and all $\alpha\in\mathbb{R}^N\setminus\{0\}$. It becomes *positive definite* if the inequality in (4) holds strictly.

In this article, we use radial basis functions, i.e. *radial kernels*

$$\mathcal{K}(\boldsymbol{y},\boldsymbol{y}'):=\varphi(\|\boldsymbol{y}-\boldsymbol{y}'\|) \tag{5}$$

with $\varphi:[0,\infty)\to\mathbb{R}$. A typical example of such a radial kernel is the *Gaussian / squared exponential kernel*, with

$$\varphi(r)=e^{-\epsilon^2 r^2}. \tag{6}$$

The method we describe can be applied to a broad set of other kernels including the the class of *Matérn kernels* given by

$$\varphi(r)=\frac{K_{\beta-\frac{d}{2}}(r)r^{\beta-\frac{d}{2}}}{2^{\beta-1}\Gamma(\beta)},\quad \beta>\frac{d}{2}.$$

Here K_ν being the modified Bessel function of the second kind of order ν and Γ is the gamma function. Note that the parameter ν dictates for the smoothness of the kernel function.

The first property of Definition 1 implies that $\mathcal{V}$ contains all functions of the form

$$u=\sum_{j=1}^{N}\alpha_j\mathcal{K}(\cdot,\boldsymbol{y}_j) \tag{7}$$

provided that the points $\{\boldsymbol{y}_1, \dots, \boldsymbol{y}_N\}$ satisfy $\boldsymbol{y}_j \in \Gamma$. Vice versa, we can construct the *native space* $\mathcal{N}_\mathcal{K}(\Gamma)$ for a given symmetric, positive definite kernel $\mathcal{K}$ by completion from the pre-Hilbert space

$$F_\mathcal{K}(\Gamma) := \operatorname{span}\{\mathcal{K}(\cdot, \boldsymbol{y}) : \boldsymbol{y} \in \Gamma\}.$$

It is shown in [35] that $\mathcal{N}_\mathcal{K}(\Gamma)$ is a RHKS for $\mathcal{K}$. For functions of the form (7), the native space carries the norm induced from the inner product

$$\left(\sum_{j=1}^N \alpha_j \mathcal{K}(\cdot, \boldsymbol{y}_j), \sum_{j'=1}^N \beta_{j'} \mathcal{K}(\cdot, \boldsymbol{y}_{j'})\right)_{\mathcal{N}_\mathcal{K}(\Gamma)} := \sum_{j=1}^N \sum_{j'=1}^N \alpha_j \beta_{j'} \mathcal{K}(\boldsymbol{y}_j, \boldsymbol{y}_{j'}).$$

For example, the native space $\mathcal{N}_\mathcal{K}(\Gamma)$ of the Matérn kernel with $\beta > d/2$ and $\Gamma = \mathbb{R}^d$ is the well-known Sobolev space $H^\beta(\mathbb{R}^d)$.

2.2 Function approximation

The focus on this paper is the approximation of functions u that are elements of the native space of a kernel $\mathcal{K}$, i.e. $u \in \mathcal{N}_\mathcal{K}(\Gamma)$. With this goal we introduce a finite-dimensional approximation subspace of $\mathcal{N}_\mathcal{K}(\Gamma)$ by choosing a finite set of sampling points

$$X = \{\boldsymbol{y}_1, \dots, \boldsymbol{y}_N\} \subset \Gamma,$$

resulting in the finite-dimensional subspace

$$V(X) := \operatorname{span}\{\mathcal{K}(\cdot, \boldsymbol{y}) : \boldsymbol{y} \in X\} \subset \mathcal{N}_\mathcal{K}(\Gamma).$$

We are then interested in constructing a 'good' orthogonal projection $\Pi_{V(X)}(u)$ of $u \in \mathcal{N}_\mathcal{K}(\Gamma)$ to $V(X)$.

Interpolation. In reproducing kernel Hilbert spaces, the best-approximation, given by the orthogonal projection $\Pi_{V(X)}$, and the *interpolation* of a function $u \in \mathcal{N}_\mathcal{K}(\Gamma)$ with given data $\{(\boldsymbol{y}_i, u(\boldsymbol{y}_i))\}_{i=1}^N$ turns out to be identical [35]. That is, we observe for the interpolant $\mathcal{I}_{V(X)}u$ the equality

$$(\mathcal{I}_{V(X)}u)(\boldsymbol{y}) = \Pi_{V(X)}(f)(\boldsymbol{y}) := \sum_{j=1}^N \alpha_j \mathcal{K}(\boldsymbol{y}, \boldsymbol{y}_j) \quad \text{for all} \quad \boldsymbol{y} \in \Gamma,$$

where the coefficients $\{\alpha_j\}_{j=1}^N, \alpha_j \in \mathbb{R}$, are computed by solving a linear system of equations with

$$\boldsymbol{A}_X \boldsymbol{\alpha} = \boldsymbol{u}, \quad \boldsymbol{\alpha} := (\alpha_1 \dots \alpha_N)^\top, \quad \boldsymbol{u} := \big(u(\boldsymbol{y}_1) \dots u(\boldsymbol{y}_N)\big)^\top \tag{8}$$

and the kernel matrix

$$\boldsymbol{A}_X := \begin{pmatrix} \mathcal{K}(\boldsymbol{y}_1, \boldsymbol{y}_1) & \dots & \mathcal{K}(\boldsymbol{y}_1, \boldsymbol{y}_N) \\ \vdots & \ddots & \vdots \\ \mathcal{K}(\boldsymbol{y}_N, \boldsymbol{y}_1) & \dots & \mathcal{K}(\boldsymbol{y}_N, \boldsymbol{y}_N) \end{pmatrix}. \tag{9}$$

For positive definite kernels, $\boldsymbol{A}_X$ is symmetric, positive definite and regular.

Worst-case error. The *power function* [29, 37] or *kriging variance* [8] for a subspace $V(X)$ is, following [35], the interpolation / approximation worst case error

$$P_{V(X)}(\boldsymbol{y}) = \sup_{f \in \mathcal{N}_\mathcal{K}(\Gamma),\, f \neq 0} \frac{|f(\boldsymbol{y}) - (\mathcal{I}_{V(X)}f)(\boldsymbol{y})|}{\|f\|_{\mathcal{N}_\mathcal{K}(\Gamma)}}. \tag{10}$$

The power function gives us a simple estimate for the interpolation error.

Theorem 2.1 (Power function interpolation error [7]). *Let Γ, $\mathcal{K}$, X be as before and we have $f \in \mathcal{N}_{\mathcal{K}}(\Gamma)$ with its interpolant $\mathcal{I}_{V(X)}(f)$ on X. Then, there holds the error bound*

$$|f(\boldsymbol{y}) - \mathcal{I}_{V(X)}(\boldsymbol{y})| \leq P_{V(X)}(\boldsymbol{y}) \|f\|_{\mathcal{N}_{\mathcal{K}}(\Gamma)}$$

for all $\boldsymbol{y} \in \Gamma$.

The proof is trivial based on the above definition of the power function. Following Theorem 2.1, the upper bound for the interpolation error decouples in a product of the norm of the function and a term which only depends on the particular kernel and the samples in X.

Computing the power function using (10) appears difficult. However the power function can easily be evaluated using the following theorem.

Theorem 2.2 ([7]). *For two sets X, X' with elements $\boldsymbol{y}_j$ and $\boldsymbol{y}'_{j'}$ and sizes $|X| = N$, $|X'| = N'$, we use the extended notation*

$$\boldsymbol{A}_{X,X'} := \begin{pmatrix} \mathcal{K}(\boldsymbol{y}_1, \boldsymbol{y}'_1) & \dots & \mathcal{K}(\boldsymbol{y}_1, \boldsymbol{y}'_{N'}) \\ \vdots & \ddots & \vdots \\ \mathcal{K}(\boldsymbol{y}_N, \boldsymbol{y}'_1) & \dots & \mathcal{K}(\boldsymbol{y}_n, \boldsymbol{y}'_{N'}) \end{pmatrix}. \tag{11}$$

Then, the power function (10) can numerically be evaluated by

$$P_{V(X)}(\boldsymbol{y}) = \sqrt{\mathcal{K}(\boldsymbol{y}, \boldsymbol{y}) - \boldsymbol{A}_{X,\{\boldsymbol{y}\}}^{\top} \boldsymbol{A}_X^{-1} \boldsymbol{A}_{X,\{\boldsymbol{y}\}}}. \tag{12}$$

Note that the problem of finding a point set $X \subset \Gamma$ which minimizes the power function corresponds to approximating a given matrix by a low-rank approximation. Finding the best possible point set is therefore strongly related to the search for submatrices of maximal volume, see [10], and to adaptive cross approximation [1, 13].

Remark 2.1. *For large sets X, the evaluation of the power function by (12) can become unstable. A stable evaluation becomes possible by using the Newton basis of $V(X)$. We will discuss this procedure in Section 3.*

Regularized function approximation. For large N, $\boldsymbol{A}_X$ often becomes ill-conditioned. This is why we usually consider a regularized function approximation. In regularized kernel-based approximation, which is sometimes also called *kernel ridge regression*, Tikhonov regularization [33] is used. It replaces the original linear system of equations by a modified, but more stable one

$$\big(\boldsymbol{A}_X + \epsilon_{reg} \boldsymbol{I}\big)\boldsymbol{\alpha} = \boldsymbol{u},$$

where $\boldsymbol{I}$ the identity matrix. Depending on the size of the regularization parameter ϵ_{reg}, the matrix $\boldsymbol{A}_X + \epsilon_{reg}\boldsymbol{I}$ can have a much smaller condition number. Nonetheless, regularization introduces a small error of the order of the regularization parameter ϵ_{reg}. In this paper we set $\epsilon_{reg} = 10^{-12}$.

2.3 Relation to Gaussian process regression

Gaussian process (GP) regression can be used for function approximation in a Bayesian setting [26, 28, 36]. Let us assume that for a given function $u : \Gamma \to \mathbb{R}$, we have a set of samples evaluation locations $X = \{\boldsymbol{y}_1, \dots, \boldsymbol{y}_N\}$ and *noisy* evaluations $\hat{u}_i := u(\boldsymbol{y}_i) + \xi_i$, $\xi_i \sim \mathcal{N}(0, \sigma^2)$, collected in a vector $\hat{\boldsymbol{u}}$. Given a prior mean function $m_0(\boldsymbol{y})$ and a covariance kernel $\mathcal{K}(\boldsymbol{y}, \boldsymbol{y}')$ we can construct an approximation by conditioning this the Gaussian process prior on the available data. The resulting posterior distribution of the Gaussian process is given by

$$u|\boldsymbol{y}, X, \hat{\boldsymbol{u}} \sim \mathcal{N}\big(m(\boldsymbol{y}), C(\boldsymbol{y})\big).$$

The posterior mean

$$m(\boldsymbol{y}) = m_0(\boldsymbol{y}) + \boldsymbol{\alpha}^{\top} \boldsymbol{A}_{X,\{\boldsymbol{y}\}}$$

it typically used as an approximation of u and the covariance

$$C(\boldsymbol{y}) = \mathcal{K}(\boldsymbol{y}, \boldsymbol{y}) - \boldsymbol{A}_{X,\{\boldsymbol{y}\}}^{\top} \big(\boldsymbol{A}_X + \sigma^2 \boldsymbol{I}\big)^{-1} \boldsymbol{A}_{X,\{\boldsymbol{y}\}}. \tag{13}$$

as a indication of the error in that approximation. Construction of the Gaussian posterior mean requires estimation of the coefficients $\boldsymbol{\alpha}$. Using the notation $\boldsymbol{A}_{X,\{\boldsymbol{y}\}}$ from equation (11), the vector $\boldsymbol{\alpha}$ is given by

$$\boldsymbol{\alpha} := \big(\boldsymbol{A}_X + \sigma^2 \boldsymbol{I}\big)^{-1} \big(\hat{\boldsymbol{u}} - m_0(\boldsymbol{y})\big).$$

Note that for $m_0 \equiv 0$, the posterior mean in Gaussian process regression corresponds to regularized function approximation with the same kernel function $\mathcal{K}$ and with a Tikhonov regularization with $\epsilon_{reg} = \sigma^2$.

In general, the kernel contains hyper-parameters $\boldsymbol{\theta}$ such as the length-scale, the signal variance, and the noise variance, which are unknown and need thus to be inferred from the data. The beauty of Gaussian process regression is that it automatically provides a tool for hyper-parameter optimization by means of the numerical maximization of the log-marginal-likelihood

$$\log p(\hat{\boldsymbol{u}}|X,\boldsymbol{\theta}) = -\frac{1}{2}\hat{\boldsymbol{u}}^\top (\boldsymbol{A}_X + \sigma^2\boldsymbol{I})^{-1}\hat{\boldsymbol{u}} - \frac{1}{2}\log|\boldsymbol{A}_X + \sigma^2\boldsymbol{I}| - \frac{N}{2}\log 2\pi\,.$$

We will use this technique in our numerical results.

3 Sampling strategies

In this section, we present various sampling strategies for kernel-based approximation and Gaussian process regression. We first present an existing approach for nested unweighted-greedy optimal sampling using the power function, cf. (10). We next extend this approach to generate samples which minimize the weighted error functional (1) of the resulting approximations. These two methods are then compared with popular sampling strategies from the Gaussian process and radial basis function communities, based upon integrated variance experimental design and low-discrepancy sequences, respectively.

3.1 Greedy nested sampling using the power function

Global optimization problem. From Theorem 2.2, we observe that we can reduce the error of the interpolation by minimizing the power function. Given a large set of candidate points X_{cand}, an optimal subset of points $X^\star$ with $|X^\star| = m$ can be found by solving the following optimization problem:

$$X^\star = \operatorname*{argmin}_{X \subset X_{cand}} \big\|P_{V(X)}\big\|_{L^\infty(\Gamma)} \quad \text{subject to} \quad |X| \leq m. \tag{14}$$

Unfortunately, solving this global optimization problem is often intractable. Consequently a number of greedy strategies have been introduced [7, 17, 35].

Greedy-optimal nested sample selection. The greedy version of the optimization problem in (14) we consider uses two properties of the power function $P_{V(X)}$. The first property is given by the following lemma.

Lemma 3.1 (Monotonicity [17]). *With Γ satisfying an interior cone condition, $\mathcal{K}$ being positive definite and $X', X'' \subseteq \Gamma$ being finite sets of samples with $X' \subseteq X''$, it holds*

$$P_{V(X')}(\boldsymbol{y}) \geq P_{V(X'')}(\boldsymbol{y}) \quad \textit{for all} \quad \boldsymbol{y} \in \Gamma.$$

The proof of this lemma can be found in [31]. The lemma states as the number of samples increases, $P_{V(X)}(\boldsymbol{y})$ decreases monotonically for all $\boldsymbol{y} \in \Gamma$. This property guarantees that, when using a greedy optimization strategy, adding a locally optimal sample will always decrease the power-function and approximation error.

The following property is also useful for constructing a greedy sampling scheme.

Lemma 3.2. *With Γ and $\mathcal{K}$ as before and $X = \{\boldsymbol{y}_1, \dots, \boldsymbol{y}_n\} \subset \Gamma$ a finite set of samples, it holds*

$$P_{V(X)}(\boldsymbol{y}_j) = 0 \quad \textit{for all} \quad \boldsymbol{y}_j \in X.$$

This lemma states that the power function $P_{V(X)}$ is zero for all elements of X. This statement is obvious since in the definition of the power function the numerator becomes zero in all samples $\boldsymbol{y} \in X$, since $\mathcal{I}_{V(X)}$ is an *interpolant*.

The two aforementioned properties can be used to design the greedy sampling algorithm, summarized in Algorithm 1. Let $X_{cand} \subset \Gamma$ be a large, but finite candidate set. Given $j-1$ samples X_{j-1} already selected from the X_{cand}, we select the j-th sample $\boldsymbol{y}_j$ from the remaining samples $X_{cand} \setminus X_{j-1}$ by *maximizing* $\|P_{V(X_{j-1})}\|_{L^\infty(\Gamma)}$. If X_{cand} is large enough, then $\boldsymbol{y}_j$ will approximate well $\boldsymbol{y}_{max} = \operatorname{argmax}_{\boldsymbol{y}\in\Gamma} P_{V(X_{j-1})}(\boldsymbol{y})$ in the sense that $P_{\mathcal{K},X_{j-1}\cup\{\boldsymbol{y}_j\}}(\boldsymbol{y}_{max}) \approx 0$. This process is repeated until the desired number of samples is reached.

Algorithm 1 Greedy sample selection by the power function [17].

function GETGREEDYSAMPLES($\mathcal{K}$, X_{cand}, m)
  $\boldsymbol{y}_1 = \operatorname{argmax}_{\boldsymbol{y}\in X_{cand}} P_{V(\{\boldsymbol{y}\})}(\boldsymbol{y})$
  $X_1 = \{\boldsymbol{y}_1\}$
  for $j = 2, 3, \ldots, m$ **do**
    $\boldsymbol{y}_j = \operatorname{argmax}_{\boldsymbol{y}\in X_{cand}\setminus X_{j-1}} |P_{V(X_{j-1})}(\boldsymbol{y})|$
    $X_j = X_{j-1} \cup \{\boldsymbol{y}_j\}$
  return X_m

Efficient and stable implementation. As mentioned before, the evaluation of the power function using equation (12) is not stable [30, 35]. To improve stability, [20] proposes to iteratively build a Newton basis $\{\mathfrak{N}_1, \ldots, \mathfrak{N}_j\}$ for $V(X_j)$ during the greedy algorithm (Algorithm 1). By using this Newton basis, the power function can be evaluated stably using the expression

$$P^2_{V(X_j)}(\boldsymbol{y}) = \mathcal{K}(\boldsymbol{y},\boldsymbol{y}) - \sum_{i=1}^{j} \mathfrak{N}_i^2(\boldsymbol{y}).$$

Moreover, it is discussed in [27] that greedy-optimal sampling using the Newton basis of a positive definite kernel $\mathcal{K}$ can be performed by the *pivoted Cholesky factorization* [13].

Given the candidate set matrix $\boldsymbol{A}_{X_{cand}}$, the first j steps of the pivoted Cholesky factorization given in Algorithm 2 compute the column matrix $L_j \in \mathbb{R}^{N_{cand}\times j}$ and a permutation vector $\boldsymbol{p}$. Each column of L_j defines a Newton basis vector of $V(X_j)$ evaluated on the candidate set X_{cand}. The first j pivots in $\boldsymbol{p}$ denote the indices of the points from the candidate set that maximize the power-function, see [27].

Note that, in practice, the entire matrix $\boldsymbol{A}_{X_{cand}}$ needs not be computed. Instead, the entries $\mathcal{K}$ should only be evaluated when required. Also note, that this Cholesky procedure produces a nested set of samples and can be used to enrich an existing set of points. Given m samples we can add an additional sample from the candidate set X_{cand} by performing another factorization step on the matrix A and adjusting the current Cholesky factor using the steps associated with the inner loop of Algorithm 2.

Algorithm 2 Pivoted Cholesky factorization [13].

function PIVOTEDCHOLESKY($\boldsymbol{A}$) ▷ $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ s.p.d.
  $r = 1$ ▷ current row
  $\boldsymbol{d} = \mathrm{diag}(\boldsymbol{A})$ ▷ diagonal of $\boldsymbol{A}$
  $\boldsymbol{p} = (1, \dots, n)$ ▷ initialization of permutation
  while $r \leq n$ **do**
    $i_{max} = \mathrm{argmax}_{j\in\{r,r+1,\dots,n\}}\, d_{p_j}$ ▷ find pivot
    swap p_r and $p_{i_{max}}$ ▷ exchange columns
    $\ell_{r,p_r} = \sqrt{d_{p_r}}$ ▷ compute diagonal entry
    for $i \in \{r+1, \dots, n\}$ **do** ▷ comp. other entries
      $\ell_{p_{i_{max}},r} = \big(a_{p_r,p_{i_{max}}} - \sum_{j=1}^{r-1} \ell_{p_r,j}\ell_{p_{i_{max}},j}\big)/\ell_{p_r,r}$
      $d_{p_{i_{max}}} = d_{p_{i_{max}}} - \ell^2_{p_{i_{max}},r}$
    $r = r + 1$ ▷ got to next row
  return $\big(\boldsymbol{L} = (\ell_{j,j'})_{j,j'}, \boldsymbol{p}\big)$

3.2 Sampling using a weighted power functions

In accordance with Theorem 2.1, the sampling strategy from the last subsection is tailored to approximating the unweighted interpolant, i.e., it minimizes $|f(\boldsymbol{y}) - \mathcal{I}_{V(X)}(\boldsymbol{y})|$. With the goal of generating approximations that are accurate with respect to the weighted L^p_ω norm (1), we aim to generate approximations that minimize

$$\big|\big(f(\boldsymbol{y}) - (\mathcal{I}_{V(X)}f)(\boldsymbol{y})\big)g(\boldsymbol{y})\big|$$

for the weight function $g(\boldsymbol{y}) = \omega(\boldsymbol{y})^{\frac{1}{p}}$. This choice is motivated by noting that

$$\epsilon_\omega(u, V, p) := \left(\int_\Gamma |u(\boldsymbol{y}) - (I_V u)(\boldsymbol{y})|^p \omega(\boldsymbol{y}) \mathrm{d}\boldsymbol{y}\right)^{1/p} = \left(\int_\Gamma |u(\boldsymbol{y}) - (I_V u)(\boldsymbol{y}) g(\boldsymbol{y})|^p \mathrm{d}\boldsymbol{y}\right)^{1/p}.$$

With the goal of generating samples tailored to fg, instead of f, we define the *weighted kernel*

$$\tilde{\mathcal{K}}(\boldsymbol{y}, \boldsymbol{y}') := g(\boldsymbol{y})\mathcal{K}(\boldsymbol{y}, \boldsymbol{y}')g(\boldsymbol{y}').$$

Then, given a function of the form

$$f(\boldsymbol{y}) = \sum_{j=1}^{N} \alpha_j \mathcal{K}(\boldsymbol{y}, \boldsymbol{y}_j),$$

we have

$$f(\boldsymbol{y})g(\boldsymbol{y}) = \sum_{j=1}^{N} \tilde{\alpha}_j \tilde{\mathcal{K}}(\boldsymbol{y}, \boldsymbol{y}_j), \quad \text{where} \quad \tilde{\alpha}_j = \frac{\alpha_j}{g(\boldsymbol{y}_j)}$$

and

$$\|fg\|_{\mathcal{N}_\mathcal{K}(\Gamma)} = \|f\|_{\mathcal{N}_{\tilde{\mathcal{K}}}(\Gamma)},$$

where $\mathcal{N}_{\tilde{\mathcal{K}}}$ is the native space of the weighted kernel. If we further define the *weighted power function*

$$P_{g,V(X)}(\boldsymbol{y}) = \sup_{f\in\mathcal{N}_\mathcal{K}(\Gamma),\, f\neq 0} \frac{\big|\big(f(\boldsymbol{y}) - (\mathcal{I}_{V(X)}f)(\boldsymbol{y})\big)g(\boldsymbol{y})\big|}{\|fg\|_{\mathcal{N}_\mathcal{K}(\Gamma)}}, \tag{15}$$

we immediately arrive at the error estimate

$$\big|\big(f(\boldsymbol{y}) - (\mathcal{I}_{V(X)}f)(\boldsymbol{y})\big)g(\boldsymbol{y})\big| \leq P_{g,V(X)}(\boldsymbol{y})\|fg\|_{\mathcal{N}_\mathcal{K}(\Gamma)}.$$

In the weighted analogy to Theorem 2.2, the weighted power function (10) is equal to

$$P_{g,V(X)}(\boldsymbol{y}) = \sqrt{g(\boldsymbol{y})\mathcal{K}(\boldsymbol{y}, \boldsymbol{y})g(\boldsymbol{y}) - \tilde{\boldsymbol{A}}^\top_{X,\{\boldsymbol{y}\}}\tilde{\boldsymbol{A}}^{-1}_{X,X}\tilde{\boldsymbol{A}}_{X,\{\boldsymbol{y}\}}},$$

where

$$\tilde{\boldsymbol{A}}_{X,X'} := \begin{pmatrix} g(\boldsymbol{y}_1)\mathcal{K}(\boldsymbol{y}_1,\boldsymbol{y}'_1)g(\boldsymbol{y}'_1) & \dots & g(\boldsymbol{y}_1)\mathcal{K}(\boldsymbol{y}_1,\boldsymbol{y}'_{N'})g(\boldsymbol{y}'_{N'}) \\ \vdots & \ddots & \vdots \\ g(\boldsymbol{y}_N)\mathcal{K}(\boldsymbol{y}_N,\boldsymbol{y}'_1)g(\boldsymbol{y}'_1) & \dots & g(\boldsymbol{y}_N)\mathcal{K}(\boldsymbol{y}_N,\boldsymbol{y}'_{N'})g(\boldsymbol{y}'_{N'}) \end{pmatrix}$$

for two sets X, X' with elements $\boldsymbol{y}_j$ and $\boldsymbol{y}'_{j'}$ and sizes $|X| = N$, $|X'| = N'$. Hence, the pivoted Cholesky factorization should be appropriately modified in order to find an optimal set of points X which minimizes the $L^\infty(\Gamma)$-norm of the weighted power function.

In order to adapt Algorithm 2 for the weighted power function, we shall make first the following considerations. Let $\boldsymbol{A} = (a_{j,j'})_{j,j'} \in \mathbb{R}^{n\times n}$ be a symmetric and positive (semi-) definite matrix and $\boldsymbol{D} = \operatorname{diag}(d_1,\dots,d_n) \in \mathbb{R}^{n\times n}$ be a diagonal matrix with positive diagonal entries $d_j > 0$. Then, the pivoted Cholesky factorization for the matrix

$$\tilde{\boldsymbol{A}} = \left(\tilde{a}_{j,j'}\right)_{j,j'} = \left(d_j a_{j,j'} d_{j'}\right)_{j,j'} = \boldsymbol{DAD}$$

yields a rank-m approximation $\tilde{\boldsymbol{L}}\tilde{\boldsymbol{L}}^\top$ with $\tilde{\boldsymbol{L}} = (\tilde{\ell}_{j,j'})_{j,j'} \in \mathbb{R}^{n\times m}$:

$$\tilde{\boldsymbol{A}} = \boldsymbol{DAD} \approx \tilde{\boldsymbol{L}}\tilde{\boldsymbol{L}}^\top.$$

The matrix $\boldsymbol{L} = (\ell_{j,j'})_{j,j'} := \boldsymbol{D}^{-1}\tilde{\boldsymbol{L}}$ obviously results in a rank-m factorization $\boldsymbol{A} \approx \boldsymbol{L}\boldsymbol{L}^\top$, where it is especially inferred that

$$\tilde{\ell}_{j,j'} = d_j \ell_{j,j'} \quad \text{for all} \quad j = 1,\dots,n,\ j' = 1,\dots,m.$$

The only difference between applying the pivoted Cholesky factorization for $\boldsymbol{A}$ and for $\tilde{\boldsymbol{A}}$ is thus the different choice of the pivots. Therefore, by setting

$$\boldsymbol{A} = \left(\mathcal{K}(\boldsymbol{y}_j,\boldsymbol{y}_{j'})\right)_{j,j'}, \quad \boldsymbol{D} = \operatorname{diag}\left(g(\boldsymbol{y}_1),\dots,g(\boldsymbol{y}_{N_{cand}})\right),$$

we readily verify that the pivoted Cholesky factorization with respect to

$$\tilde{\boldsymbol{A}} = \left(g(\boldsymbol{y}_j)\mathcal{K}(\boldsymbol{y}_j,\boldsymbol{y}_{j'})g(\boldsymbol{y}_{j'})\right)_{j,j'}$$

can be realized by Algorithm 3, where we directly compute the matrix $\boldsymbol{L}$ as output instead of the matrix $\tilde{\boldsymbol{L}}$. Note that this procedure can be reinterpreted as a type of importance sampling.

Algorithm 3 Weighted pivoted Cholesky factorization for kernels.

function WEIGHTEDPIVOTEDCHOLESKY(X_{cand}, $\mathcal{K}$, g, m)
  $r = 1$ ▷ current row
  $N_{cand} = |X_{cand}|$
  $\boldsymbol{d} = \operatorname{diag}\left(\mathcal{K}(\boldsymbol{y}_1,\boldsymbol{y}_1),\dots,\mathcal{K}(\boldsymbol{y}_{N_{cand}},\boldsymbol{y}_{N_{cand}})\right)$
  $\boldsymbol{p} = (1,\dots,N_{cand})$ ▷ initialization of permutation
  while $r \le m$ **do**
    $i_{max} = \operatorname{argmax}_{j\in\{r,r+1,\dots,N_{cand}\}}\left\{d_{p_j} g^2(\boldsymbol{y}_{p_j})\right\}$ ▷ find weighted pivot
    swap p_r and $p_{i_{max}}$ ▷ exchange columns
    $\ell_{p_r,r} = \sqrt{d_{p_r}}$ ▷ compute diagonal entry
    for $i \in \{r+1,\dots,N_{cand}\}$ **do** ▷ comp. other entries
      $\ell_{p_{i_{max}},r} = \left(\mathcal{K}(\boldsymbol{y}_{p_r},\boldsymbol{y}_{p_{i_{max}}}) - \sum_{j=1}^{r-1}\ell_{p_r,j}\ell_{p_{i_{max}},j}\right)/\ell_{p_r,r}$
      $d_{p_{i_{max}}} = d_{p_{i_{max}}} - \ell^2_{p_{i_{max}},r}$
    $r = r+1$ ▷ got to next row
  return $\left(\boldsymbol{L} = (\ell_{j,j'})_{j,j'},\ \boldsymbol{p}\right)$

3.3 Integrated variance experimental design

In the field of Gaussian process regression, a methodology similar to the greedy optimization strategy above has been developed in [11]. In this approach, optimal experimental designs, i.e. optimal samplings, are created by minimizing the a-posterior integrated variance (IVAR).

The a posteriori variance of a Gaussian process, cf. Subsection 2.3, conditioned to given input samples X is

$$c(\boldsymbol{y}\,|\,X) = \mathcal{K}(\boldsymbol{y},\boldsymbol{y}) - A_{X,\{\boldsymbol{y}\}}^{\top}\big(A_X + \sigma^2\boldsymbol{I}\big)^{-1}A_{X,\{\boldsymbol{y}\}}.$$

In general, the IVAR approach aims at finding an optimal set of samples $X^\star \subset \mathcal{U} \subset \Gamma$ of size $|X^\star| = m$ from a set of feasible *experiments* $\mathcal{U}$ by solving the minimization problem

$$X^\star = \operatorname*{argmin}_{X\subset\mathcal{U},\,|X|=m} \int_\Gamma c(\boldsymbol{y}|X)\omega(\boldsymbol{y})\mathrm{d}\boldsymbol{y}. \tag{16}$$

Note that we adapted the notation from [11] appropriately.

In accordance with (12), the posterior variance $c(\boldsymbol{y}\,|\,X)$ in Gaussian process regression for a noise variance of $\sigma^2 = 0$ is just the square of the power function, i.e. we have

$$c(\boldsymbol{y}|X) \equiv P_{V(X)}^2.$$

Hence, the minimization of (16) is in fact equivalent to the minimization

$$X^\star = \operatorname*{argmin}_{X\subset\mathcal{U},\,|X|=m} \int_\Gamma P_{V(X)}^2(\boldsymbol{y})\omega(\boldsymbol{y})\mathrm{d}\boldsymbol{y}.$$

This means, instead of minimizing the $L^\infty(\Gamma)$-norm of the power function as proposed in the last section, IVAR minimizes the (squared) weighted $L^2_\omega(\Gamma)$-norm of the power function.

3.4 Low discrepancy sequences

It is common in both, the Gaussian process regression and radial basis function literature, to use low-discrepancy sequences as point sets for approximation [5, 21, 38]. Low discrepancy sequences are typically used for approximation in unweighted spaces. However, if the PDF ω has product structure, i.e.

$$\omega(\boldsymbol{y}) = \omega_1(y_1)\cdot\ldots\cdot\omega_d(y_d),$$

then we can apply *inverse transform sampling* [6, 16] to generate points sets $X^\star$ that follow the underlying distribution of ω. To this end, let

$$F_{\omega_j}(y_j) = \int_{t\le y_j} \omega_j(t)\mathrm{d}t$$

denote the cumulative distribution function of the j-th variable y_j. Given a set of samples drawn uniformly on [0,1], e.g. from a unweighted low-discrepancy sequence, we can obtain a new set of samples

$$X^\star := \left\{\big(F_{\rho_1}^{-1}(y_1),\ldots,F_{\rho_d}^{-1}(y_d)\big) : \boldsymbol{y}\in X\right\},$$

which can be used for weighted approximation. In this article, we will Halton sequences for building RBF approximations.

Note that inverse transform sampling can be applied to non-product densities. However, doing so requires the use of non-linear transformations, such as the Rosenblatt transformation which can be infeasible even in moderate dimensions and introduces strong non-linearities which decrease the accuracy of approximations for fixed sample sizes. Consequently, unlike our weighted greedy strategy and weighted IVAR, low-discrepancy sequences cannot be used for approximation for PDFs of dependent variables. Nevertheless, we include the use of inverse transform sampling in a numerical test case with tensor-product density in order to provide a truly competitive method comparison.

4 Numerical examples

In this section, we demonstrate the efficacy of our proposed approach on a number of numerical examples and compare its performance with a number of existing sampling approaches which were presented in Section 3.

In all the following examples, we will measure the performance of an approximation using the relative weighed $L^p_\omega(\Gamma)$-norm, i.e.

$$\epsilon_{\omega,p}\big(u, V(X)\big) := \frac{\left(\int_\Gamma \left|u(\boldsymbol{y}) - (\Pi_{V(X)}u)(\boldsymbol{y})\right|^p \omega(\boldsymbol{y})\mathrm{d}\boldsymbol{y}\right)^{1/p}}{\left(\int_\Gamma |u(\boldsymbol{y})|^p \omega(\boldsymbol{y})\mathrm{d}\boldsymbol{y}\right)^{1/p}}. \tag{17}$$

Unless otherwise stated, we set $p = 2$. In Section 4.4, we investigate the performance our weighted Cholesky sampling scheme for several different $p \geq 1$.

For the examples considered here, we cannot compute the error (17) exactly. Consequently, we approximate it by the Monte Carlo estimator

$$\tilde{\epsilon}_{\omega,p}\big(u, V(X)\big) := \frac{\left(\sum_{i=1}^N \left|u(\boldsymbol{y}_i) - (\Pi_{V(X)}u)(\boldsymbol{y}_i)\right|^p\right)^{1/p}}{\left(\sum_{i=1}^N |u(\boldsymbol{y}_i)|^p\right)^{1/p}}.$$

The samples $\boldsymbol{y}_i$ in the Monte Carlo estimator are drawn i.i.d. from the underlying distribution of the PDF ω.

In the following examples we set $N = 10,000$ and use the Gaussian/squared exponential kernel from equations (5) and (6). We explore the the generation of samples in two situations; when the hyper-parameters of the kernel are fixed and when they are optimized with the procedure described in Subsection 2.3. Unless otherwise stated, we create optimal samplings $X^\star$ using the weighted pivoted Cholesky factorization when X_{cand} consists of the first 5000 samples of the Halton sequence and 5000 samples drawn from $\omega(\boldsymbol{y})$.

4.1 Impact of using the domination method

In Section 1 we argued, based upon Lemma 1.1, that incorporating knowledge of the density ω into the sampling process improves the convergence of the L^p_ω error of an approximation by a (possibly large) constant factor. In this section, we provide numerical evidence to support this claim. With this goal, consider the function

$$u(\boldsymbol{y}) := \cos\left(2\pi + 40\frac{1}{d^2}\sum_{j=1}^d y_j\right), \tag{18}$$

defined for for arbitrary dimension $d > 0$, and let PDF $\omega(\boldsymbol{y})$ of $\boldsymbol{y}$ be the product

$$\omega^{\text{Beta}}(\boldsymbol{y}) := \prod_{j=1}^d \omega_j^{\text{Beta}}(y_j)$$

of the PDFs of univariate Beta random variables

$$\omega_j^{\text{Beta}}(y_j) := \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} y_j^{\alpha-1}(1-y_j)^{\beta-1}$$

with parameters α and β, where Γ is the gamma function.

In Figure 1, we compare the convergence in error for different dominating measures g when $d = 2$ and the parameters of the product-Beta density ω are $\alpha = \beta = 20$. We define the dominating measures to also be product-Beta densities with parameters α_g and β_g and compare errors for different values of these parameters $\alpha_g = \beta_g = \tau$. For each choice of dominating measure we compute the constant C_r from Lemma 1.1 as a measure of the distance between the densities g and ω. For this example $\delta = 0$ (see Lemma 1.1). As the

dominating measure approaches ω, i.e. $\alpha_g \to \alpha$ and $\beta_g \to \beta$ which corresponds to $C_r \to 1$, the dominating measure approximation becomes more efficient. The plot clearly shows that constructing an approximation from sample designs targeting a density g, which is not ω, results in a degradation of accuracy, and the penalty grows as the quantity C_r grows. Although using a dominating measure only effects the constant of convergence and not the rate of convergence (see Lemma 1.1), the change in the constant is significant.

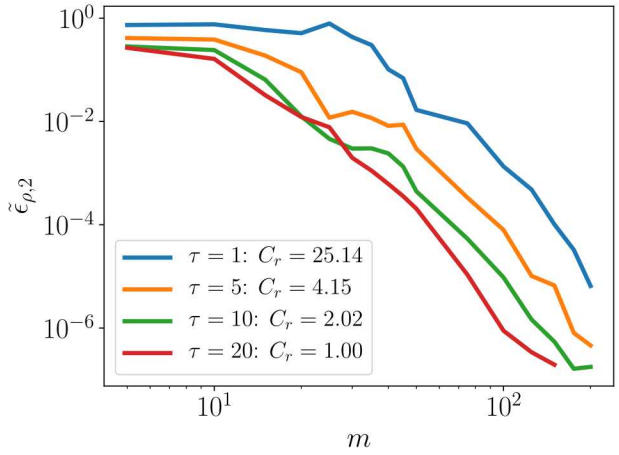

Figure 1: A comparison of the error in approximations of function (18) obtained by the weighted Cholesky sampling using different domination measures. Here, ω is a tensor product of identical univariate Beta PDFs with parameters $\alpha = \beta = 20$ and we vary the distribution parameters of the domination measure $\alpha_g = \beta_g = \tau$. As predicted by Theorem 1.1 the error in the approximation increases as the distance between the measure increases.

4.2 Comparison to existing weighted sampling strategies

With the next numerical study, we compare the performance of the weighted Cholesky sampling strategy to other existing approaches in the field, namely the unweighted Cholesky sampling strategy, the IVAR approach, and the inverse transform sampling approach. Note that the latter is only feasible for product densities. Therefore, we again consider (18) with the same Beta density.

Figure 2 plots the error in the RBF approximation for various sampling strategies as the number of samples is increased. Weighted Cholesky outperforms all methods when $d = 2$ and has similar performance to Weighted Halton when $d = 10$. We do not plot IVAR when $d = 10$ because for large sample sizes the optimization was taking over a day. This could likely improved with some code optimization, however any conclusions we would draw would remain the same. The IVAR optimization problem is computationally much more expensive than our approach, it involves a non-convex optimization of md unknowns, yet the error obtained using IVAR will be similar to our approach or worse.

4.3 Improvements by weighting for non-product PDFs

In our second test case, we again use the cosine function from (18). However, we use a non-product density, which we construct by using the *Nataf transform* [23]. The PDF ω^{Nat} is given by

$$\omega^{Nat}(\boldsymbol{y}) = \frac{\eta_{R^V}(\boldsymbol{z})}{\prod_{j=1}^{d} \eta(z_i)} \prod_{j=1}^{d} \bar{\omega}_j(y_j), \tag{19}$$

with

$$\eta_{R^V}(\boldsymbol{z}) = \frac{1}{\sqrt{(2\pi)^d \det(\boldsymbol{R^V})}} \exp\left(-\frac{1}{2}\boldsymbol{z}^\top (\boldsymbol{R^V})^{-1}\boldsymbol{z}\right).$$

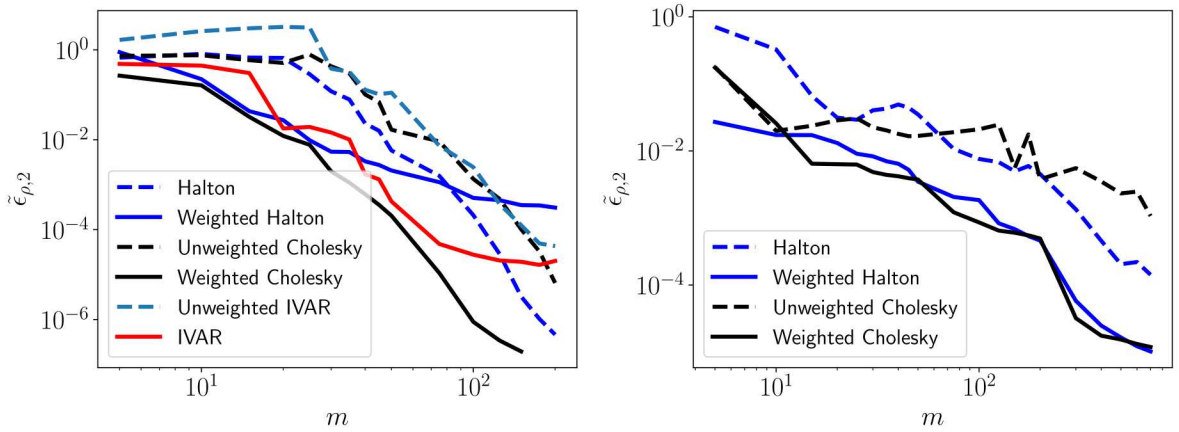

Figure 2: Comparison of the sampling strategies for building RBF approximations of (18) with $d = 2$ *(left)* and $d = 10$ *(right)*.

The correlation matrix $\boldsymbol{R^V}$ is given by

$$\left(\boldsymbol{R^V}\right)_{j,j'} = \begin{cases} 1, & \text{if } j = j', \\ (-1)^{j+j'}\, 0.9, & \text{otherwise.} \end{cases} \tag{20}$$

The $\bar{\omega}_j$ are the marginal probability distributions of ω^{Nat}, η is the univariate standard normal density function, $z_j := \Phi_g^{-1}\big(F_j(y_i)\big)$ with $F_j^{\omega^{Nat}}(y_i)$ are the marginal cumulative distributions of ω^{Nat}, and Φ_g is the cumulative distribution function. Figure 5 depicts the PDF ω^{Nat} for $d = 2$ with univariate Beta marginals each with parameters $(\alpha, \beta) = (2, 5)$.

For this test case, with non-product PDF, the inverse transform sampling approach is not applicable. Consequently, in Figure 3, we compare the convergence properties for the Halton sequence with the help of the unweighted pivoted Cholesky factorization and the weighted pivoted Cholesky factorization, as shown in Figure 3 for $d = 2$ on the left-hand side and for $d = 10$ on the right-hand side. For both dimensionalities, the unweighted Cholesky sampling strategy gives only slightly better results than the Halton sequence. In contrast, the weighted Cholesky sampling strategy clearly outperforms the other methods. These results become even more pronounced in higher dimensions.

Figure 3 also plots the error in the approximation constructed using weighted Cholesky samples when the correlation length of the kernel is optimized each time a batch of samples is requested (Weighted Cholesky opt.). When $d = 2$ we increase the number of samples in X in batches of size 5 for $m \leq 50$ and 25 otherwise. After each of these batches, we re-optimized the hyper-parameters. In $d = 10$ we increase the number of samples in X in batches of size 5 for $m \leq 50$ and $50 \leq m \leq 200$ and 100 otherwise. Regardless of the dimension, optimizing for the best correlation length degrades the performance for small sample sizes, but the effect is minor and decreases as more samples are added.

4.4 Different weighted p-norms

In this experiment, we vary q in $\epsilon_{\omega,q}\big(u, V(X)\big)$ in accordance with $q = 1, 2, 4, 6$ using (18) and the dependent beta density with $d = 2$ and $d = 10$. The results are found in Figure 4, where the set-up of the experiment is the same as in the previous one. We observe again that the weighted Cholesky sampling strategy clearly outperforms the unweighted strategies independent on the particular choice of q.

Figure 5 depicts the samples sets ($m = 100$) generated, for the non-product PDF, by the weighted and the unweighted Cholesky sampling strategies with the non-product PDF with $d = 2$. The weighted approach clearly allocates more samples to regions of high-probability. This is the major reason for its better performance. However, as p is increased, more samples are allocated to regions of lower probability.

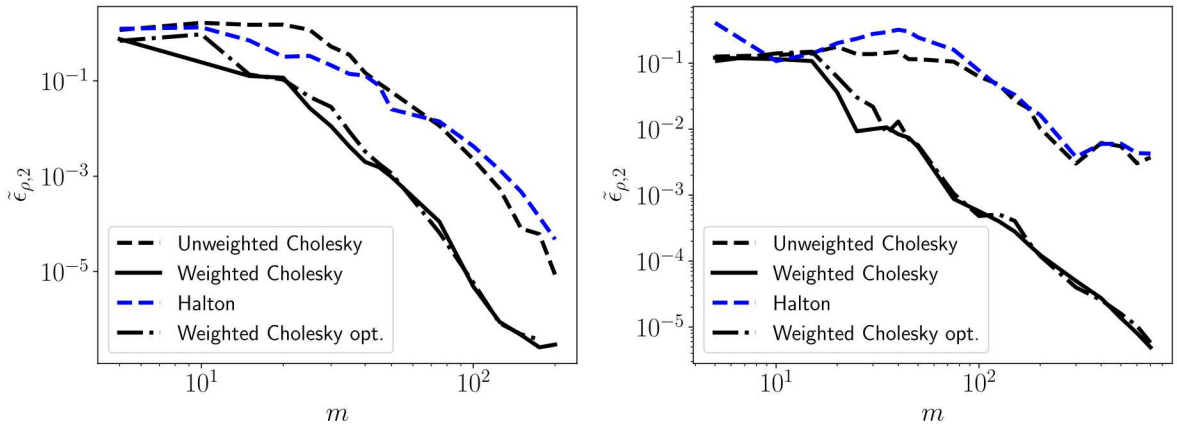

Figure 3: Comparison of the weighted and the unweighted Cholesky sampling strategies with Halton sequence applied to the uniform dominating measure for (18) with a non-product PDF. The weighted Cholesky sampling strategy clearly outperforms the other results for $d = 2$ *(left)* and $d = 10$ *(right)*.

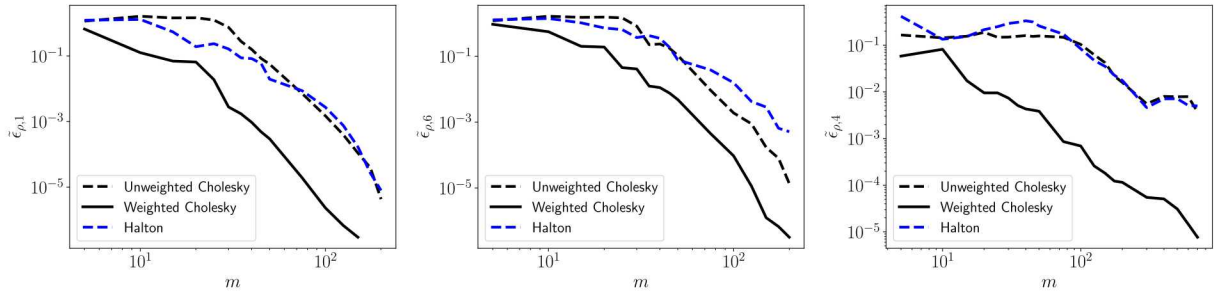

Figure 4: A comparison of the error in approximations of function (18) obtained using weighted Cholesky sampling using different values of q targeting different norms. Here we use the non-product PDF (19) with correlation (20), $d = 2$ (left and middle plots), $d = 10$ (right plot) and univariate Beta marginals each with parameters $(\alpha, \beta) = (2, 5)$. The values of q from left to right are 1,6,4, respectively.

5 Bayesian inference

In this section, we will use Cholesky sampling to construct posterior surrogates for efficient Bayesian inference of model parameters from observational data, cf. [32]. To make this precise, let $f(\boldsymbol{y}) : \mathbb{R}^d \to \mathbb{R}^n$ be a vector of n observable quantities, parameterized by d random variables $\boldsymbol{y}$. Bayes' rule can be used to define the posterior density for the model parameters $\boldsymbol{y}$ given observational data $\boldsymbol{d}$:

$$\rho(\boldsymbol{y}) = \pi(\boldsymbol{y}|\boldsymbol{d}) = \frac{\pi(\boldsymbol{d}|\boldsymbol{y})\pi(\boldsymbol{y})}{\int_{\mathcal{D}} \pi(\boldsymbol{d}|\boldsymbol{y})\pi(\boldsymbol{y})\mathrm{d}\boldsymbol{y}},$$

where any prior knowledge on the model parameters is incorporated into the prior density $\pi(\boldsymbol{y})$ and $\pi(\boldsymbol{d}|\boldsymbol{y})$ is the likelihood function which specifies the probability of observing data $\boldsymbol{d}$ given a realization of the model parameters.

5.1 Determining the likelihood

The form of the likelihood is determined by the statistical model that relates the data to the simulation model. In the following, we will assume the following relationship $\boldsymbol{d} = f(\boldsymbol{y}) + \boldsymbol{\epsilon}$, where the noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{\Sigma})$ is normally distributed with mean zero and covariance $\boldsymbol{\Sigma}$. Under this assumption, the likelihood takes the

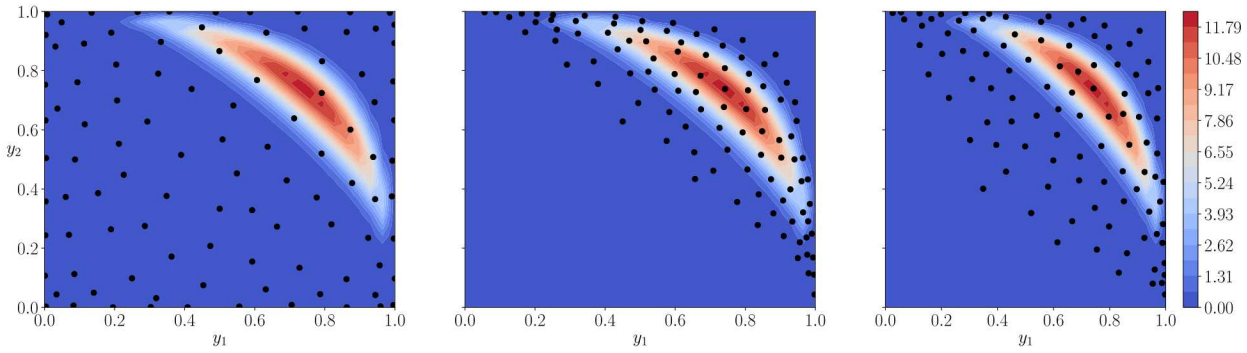

Figure 5: Comparison of samples sets, for the non-product PDF, generated by *(left)* the unweighted Cholesky sampling strategy and *(middle)* the weighted strategy with $q = 2$ and *(right)* $q = 6$. The samples are plotted on top of the contours of the non-product density (19) with correlation (20), $d = 2$, and univariate Beta marginals each with parameters $(\alpha, \beta) = (2, 5)$.

form

$$\pi(\boldsymbol{d}|\boldsymbol{y}) = \exp\left(-\frac{1}{2}\big(\boldsymbol{d} - f(\boldsymbol{y})\big)^\top \boldsymbol{\Sigma}^{-1}\big(\boldsymbol{d} - f(\boldsymbol{y})\big)\right) = \exp\big(-u(\boldsymbol{y})\big),$$

where u is often referred to as the negative log likelihood. Often, when evaluating f, and thus u, is computationally expensive, a surrogate of either f or u is built to avoid the expensive evaluation of the model. For example, [18] proposes to replace the misfit u in the likelihood by an m-point approximation $\Pi_{m,g}u$ that is used to compute an approximation of the posterior distribution, i.e.

$$\rho_{m,g} = \pi_{m,g}(\boldsymbol{y}|\boldsymbol{d}) = \frac{\exp\big(-\Pi_{m,g}u(\boldsymbol{y})\big)\pi(\boldsymbol{y})}{\int_{\mathcal{D}} \exp\big(-\Pi_{m,g}u(\boldsymbol{y})\big)\pi(\boldsymbol{y})\mathrm{d}\boldsymbol{y}}. \tag{21}$$

The authors of [18] construct a polynomial chaos expansion (PCE) which is tailored to the prior distribution, i.e. $g(\boldsymbol{y}) = \rho(\boldsymbol{y})^{\frac{1}{2}}$. Using the prior as a dominating measure is inefficient, as it has often a much larger non-zero support than the posterior. Attempts have been made to increase the efficiency of Bayesian inference by using low-order localized surrogates to facilitate sampling in regions of high probability [4, 19]. The use of localized surrogates results in small-rates of convergence. Recently, PCE tailored to the posterior distribution has been used to obtain higher rates of convergence [15, 34].

5.2 Using weighted Cholesky sampling

In the following, we demonstrate that our weighted Cholesky sampling scheme can be used to efficiently construct RBF/Gaussian process surrogates of the misfit u. The adaptive algorithm we use is summarized in Algorithm 4 and described here.

Algorithm 4 Adaptive Cholesky sampling for Bayesian inference.

function GETPOSTERIORAPPROX($\mathcal{K}$, X_{cand}, $\{\Delta m_j\}_{j=1}^s$)
  $X_0 = \emptyset$, $u_0 = \emptyset$, $\beta_0 = 0$
  for $j = 1, 2, \ldots, s$ **do**
    $g_j = \exp\left(-\Pi_g u\right)^{\beta_{j-1}} \pi$
    ΔX_j =UPDATEWEIGHTEDPIVOTEDCHOLESKY(X_{cand}, $\mathcal{K}$, X_{j-1},g_j, Δm_j)
    $X_j = X_{j-1} \cup \Delta X_j$
    $u_j = u_{j-1} \cup u(\Delta X_j)$ ▷ evaluate model at new samples
    $\Pi_{g,m_j} u$ =APPROXIMATEFUNCTION(X_j, u_j)
    β_j = UPDATETEMPERINGPARAMETER(β_{j-1},$\Pi_{g,m_j} u$)
  return X_s, $\Pi_{g,m_s} u$

Because the true posterior density ρ is unknown, we cannot directly use it to specify g. Instead, we will use a series of intermediate PDFs which converge to the true distribution

$$g_j(\boldsymbol{y}) = \exp\left(-\Pi_{g_{j-1}} u(\boldsymbol{y})\right)^{\beta_{j-1}} \pi(\boldsymbol{y}), \quad j = 1, \dots, s,$$

where $0 = \beta_0 < \beta_1 < \ldots < \beta_s \leq 1$. A similar approach has been used to improve the performance of Markov chain Monte Carlo (MCMC) when sampling multi-modal posteriors [3]. Such methods are referred to as transitional MCMC.

Starting with an initial set of points X_0 which is often the empty set, Algorithm 4 begins by setting the initial dominating measure to be the prior, i.e. $g_0(\boldsymbol{y}) = \rho(\boldsymbol{y})^{\frac{1}{p}}$. This measure is then adapted as information (evaluations) about u is obtained. Specifically, given an approximation $\Pi_{m_{k-1},g_{k-1}} u$ at iteration $k \geq 1$ built with m_{k-1} samples X_{k-1}, we compute an estimate of the posterior density $\rho_{m_{k-1},g_{k-1}}(\boldsymbol{y})$ and set $g_k(\boldsymbol{y}) = \rho_{m_{k-1},g_{k-1}}(\boldsymbol{y})^{\frac{1}{q}}$.[1] This domination measure is then used to enrich the existing set of samples with another Δm_k samples ΔX_k, yielding $X_{k+1} = X_k \cup \Delta X_k$. This process continues until a sufficient accuracy in the posterior is reached or a computational budget is exhausted. Note that unlike the examples in Section 4 when the weighting density is known we cannot use samples from the unknown density in the candidate set X_{cand}.

Our transitional approach has two advantages. Firstly, it can sample multi-model and/or concentrated PDFs. Moreover, it allows us to build in a level of conservativeness to prevent our weighted sampling approach from being misled during the initial stages of the algorithm when the approximate posterior is highly inaccurate. At the first iteration when we have no knowledge of the posterior, we simply revert to sampling from the prior π. However, as our approximation becomes more accurate, so does our approximation of the posterior which allows us to place increasing trust in the surrogate for determining the weighting function used with our pivoted Cholesky procedure. The level of trust is dictated by the value of β. The value chosen should be the largest $\beta_j \in (\beta_{j-1}, 1]$ such that the ratio of the previous posterior, using β_{j-1} and the posterior obtained using the new β are "close". Following [3], we choose β_j such that the coefficient of variation of the densities is equal to a pre-defined threshold τ, i.e.

$$\frac{\mathbb{V}\mathrm{ar}\left[\exp\left(-\Pi_g u(\boldsymbol{y})\right)^{\beta-\beta_{j-1}} \pi(\boldsymbol{y})\right]^{\frac{1}{2}}}{\mathbb{E}\left[\exp\left(-\Pi_g u(\boldsymbol{y})\right)^{\beta-\beta_{j-1}} \pi(\boldsymbol{y})\right]} = \tau.$$

Similar to [3], we found $\tau = 1$ to be a reasonable choice. To compute the expectation and variance we use 1000 samples drawn uniformly from Γ. The cost of this step is negligible as it only requires the evaluation of the surrogate.

5.3 Example: Rosenbrock function

This subsection demonstrates the benefit of using our transitional Cholesky sampling algorithm to construct surrogates for Bayesian inference. In the following, assume that the observational quantities are modeled by the function $f : \mathbb{R}^2 \to \mathbb{R}^2$, where

$$f_1(\boldsymbol{y}) = \sqrt{2}(4y_1 - 2), \qquad f_2(\boldsymbol{y}) = -\sqrt{2}\left((4y_2 - 2) - (4y_1 - 2)^2\right).$$

In addition assume that the prior distribution $\pi(\boldsymbol{y})$ is a uniform distribution on $[0, 1]^2$ and the observational data is $\boldsymbol{d} = (\sqrt{2}, 0)^T$. Using a Gaussian error model with covariance $\Sigma = \mathrm{diag}(1, 0.01)$ the negative log likelihood is an affine transformation of the two-dimensional Rosenbrock function

$$u(\boldsymbol{y}) = (1 - \hat{y}_1)^2 + 100(\hat{y}_2 - \hat{y}_1^2)^2, \qquad \hat{\boldsymbol{y}} = 4\boldsymbol{y} - 2.$$

Figure 6 compares the accuracy of surrogates constructed using three sampling schemes. The "Prior Weighted Cholesky" label in the legend refers to the weighted Cholesky sampling with the prior as the dominating measure, the "Adaptive Weighted Cholesky" refers to our transitional algorithm and "Halton" refers to simply using a untransformed Halton sequence. In all cases we simultaneously select samples and optimize the correlation lengths of the RBF/Gaussian process kernel.

[1] Note that when using the weighted Cholesky procedure we can ignore the constant denominator in the approximate posterior (21) as this does not effect the pivoting procedure.

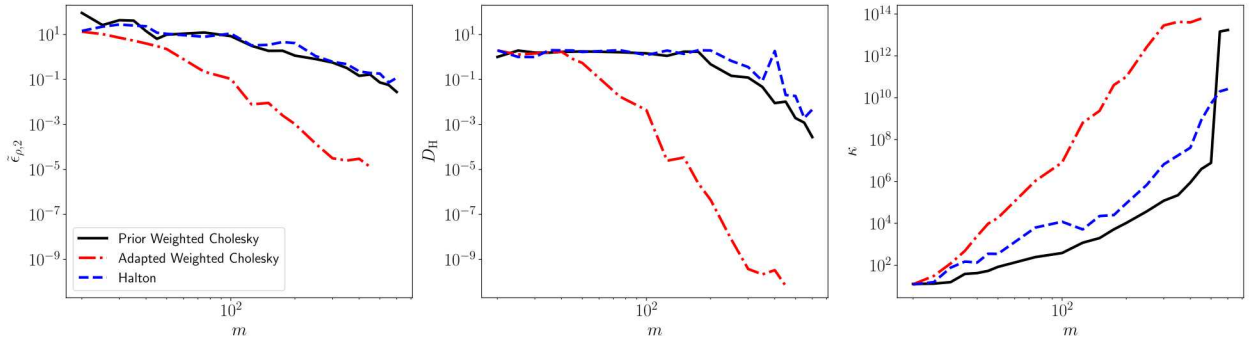

Figure 6: Comparison of sampling strategies used to build surrogates for Bayesian inference. L^2_ρ error in the surrogates as a function of the number of samples *(left)*. Squared Hellinger divergence between the true posterior and approximate posterior obtained using a surrogate *(middle)*. Condition number of the kernel training matrix $\mathcal{K}(X_j, X_j)$ at each iteration j *(right)*.

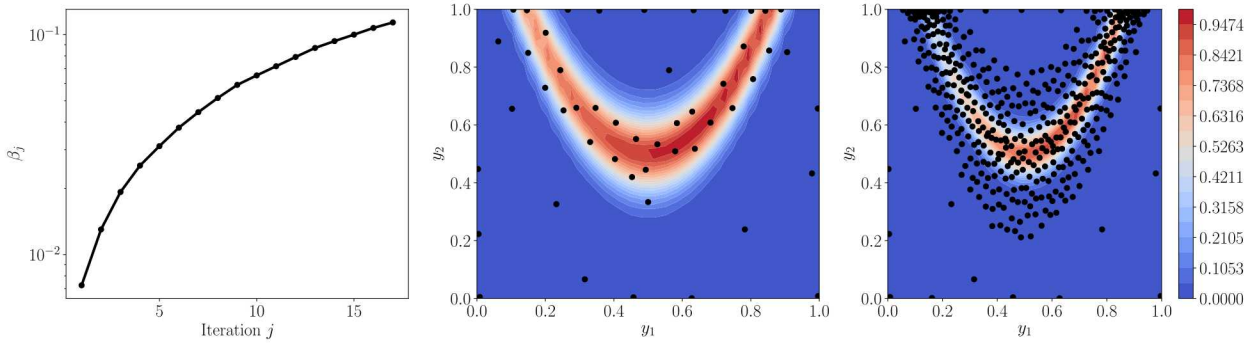

Figure 7: Transitional parameters β_j at each iteration of the adaptive weighted Cholesky algorithm *(left)*. The training set and approximate unnormalized posterior at iteration 6 *(middle)* and at the final iteration *(right)*.

The left plot of Figure 6 depicts the L^2_ρ error of each method, computed using 10,000 samples drawn from the true posterior distribution using rejection sampling. The middle plot depicts the squared Hellinger divergence

$$D^2_{\mathrm{H}}(\rho_{m,g}, \rho) = 1 - \int_\Gamma \rho_{m,g}(\boldsymbol{y})\rho(\boldsymbol{y})\mathrm{d}\boldsymbol{y}$$

between the true and the approximate posteriors. The integral is evaluated using a high-degree tensor-product Gaussian quadrature rule. By both, the L^2_ρ and Hellinger divergence metrics, our adaptive weighted Cholesky algorithm is significantly more efficient than the alternatives. Notice that the convergence curves terminate at different sample sizes for each sampling method. The curves terminate when the condition number of the kernel matrix $\mathcal{K}(X_j, X_j)$ constructed using the training data X_j becomes highly ill-conditioned.

The right plot of Figure 6 compares the condition number of each sampling strategy. For a fixed sample size m, the condition number of our approach is largest. Using the prior as the dominating measure improves the condition number, however the error degrades significantly.

The performance of the adaptive algorithm, is dependent on the transitional parameter β_j. In the left of Figure 7, we plot the evolution of β_j at each iteration of the sampling algorithm. For small sample sizes the approximate posterior changes significantly each time the training set is enriched. This causes β_j to remain small. However, as the accuracy of the surrogate of u improves, β also increase. The middle plot shows the approximate posterior and training samples after 50 model evaluations and the right plot after 450 evaluations. The approximate posterior is 'closer' to the prior for small sample sizes.

6 Conclusion

In this article, we presented a greedy algorithm for generating samples with the goal of minimizing the weighted L^p-error of kernel-based approximations. Most existing literature focuses on strategies for approximations that minimize the unweighted error. The major contribution of our work to the existing literature is the construction of a computationally simple and efficient algorithm based upon pivoted Cholesky factorization for selecting samples for weighted approximation. We demonstrate through extensive numerical examples that our results are almost always significantly more accurate than existing approaches. The improved performance is obtained by concentrating samples in regions of high probability while minimizing the ill-conditioning of the interpolation point set.

In addition to generating accurate interpolants, the algorithm presented has three useful properties. Firstly, the sample sets are nested. Consequently they can easily be enriched with additional samples if additional computational resources become available. Secondly, although we focused on the use of the squared exponential kernel, our algorithm can be used in conjunction with any other radial kernel. And finally the sample sets remain stable even when the length scales of the kernel are changed each time additional samples are added. This is extremely useful because the best length scales of a function are not typically known a priori.

The first part of the article focuses on the weighted approximation of functions when the weight-function is known. The article concludes with an example of how the proposed algorithm can be used for efficiently generating surrogates for the purpose of inferring unknown model parameters from data using Bayesian inference. In this setting the true weight function is the posterior distribution of the model parameters, which is unknown. The algorithm described iteratively builds up an approximation of the posterior, starting from the prior, which is used as a weighting function in the weighted Cholesky sampling procedure. For the example presented, the proposed approach produces approximations of the true posterior which are orders of magnitude more accurate for a pre-specified budget.

7 Acknowledgments

J.D. Jakeman was supported by the Laboratory Directed Research Development (LDRD) program at Sandia National Laboratories. Sandia National Laboratories is a multi-mission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525. The views expressed in the article do not necessarily represent the views of the U.S. Department of Energy or the United States Government.

References

[1] M. Bebendorf. Approximation of boundary element matrices. *Numer. Math.*, 86:565–589 (2000).

[2] X. Chen, E.-J. Park, and D. Xiu. A flexible numerical approach for quantification of epistemic uncertainty. *J. Comput. Phys.*, 240(1):211–224 (2013).

[3] J. Ching and Y.-C. Chen. Transitional Markov chain Monte Carlo method for Bayesian model updating, model class selection, and model averaging. *J. Eng. Mech.*, 133(7):816–832 (2007).

[4] P.R. Conrad, Y.M. Marzouk, N.S. Pillai, and A. Smith. Accelerating asymptotically exact MCMC for computationally intensive models via local approximations. *J. Amer. Statist. Assoc.*, 111(516):1591–1607 (2016).

[5] H. Dette and A. Pepelyshev. Generalized Latin hypercube design for computer experiments. *Technometrics*, 52(4):421–429 (2010).

[6] L. Devroye. *Non-Uniform Random Variate Generation.* Springer, New York, 1986.

[7] G.E. Fasshauer. *Meshfree Approximation Methods with MATLAB.* World Scientific Publishing, River Edge, NJ, 2007.

[8] G.E. Fasshauer. Positive definite kernels: Past, present and future. *Dolomite Res. Notes Approx.*, 4:21–63 (2011).

[9] R.G. Ghanem and P.D. Spanos. *Stochastic finite elements: a spectral approach.* Springer, New York, Inc., 1991.

[10] S.A. Goreinov, N.L. Zamarashkin, and E.E. Tyrtyshnikov. Pseudo-skeleton approximations by matrices of maximal volume. *Math. Notes*, 62(4):515–519 (1997).

[11] A. Gorodetsky and Y. Marzouk. Mercer kernels and integrated variance experimental design. Connections between Gaussian process regression and polynomial approximation. *SIAM/ASA J. Uncertain. Quantif.*, 4(1):796–828 (2016).

[12] A. Gorodetsky and J.D. Jakeman. Gradient-based optimization for regression in the functional tensor-train format. *J. Comput. Phys.* 374: 219–1238 (2018).

[13] H. Harbrecht, M. Peters, and R. Schneider. On the low-rank approximation by the pivoted Cholesky decomposition. *Appl. Numer. Math.*, 62(4):428–440 (2012).

[14] J.D. Jakeman, M. Eldred, and D. Xiu. Numerical approach for quantification of epistemic uncertainty. *J. Comput. Phys.*, 229(12):4648–4663 (2010).

[15] J.D. Jakeman, F. Franzelin, A. Narayan, M. Eldred, and D. Pflüger. Polynomial chaos expansions for dependent random variables. *Comput. Methods Appl. Mech. Engrg.*, 351:643–666 (2019).

[16] M. Kolonko. *Stochastische Simulation. Grundlagen, Algorithmen und Anwendungen.* Vieweg+Teubner, Wiesbaden, 2008.

[17] S.D. Marchi, R. Schaback, and H. Wendland. Near-optimal data-independent point locations for radial basis function interpolation. *Adv. Comp. Math.*, 23:317–330 (2005).

[18] Y. M. Marzouk and D. Xiu. A stochastic collocation approach to Bayesian inference in inverse problems. *Commun. Comput. Phys.* 6(1):826–847 (2009).

[19] S.A. Mattis and B. Wohlmuth. Goal-oriented adaptive surrogate construction for stochastic inversion. *Comput. Methods Appl. Mech. Engrg.*, 339:36–60 (2018).

[20] S. Müller and R. Schaback. A Newton basis for kernel spaces. *J. Approx. Theory*, 161:645–655 (2009).

[21] A.A. Mullur, and A. Messac. Metamodeling using extended radial basis functions: a comparative approach. *Eng. Comput.*, 21:203 (2006).

[22] Akil Narayan and Tao Zhou. Stochastic collocation on unstructured multivariate meshes. *Communications in Computational Physics*, 18(1):1–36, (2015).

[23] A. Nataf. Determination des distributions dont les marges sont donnees. *C. R. Acad. Sci. Paris*, 225:42–43 (1962).

[24] F. Nobile, R. Tempone, and C.G. Webster. A sparse grid stochastic collocation method for partial differential equations with random input data. *SIAM J. Numer. Anal.*, 46(5):2309–2345 (2008).

[25] I.V. Oseledets. Tensor-train decomposition. *SIAM J. Sci. Comput.*, 33(5):2295–2317 (2011).

[26] A. O'Hagan and J.F.C. Kingman. Curve fitting and optimal design for prediction. *J. Roy. Statist. Soc. Ser. B*, 40:1–42 (1978).

[27] M. Pazouki and R. Schaback. Bases for kernel-based spaces. *J. Comput. Appl. Math.*, 236:575–588 (2011).

[28] C.E. Rasmussen and C.K.I. Williams. *Gaussian Processes for Machine Learning.* The MIT Press, Cambridge, MA, 2006.

[29] R. Schaback. Error estimates and condition numbers for radial basis function interpolation. *Adv. Comp. Math.*, 3:251–264 (1995).

[30] R. Schaback and H. Wendland. Kernel techniques: From machine learning to meshless methods. *Acta Numer.*, 15:543–639 (2006).

[31] R. Schaback and J. Werner. Linearly constrained reconstruction of functions by kernels with applications to machine learning *Adv. Comp. Math.*, 25:237 (2006).

[32] A.M. Stuart. Inverse problems: A Bayesian perspective. *Acta Numer.*, 19:451–559 (2010).

[33] A.N. Tikhonov and V.Y. Arsenin. *Solution of Ill-posed Problems.* Winston & Sons, Washington, D.C., 1977.

[34] L.M.M. van den Bos, B. Sanderse, W.A.A.M. Bierbooms, and G.J.W. van Bussel. Bayesian model calibration with interpolating polynomials based on adaptively weighted Leja nodes. *Commun. Comput. Phys.*, 27(1):33–69 (2020).

[35] H. Wendland. *Scattered Data Approximation.* Cambridge University Press, Cambridge, 2004.

[36] C.K.I. Williams. Prediction with Gaussian processes. From linear regression to linear prediction and beyond. In: M.I. Jordan (eds) *Learning in Graphical Models.* NATO ASI Series (Series D: Behavioural and Social Sciences), vol 89. Springer, Dordrecht, 1998.

[37] Z. Wu and R. Schaback. Local error estimates for radial basis function interpolation of scattered data. *IMA J. Numer. Anal.*, 13(1):13–27 (1993).

[38] Z. Wu, D. Wang, P. Okolo, F. Hu, and W. Zhang. Global sensitivity analysis using a Gaussian Radial Basis Function metamodel. *Reliab. Eng. Syst. Safe.*, 154:171–179 (2016).

[39] D. Xiu and J. S. Hesthaven. High-order collocation methods for differential equations with random inputs. *SIAM J. Sci. Comput.*, 27(3):1118–1139 (2005).

[40] D. Xiu and G.E. Karniadakis. The Wiener-Askey polynomial chaos for stochastic differential equations. *SIAM J. Sci. Comput.*, 24(2):619–644 (2002).