Commun. Comput. Phys.
doi: 10.4208/cicp.060712.281212a

Vol. **14**, No. 4, pp. 851-878
October 2013

Adaptive Locally Weighted Projection Regression Method for Uncertainty Quantification

Peng Chen and Nicholas Zabaras*

Materials Process Design and Control Laboratory, 101 Frank H. T. Rhodes Hall, Sibley School of Mechanical and Aerospace Engineering, Cornell University, Ithaca, NY 14853-3801, USA.

Received 6 July 2012; Accepted (in revised version) 28 December 2012

Available online 19 March 2013

Abstract. We develop an efficient, adaptive locally weighted projection regression (ALWPR) framework for uncertainty quantification (UQ) of systems governed by ordinary and partial differential equations. The algorithm adaptively selects the new input points with the largest predictive variance and decides when and where to add new local models. It effectively learns the local features and accurately quantifies the uncertainty in the prediction of the statistics. The developed methodology provides predictions and confidence intervals at any query input and can deal with multi-output cases. Numerical examples are presented to show the accuracy and efficiency of the ALWPR framework including problems with non-smooth local features such as discontinuities in the stochastic space.

AMS subject classifications: 65C30, 65C05, 60-08, 62J05

Key words: Locally weighted projection regression, multi-output, adaptivity, uncertainty quantification.

1 Introduction

Uncertainty Quantification (UQ) is critical in all engineering and scientific fields. UQ is a broad topic involving many aspects, for example, representation of uncertainty, propagation of uncertainty across scales, validation and verification for predictive computational science, visualization of uncertainty in high-dimensional spaces and so on [1–5]. The aim of this paper is to present a methodology for investigating the propagation of uncertainty from the input space to the response space using a deterministic code. The Monte Carlo (MC) is the traditional method for addressing such UQ tasks. Its wide acceptance is due to the fact that it can compute the complete statistics of the solution, while having

*Corresponding author. *Email address:* nzabaras@gmail.com (N. Zabaras)

a convergence rate that is independent of the input dimension. Nevertheless, it quickly becomes inefficient in high dimensional and computationally intensive problems, where only a few samples are available.

Another well-known approach for uncertainty quantification is the spectral finite element method [6]. It involves the projection of the response on a space spanned by orthogonal polynomials of the random variables and the solution of a system of coupled deterministic equations involving the coefficients of the expansion in these polynomials. The scheme was originally developed for Gaussian random variables which correspond to Hermite polynomials (polynomial chaos (PC)). It was later generalized to include other types of random variables (generalized PC (gPC)) [7] and then expanded to the multi-element case. The multi-element generalized polynomial chaos (ME-gPC) method [8, 9] decomposes the stochastic space in disjoint elements and then employs gPC on each element. The coupled nature of the resulting equations that determine the coefficients of the polynomials make the application of the method to high input dimensions rather difficult [10].

Another commonly used UQ method is stochastic collocation. The response is represented as an interpolative polynomial of the system response (output) in the random input space constructed by calls to the computer code at specific input points. In [11, 12], a Galerkin based approximation was introduced alongside a collocation scheme based on a tensor product rule using one-dimensional Gauss quadrature points. These methods do not scale well with the number of random input dimensions. To address high dimensionality problems, various sparse grid collocation (SGC) methodologies were developed based on the Smolyak algorithm [13]. In [14], the authors developed an adaptive hierarchical sparse grid collocation algorithm and considered a number of applications with non-smooth behavior in the stochastic space. However, the piecewise local linear nature of the scheme performed poorly when only a few data points were used while interpolation of adverse functions was shown that it can trick the adaptive algorithm into stopping prior to convergence.

While it is evident that a local approach to uncertainty propagation is required to capture localized features in the stochastic space, it is essential to select within each local model the most informative input to maximize predictive capability. In [15, 16], the authors developed such kind of method, specifically, a treed Gaussian process model where on each leaf of the tree, Bayesian Experimental Design techniques were used to learn a multi-output Gaussian process. The active learning aspects of these Bayesian approaches was shown to lead to better convergence than interpolation-based methods such as adaptive sparse grids [15].

Locally weighted projection regression (LWPR) is an algorithm for incremental nonlinear function approximation in high-dimensional spaces [17–19]. At its core, it employs nonparametric partial least squares regression to locally approximate the relationship between input and output. This methodology has several merits including no need to memorize the training data, adjusting the local models only by the local information, an ability to deal with high dimensional correlated data and providing a confidence interval

for each prediction. However, there still exist several problems that limit its application to uncertainty quantification tasks, for example, (1) the accuracy of this approach cannot be guaranteed, (2) the training data points are randomly sampled and (3) the learning process is not optimized. Hence, in this paper, we propose an adaptive way to improve the learning process of the LWPR method, in order to solve the aforementioned problems with emphasis on uncertainty quantification tasks. For brevity, we name the method as the *adaptive locally weighted projection regression* (ALWPR) method.

The paper is organized as follows. First, the mathematical framework of the stochastic problem is introduced in next followed by a brief review of the LWPR method in Section 2.1. The ALWPR algorithm is described in detail in Section 2.2. Various examples are given in Section 3 demonstrating the accuracy and efficiency of the ALWPR method when applied to UQ tasks. Brief conclusions are provided in Section 4.

2 Methodology

Let us define a complete probability space $(\mathcal{X},\mathcal{F},\mathcal{P})$ with sample space $\mathcal{X}$ which corresponds to the outcomes of some experiments, $\mathcal{F}$ is the $\sigma-$algebra of subsets of $\mathcal{X}$ and $\mathcal{P}:\mathcal{F}\to[0,1]$ is the probability measure. We assume that the stochastic problem has been formulated in such a way that $\mathcal{X}$ is a compact subset of $\mathbf{R}^K$ for some $K\geq 1$:

$$\mathcal{X}=\times_{k=1}^{K}[a_k,b_k], \tag{2.1}$$

with $-\infty\leq a_k<b_k\leq+\infty$ as the upper and lower bounds of each dimension. The underlying $\sigma-$algebra is then:

$$\mathcal{F}=\{B\cap\mathcal{X}:\forall B\in\mathcal{B}^K\}, \tag{2.2}$$

where $\mathcal{B}^K$ is the Borel $\sigma-$algebra of $\mathbf{R}^K$. Then, we let $\mathcal{P}$ be absolutely continuous (with respect to the underlying Lebesgue measure), i.e., there exists a density function $p:\mathcal{X}\to\mathbf{R}$ s.t. for any $A\in\mathcal{F}$ we have

$$P(A)=\int_A p(\mathbf{x})d\mathbf{x}. \tag{2.3}$$

Let us now consider the multi-output function $\mathbf{f}:\mathcal{X}\to\mathbf{R}^M$ representing the result of a deterministic solver modeling a physical system, i.e. at a given input point $\mathbf{x}\in\mathcal{X}$ the predicted response of the system is $\mathbf{f}(\mathbf{x})$. We will write

$$\mathbf{f}=(f_1,\cdots,f_M), \tag{2.4}$$

where f_r is the r-th output of the response function, $r=1,\cdots,M$. In this work, we assume there is no modeling error. The input distribution induces a probability distribution on the output. The UQ problem involves the calculation of the statistics of the output $\mathbf{y}=\mathbf{f}(\mathbf{x})$. Quantities of particular interest are the q-moments $m^q=(m_1^q,\cdots,m_M^q)$ for $q\geq 1$ and $r=1,\cdots,M$:

$$m_r^q:=\int_{\mathcal{X}} f_r^q(\mathbf{x})p(\mathbf{x})d\mathbf{x}. \tag{2.5}$$

In particular, the mean $\mathbf{m}=(m_1,\cdots,m_M)$:

$$\mu_r := m_r^1 = \int_{\mathcal{X}} f_r(\mathbf{x})p(\mathbf{x})d\mathbf{x}, \tag{2.6}$$

and the variance $\mathbf{v}=(v_1,\cdots,v_M)$:

$$v_r := \int_{\mathcal{X}} (f_r(\mathbf{x})-\mu_r)^2 p(\mathbf{x})d\mathbf{x} = m_r^2 - (m_r^1)^2. \tag{2.7}$$

In this work, we build a surrogate model $\tilde{\mathbf{f}}(\cdot)$ to approximate the nonlinear output function $\mathbf{f}(\cdot)$ and the aforementioned UQ problem will be investigated using the surrogate model. As it is typical with other UQ methods (e.g. sparse grids [14]), we concentrate on building a surrogate of individual responses (i.e. for each given r) without considering correlations between the output variables.

2.1 Local weighted projection regression

The core of the LWPR [19] method is to find piecewise low-dimensional linear approximations to the nonlinear output function $\mathbf{f}(\cdot)$ (Eq. (2.4)). For the multi output case, we assume independence of each dimension. Thus for the r-th output, we build a separate LWPR model to approximate f_r. The LWPR method combines Local Weighted Regression (LWR) and Partial Least Squares (PLS) that are briefly discussed next.

2.1.1 Local weighted regression framework

The LWPR regression function for the r-th output is constructed by blending S local linear models (so called receptive fields) $\phi_r^{(s)}(\mathbf{x})$ in the form

$$\tilde{f}_r(\mathbf{x}) = \frac{1}{W(\mathbf{x})}\sum_{s=1}^{S} w_s(\mathbf{x})\phi_r^{(s)}(\mathbf{x}), \quad W(\mathbf{x}) = \sum_{s=1}^{S} w_s(\mathbf{x}). \tag{2.8}$$

Here, $w_s(\mathbf{x})$ is a measure of locality for each data point, which is usually modeled by a Gaussian kernel

$$w_s(\mathbf{x}) = \exp\Big\{-\frac{1}{2}(\mathbf{x}-\mathbf{c}_s)^T\mathbf{D}_s(\mathbf{x}-\mathbf{c}_s)\Big\}, \tag{2.9}$$

where $\mathbf{c}_s$ is the center of the s^{th} local model and $\mathbf{D}_s$ is positive semi-definite distance metric that determines the size and shape of the local model. The dependence of the weights on s is not shown here to simplify the notation.

The local linear model $\phi_r^{(s)}(\mathbf{x})$ can be built by various linear regression methods [20], such as ordinary least squares [21], principal component regression [22,23], ridge regression [24,25] and partial least squares [26,27]. In general, the local linear model $\phi_r^{(s)}(\mathbf{x})$ can be expressed as:

$$\tilde{y}_r^{(s)}(\mathbf{x}) = \phi_r^{(s)}(\mathbf{x}) = \phi_r^{(s)}(\mathbf{x};\boldsymbol{\beta}_s), \tag{2.10}$$

where $\boldsymbol{\beta}_s$ are the regression associated parameters. Given a set of training data, the learning process includes the calculation of the regression parameters $\boldsymbol{\beta}_s$ and the distance metric $\mathbf{D}_s$. In this work, for mathematical convenience, we use the Cholesky decomposition of $\mathbf{D}_s$, where $\mathbf{D}_s=\mathbf{M}_s^T\mathbf{M}_s$ and learn the upper triangular matrix $\mathbf{M}_s$ instead of $\mathbf{D}_s$. With S predictions from all the local models, $\tilde{y}_r^{(s)}(\mathbf{x}_q)$, at query point $\mathbf{x}_q$, the final output of LWPR for the r-th output is simply given as follows:

$$\tilde{y}_r(\mathbf{x}_q)=\frac{1}{\sum_s w_s(\mathbf{x}_q)}\sum_s w_s(\mathbf{x}_q)\tilde{y}_r^{(s)}(\mathbf{x}_q). \tag{2.11}$$

2.1.2 Partial least squares

In LWPR [19], Partial Least Squares (PLS) is chosen as the basis for the local linear models $\phi_r^{(s)}(\mathbf{x})$ (Eq. (2.10)). PLS is a regression technique that predicts the dependent response y (here taken as scalar) in terms of the K-th dimensional input $\mathbf{x}$ [26–28] (note that in this section, we work with a generic output y and all subscripts r are dropped). Let us denote with $\mathbf{y}$ ($n\times 1$) and $\mathbf{X}$ ($n\times K$) the centered training output and input data, respectively. PLS regression computes the directions $\mathbf{u}$ ($K\times 1$) in the input space (also called latent vectors) that maximize the covariance between $\mathbf{X}$ and $\mathbf{y}$. The score vectors $\mathbf{z}$ ($n\times 1$) are then formed as a linear combination of the columns of $\mathbf{X}$ with weights $\mathbf{u}$ (in some sense providing the best linear combination of the columns of $\mathbf{X}$ for predicting $\mathbf{y}$). Ordinary linear regression is then performed of $\mathbf{y}$ on the score vectors. The residuals after regressing $\mathbf{y}$ and $\mathbf{X}$ are defined such that orthogonality of the latent vectors is enforced. This ensures that the multiple regressions of $\mathbf{y}$ on the score vectors can be obtained one column at a time. The algorithm iteratively reveals more and more information about the connection between $\mathbf{y}$ and $\mathbf{X}$.

The basic algorithm is summarized below [19]:

---

1. Initialization: $\mathbf{X}_1=\mathbf{X}$, $\mathbf{y}_1=\mathbf{y}$.
2. For $i=1$ to R (number of latent variables) do
   (a) Find the direction that maximizes the correlations between $\mathbf{X}_i$ and $\mathbf{y}_i$: $\mathbf{u}_i=\mathbf{X}_i^T\mathbf{y}_i$.
   (b) Compute the latent variables (also called scores): $\mathbf{z}_i=\mathbf{X}_i\mathbf{u}_i$.
   (c) Compute the regression coefficient: $\beta_i=\mathbf{z}_i^T\mathbf{y}_i/(\mathbf{z}_i^T\mathbf{z}_i)$.
   (d) Update the residual after regressing $\mathbf{y}_i$ on $\mathbf{z}_i$: $\mathbf{y}_{i+1}=\mathbf{y}_i-\beta_i\mathbf{z}_i$.
   (e) Update the residual after regressing $\mathbf{X}_i$ on the score vector $\mathbf{z}_i$: $\mathbf{X}_{i+1}=\mathbf{X}_i-\mathbf{z}_i\mathbf{p}_i^T$, where $\mathbf{p}_i=\mathbf{X}_i^T\mathbf{z}_i/(\mathbf{z}_i^T\mathbf{z}_i)$ (transpose of the vector of regression coefficients obtained from linear regression of the columns of $\mathbf{X}_i$ on $\mathbf{z}_i$). This step enforces the orthogonality condition $\mathbf{X}_{i+1}\mathbf{u}_i=0$.

R is often automatically determined by tracking the mean square error of the prediction [19]. The prediction for a new input $\mathbf{x}_{new}$ is performed by essentially retracing the steps of the algorithm above. Let $\bar{\mathbf{x}}$ and $\bar{y}$ be the mean of the inputs and output. The prediction process goes through the following steps:

1. Initialization: $\mathbf{x}_1=\mathbf{x}_{new}-\bar{\mathbf{x}}$.
2. For $i=1$ to R do
   (a) Compute the latent variables: $z_i=\mathbf{x}_i\mathbf{u}_i$.
   (b) Update the residual of $\mathbf{x}$: $\mathbf{x}_{i+1}=\mathbf{x}_i-z_i\mathbf{p}_i^T$.

The predicted output of the local model is then formed by a linear combination of the latent variables as

$$\phi_s(\mathbf{x}_{new})=\bar{y}+\sum_{i=1}^{R}\beta_i z_i. \tag{2.12}$$

2.1.3 Updating the distance metric of each receptive field

The distance metric $\mathbf{D}_s$, controls the shape and size for the local model $\phi_r^{(s)}(\mathbf{x})$ (Eq. (2.10)). In LWPR [19], the distance metric of each local model starts from a predefined value and then can be adjusted according to the observed data. Specifically, the distance metric for each local model can be learned individually by stochastic gradient descent using a penalized cross-validation cost function [18]:

$$J_s=\frac{1}{\sum_{i=1}^{n}w_s(\mathbf{x}_i)}\sum_{i=1}^{n}\frac{w_s(\mathbf{x}_i)(y_r(\mathbf{x}_i)-\tilde{y}_r^{(s)}(\mathbf{x}_i))^2}{(1-w_s(\mathbf{x}_i)\mathbf{x}_i^T\mathbf{P}\mathbf{x}_i)^2}+\frac{\lambda}{K}\sum_{i,j=1}^{K}(\mathbf{D}_s)_{ij}^2, \tag{2.13}$$

where n denotes the number of data points in the training set and K denotes the dimension of the distance metric $\mathbf{D}_s$ (same as the dimension of the input), $w_s(\mathbf{x}_i)$ is defined in Eq. (2.9), $\mathbf{P}$ corresponds to the inverted weighted covariance matrix of the input data (defined in [19]), $\tilde{y}_r^{(s)}(\mathbf{x}_i)$ is the local prediction given by the s^{th} local model for the r-th output and λ is the trade-off parameter that determines the strength of the penalty term. The first term of the cost function J_s is the mean leave-one-out cross-validation error of the local model which ensures proper generalization [18]. The second regularization term penalizes the sum of squared coefficients of the distance metric $\mathbf{D}_s$ to allow smoother local predictions for the model [18]. Some details on the derivation of Eq. (2.13) are provided in Appendix A. Based on the cost function, the distance metric can be learned via:

$$\mathbf{M}_s^{n+1}=\mathbf{M}_s^n-\alpha\frac{\partial J_s}{\partial\mathbf{M}_s}, \tag{2.14}$$

where $\mathbf{D}_s=\mathbf{M}_s^T\mathbf{M}_s$ and $\mathbf{M}_s$ is an upper triangular matrix, α is the step size [18, 19]. The gradient of J_s can be computed analytically in terms of several sufficient statistics.

Note that the adjustment of the distance metric described above is not capable of modeling alone local features. In Section 2.2, we address how to adaptively define the initial D_s and subsequently control its size.

2.1.4 Adding a receptive field

In LWPR [19], if a training sample $(\mathbf{x},y)$ does not activate any of the existing receptive fields by more than a threshold w_{gen}, i.e., $\max_s w_s(\mathbf{x}) < w_{\text{gen}}$, then a new receptive field is created (w_{gen} is defined by the user, here $w_{\text{gen}} = 0.2$). The center of the new receptive field is taken as $\mathbf{c} = \mathbf{x}$ and the initial distance metric $\mathbf{D}$ is set to a default value, $\mathbf{D}_{\text{def}}$ (usually, $\mathbf{D}_{\text{def}}$ is a diagonal matrix, as $\mathbf{D}_{\text{def}} = a\mathbf{I}_{K\times K}$, where a is problem dependent). $\mathbf{D}_{\text{def}}$ determines the initial size and shape for the local model. It can be understood as the inverse of the covariance matrix of the Gaussian kernel, as shown in Eq. (2.9).

All other regression associated parameters ($\boldsymbol{\beta}$, the sufficient statistics to calculate $\boldsymbol{\beta}$ and $\mathbf{P}$ (Eq. (2.13))) are initialized to predefined values (zero except the matrix $\mathbf{P}$). A suitable initialization of $\mathbf{P}$ is a diagonal matrix with $\mathbf{P}_{ii} = 1/r_i^2$, where the parameters r_i take very small value, e.g., 0.001 [19].

This approach has been shown to be robust but not very accurate. As shown in [19], although the mean squares error (MSE) for all examples considered eventually converges, it actually converges to a rather large value. Furthermore, a large amount of training data were often required to achieve convergence. In addition, it is not appropriate to set the distance metric of each new receptive field to a default value $\mathbf{D}_{\text{def}}$. For example, in problems with strong local features, the default size of the local model is much larger than the span of the local features. One needs to select the initial distance metric based on the local environment. Given a set of training data, we discuss in Section 2.2 when we need to add a local model and how to select the initial distance metric.

2.1.5 Computing confidence intervals

LWPR has the ability to give us a confidence interval for the prediction at a query point [19]. The prediction for a query point $\mathbf{x}_q$ is taken as a noisy observation of the true response, where the noise comes from two sources. The first noise source models the predictive error of local models, in this work, the error bar given by the local PLS method. The second noise process accounts for the difference between predictions of local models. This term comes into the picture because of the Local Weighted Regression framework, since the final prediction is obtained by averaging all the local predictions. The overall predictive variance can be approximated as (see Appendix B):

$$\sigma_{\text{pred}}^2 = \frac{\sum_s w_s(\mathbf{x}_q)\sigma_{pred,s}^2}{(\sum_s w_s(\mathbf{x}_q))^2} + \frac{\sigma^2}{\sum_s w_s(\mathbf{x}_q)}. \tag{2.15}$$

The first term on the righthand of the equation accumulates the predicted variances for all the local models, where $\sigma_{pred,s}^2$ is the local predicted variance for the s^{th} model [19]. The second term calculates the predicted variance due to the overlap of local models, here σ^2 is defined by

$$\sigma^2 = \frac{\sum_s w_s(\mathbf{x}_q)(\tilde{y}(\mathbf{x}_q) - \tilde{y}^{(s)}(\mathbf{x}_q))^2}{\sum_s w_s(\mathbf{x}_q)}. \tag{2.16}$$

Substituting this into Eq. (2.15) results in the following:

$$\sigma^2_{\text{pred}} = \frac{1}{(\sum_s w_s(\mathbf{x}_q))^2} \sum_s^S w_s(\mathbf{x}_q)[(\tilde{y}(\mathbf{x}_q) - \tilde{y}^{(s)}(\mathbf{x}_q))^2 + \sigma^2_{pred,s}]. \tag{2.17}$$

2.2 Adaptive LWPR

Given a set of training data, one can build the LWPR model as summarized in Algorithm 2.1. w_{gen} is the main parameter of this algorithm controlling when a sample point $\mathbf{x}_i$ is to be sent to a particular local model s via the criterion $w_s(\mathbf{x}_i) > w_{gen}$. In this paper, our interest is on developing an adaptive LWPR algorithm (Algorithm 2.2). Based on the standard LWPR model, the following question needs to be addressed: If we are to choose the next observation, what is the most informative input we should select from the input distribution? This is the classical experimental design or active learning problem [29,30]. In [31], the authors concluded that the most informative data point is the one that maximizes the error bar (predictive variance). In this work, we adaptively select the sample which has the maximum predictive variance. However, if the response surface has a local feature (e.g. a discontinuity), then the largest predictive variance always occurs around the local feature. This results in a cluttering of points in the training set around the local features with insufficient observations at other locations. This situation can be avoided by adding a distance penalization factor $\eta(\mathbf{x}_i)$ [32] (Eq. (2.18)) to prevent samples from lying too close to the current training data set. The scaling γ in the definition of the penalization factor attempts to balance the goal of sampling in areas of large predictive variance with the ability to detect unexplored (less-sampled) areas.

Algorithm 2.1: The complete LWPR algorithm

Initialize LWPR model with no local models
for $i = 1,\cdots,l_0$ **do**
  for $s = 1,\cdots,S_{curr}$ (S_{curr} is the number of current local models) **do**
    Calculate the weight $w_s(\mathbf{x}_i)$ from Eq. (2.9)
    if $w_s(\mathbf{x}_i) > w_{gen}$ **then**
      Update the PLS regression parameters to include $\mathbf{x}_i$ (Section 2.1.2)
      Update the distance metric $\mathbf{D}_s$ to include $\mathbf{x}_i$ (Section 2.1.3)
    end if
  end for
  if no current local model was activated by more than w_{gen} **then**
    Create a new local model centered at $\mathbf{x}_i$ with an initial distance metric $\mathbf{D}_{\text{def}}$ (Section 2.1.4)
  end if
end for

The penalty function $\eta(\mathbf{x}_i)$ is introduced with the following properties (Algorithm 2.2). At first, it should give a very small value when there exists a data point in the training set that is very close to the candidate sample $\mathbf{x}_i$. In that way, this candidate

point will not be selected simply because the predictive variance at this point is large. Secondly, $\eta(\mathbf{x}_i)$ should take a relatively large value if no points in the training set are close to the candidate sample. In this work, the penalty function contains a parameter γ that controls its strength. After several tests, we here select $\gamma=10^{-2}$. We can now select as a candidate sample for our regression scheme the sample h out of N samples from the distribution $p(\mathbf{x})$ that maximizes the weighted predictive variance (Eq. (2.19)). Therefore, one can build an adaptive version of LWPR (ALWPR) by selecting the most informative input samples from the input space $\mathcal{X}$ (Eq. (2.1)) biased by the input probability distribution.

Algorithm 2.2: The complete adaptive LWPR framework

Start with l_0 initial training points, construct a LWPR model (Algorithm 2.1).
Set the initial ξ larger than δ.
while $\xi>\delta$ **do**
Randomly sample N data points from $p(\mathbf{x})$.
Calculate the predictive variance $\sigma^2(\mathbf{x}_i)$ for each sample $\mathbf{x}_i$.
Calculate the distance factor $\eta(\mathbf{x}_i)$ as

$$\eta(\mathbf{x}_i)=1-\exp\Big\{-\gamma\min_j(\mathbf{x}_i-\mathbf{x}^{(j)})^T(\mathbf{x}_i-\mathbf{x}^{(j)})\Big\}, \tag{2.18}$$

where $\mathbf{x}^{(j)}$ is one point in the training set.
The one that maximizes the weighted predictive variance is chosen,

$$\mathbf{x}_{new}=\mathbf{x}_h, \text{ where } h=\operatorname{argmax}_i(\sigma^2(\mathbf{x}_i)p(\mathbf{x}_i)\eta(\mathbf{x}_i)), \tag{2.19}$$

Send the new selected input $\mathbf{x}_{new}$ to the deterministic solver.
Calculate ξ as

$$\xi=\int\sigma^2(\mathbf{x})\eta(\mathbf{x})p(\mathbf{x})d\mathbf{x}\approx\frac{1}{N}\sum_{i=1}^{N}\sigma^2(\mathbf{x}_i)\eta(\mathbf{x}_i)p(\mathbf{x}_i). \tag{2.20}$$

Update the LWPR model with the new training data $\{\mathbf{x}_{new},y_{new}\}$.
if $|\widetilde{y}_{new}-y_{new}|>\epsilon$ **then**
for $s=1,\cdots,S_{curr}$ **do**
Calculate the weights $w_s(\mathbf{x}_{new})=\exp\{-\frac{1}{2}(\mathbf{x}_{new}-\mathbf{c}_s)^T\mathbf{D}_s(\mathbf{x}_{new}-\mathbf{c}_s)\}$.
if $w_s(\mathbf{x}_{new})>w_e$ **then**
Reset the distance metric as:

$$\mathbf{D}_s=\frac{1}{\alpha_s^2}I_{K\times K}, \tag{2.21a}$$

$$\text{where}\quad \alpha_s=\frac{1}{2}\sqrt{(\mathbf{x}_{new}-\mathbf{c}_s)^T(\mathbf{x}_{new}-\mathbf{c}_s)}. \tag{2.21b}$$

Relearn the regression coefficients of local model s.
end if
end for

Create a new receptive field centered at $\mathbf{x}_{new}$
Set the initial distance metric as:

$$\mathbf{D}_{new} = \frac{1}{\alpha_{\min}^2} I_{K\times K}, \tag{2.22a}$$

$$\text{where} \quad \alpha_{\min} = \arg\min_s \frac{1}{2}\sqrt{(\mathbf{x}_{new}-\mathbf{c}_s)^T(\mathbf{x}_{new}-\mathbf{c}_s)}. \tag{2.22b}$$

end if
end while

The convergence criterion $\xi < \delta$ (with δ a given tolerance that defines a stopping criterion for the Algorithm 2.2) is chosen with ξ defined in Eq. (2.20) as the average weighted predictive variance $\sigma_i^2\eta_i$ over the whole domain biased by the input distribution. As we know, the predictive variance around local features is higher than in other regions, however, the probability that candidate samples go to the local feature region is relatively small (discontinuities have zero probability measure). As the number of data points observed increases, ξ will gradually decrease leading to higher reliability of the prediction over the whole domain.

The Algorithm 2.2 starts with l_0 training points and constructs an initial LWPR model (see Algorithm 2.1). The value of l_0 is selected based on the input dimensionality (e.g., in this work, we set $l_0 = 50$ for $K = 2$ and $l_0 = 100$ for $K = 4$). Then N candidate points (here $N = 1000$) are sampled from the input distribution $p(\mathbf{x})$. From them, we add to the training data set the input point $\mathbf{x}_{new}$ that maximizes the weighted predictive variance (Eq. (2.19)). The next algorithmic step is updating the LWPR model using the new training set $\{\mathbf{x}_{new}, y_{new}\}$. This step includes the creation of a new receptive field if $\max_s w_s(\mathbf{x}_{new}) < w_{gen}$, or otherwise updating all neighboring local models. These calculations were discussed in the earlier sections.

If a new receptive field is created at a point at $\mathbf{x}_{new}$, the prediction error at that point will be close to zero (not exactly zero owing to the contribution from other local models). Let us assume now that no new receptive field was created at $\mathbf{x}_{new}$. In this case, we will need to check if the update of neighboring local models provides a reasonable prediction y_{new} at $\mathbf{x}_{new}$. The parameter ϵ is introduced to control this error. In particular, when $|\widetilde{y}_{new} - y_{new}| > \epsilon$ (with $\widetilde{y}_{new}$ the estimate at $\mathbf{x}_{new}$ before the update), a new receptive field will be introduced centered at $\mathbf{x}_{new}$ with an appropriate distance metric (Eqs. (2.22a) and (2.22b)). A user-defined parameter w_e is introduced to control the update of the neighboring local models. In particular, the models s for which $w_s(\mathbf{x}_{new}) > w_e$ are updated. The parameter w_e controls in certain way the overlap of these neighboring to $\mathbf{x}_{new}$ local models. The updated distance metric for these models s is taken as in Eqs. (2.21a) and (2.21b) and is such that the influence of local models at the point $\mathbf{x}_{new}$ is decreased, i.e., the weight $w_s(\mathbf{x}_{new})$ reduces from some large value to 0.135. This value is obtained from the definition of the weight in Eq. (2.9) using the distance metric given in Eq. (2.21b).

Now suppose $w_e < w_s(\mathbf{x}_{new}) < 0.135$. This means that the current model s needs to

be updated. Thus the weight $w_s(\mathbf{x}_{new})$ will increase to 0.135. With a fixed center of the local model s and fixed $\mathbf{x}_{new}$, the only way for this to happen is by increasing the size of the local model s, which of course is not desirable. This implies that we need to choose $w_e > 0.135$. For w_e close to 1, only the local models that are very close to $\mathbf{x}_{new}$ will be updated. From numerical experimentation, we found that the optimal choice for all the examples reported in this paper is $w_e = 0.3$.

Yet another consideration is relearning of the neighboring local models s after the update of the new distance metric. This is because several points that were previously included in model s may not satisfy the condition $w_s(\mathbf{x}) > w_{gen}$. Thus we need to remove these points from model s (i.e. clean the sufficient statistics of local model s) and recalculate its regression parameters. Lastly, note that in order to allow reasonable local predictions, at least $K+1$ points need to be observed at each local model.

Depending on the input distribution $p(\mathbf{x})$, the ALWPR algorithm will result in the creation of new models for many of the added points in the non-smooth regions of the stochastic space. In general for smooth stochastic responses, this will not be the case and the ALWPR algorithm will preferably refine the parameters in current local models than adding new local models.

For the multi-output case, we assume that the output dimensions are independent from each other. Thus we construct M independent K-inputs-to-single-output ALWPR models. All ALWPR models share the same training input data. When calculating the predictive variance for a sample input $\mathbf{x}_i$, we accumulate the predictive variances of all the outputs, $\sigma^2(\mathbf{x}_i) = \sum_{r=1}^{M} \sigma_r^2(\mathbf{x}_i)$, where $\sigma_r^2(\mathbf{x}_i)$ is the predictive variance at input $\mathbf{x}_i$ for the r^{th} output. The convergence criterion in Eq. (2.20) is then adjusted with ξ defined as follows:

$$\xi = \int \sigma^2(\mathbf{x})\eta(\mathbf{x})p(\mathbf{x})d\mathbf{x} \approx \frac{1}{N}\sum_{i=1}^{N}\sum_{r=1}^{M}\sigma_r^2(\mathbf{x}_i)\eta(\mathbf{x}_i)p(\mathbf{x}_i). \tag{2.23}$$

To accelerate the data selection process, instead of taking one input sample with the largest weighted predictive variance, we can take the first n input samples with the highest weighted predictive variance. Using these input points, we can run the deterministic solver independently using different processors. After all the calculations are completed, we gather all the outputs and include these new observations into the training set to update the model. Note that for the M multi-output case, all the M LWPR models share the same input training data, but they do not have the same local models.

For $q+n+1$ processors available, we take one processor (P_0) as the root node, q processors (P_1 to P_q) to build the LWPR model for the M-output problem and n processors (P_{q+1} to P_{q+n}) to run the deterministic solver. At first, the root node sends the training data set to processors P_1 to P_q. After the model has been built, the root node will receive a signal from P_1 to P_q, then it will start to sample N points from the input distribution $p(\mathbf{x})$. These samples are going to be sent to P_1 to P_q to calculate the predictive variance for each output, while the root node is calculating the value of the distance penalty function for each sample. After all the predictive variances for all the outputs have been calcu-

lated and sent back to the root node, the root node sums them up, finds the first n largest weighted predictive variances and calculates ξ using Eq. (2.23). Finally, the new selected n inputs will be sent to processors P_{q+1} to P_{q+n} to run the deterministic solver. For each output, we check the prediction for the newly added points. If it is not satisfying the set accuracy requirements at a newly added point, a local model centered at this point will be added and the size of the neighboring local models will manually be changed. The process is repeated until convergence. The calculated responses are sent to the root node and added to the training set.

3 Numerical examples

The examples considered here are designed to demonstrate that ALWPR has the ability to learn local features in the stochastic space such as discontinuities, adaptively choose the inputs for the model (active learning/experimental design) and accurately provide predictions with uncertainty for a new input.

All the examples are run on massively parallel computers at the National Energy Scientific Computing Center (NERSCC) [33].

3.1 Kraichnan-Orszag (K-O) problem

The transformed Kraichnan-Orszag three-mode problem is expressed as the following dynamical system [8,14]

$$\frac{dy_1}{dt}=y_1y_3,\qquad \frac{dy_2}{dt}=-y_2y_3,\qquad \frac{dy_3}{dt}=-y_1^2+y_2^2, \tag{3.1}$$

subject to initial conditions

$$y_1(0)=Y_1(0;\omega),\qquad y_2(0)=Y_2(0;\omega),\qquad y_3(0)=Y_3(0;\omega). \tag{3.2}$$

The problem exhibits a bifurcation on the parameters $y_1(0)$ and $y_2(0)$, in particular a discontinuity occurs when the initial conditions cross the planes of $y_1=0$ and $y_2=0$. The deterministic solver we use is a 4-th order Runge-Kutta method as implemented in the GNU Scientific Library [33]. We solve the system for the time interval $[0,10]$ and record the responses at time step interval of $\Delta t=0.01$. This results in a total of $M=300$ outputs (100 for each of the three dimensions of the response). The error of the statistics will be evaluated using the (normalized) L_2 norm of the error in variance defined by:

$$E_{L_2}=\frac{1}{M}\sum_{r=1}^{M}(v_{r,MC}-\widetilde{v}_r)^2, \tag{3.3}$$

where $v_{r,MC}$ is the Monte Carlo estimate of the variance using 10^6 samples and $\widetilde{v}_r$ is the predictive variance, $r=1,\cdots,M$.

For brevity, we only consider here the two dimensional case. The initial conditions for the problem are taken as:

$$y_1 = 0, \qquad y_2 = 0.1x_1, \qquad y_3 = x_2,$$

where

$$x_i \sim \mathcal{U}([-1,1]), \qquad i = 1,2.$$

This problem has a line discontinuity at $x_1 = 0$. The algorithm starts with 50 random samples. Figs. 1-3 show the comparison of the prediction of $y_3(t=10)$ with the true response at tolerance levels $\delta = 10^{-5}, 10^{-7}$ and 10^{-9}, respectively. As shown in these figures, with decreasing δ, most of the new samples selected by the algorithm are placed near the discontinuity that is gradually being resolved. Fig. 4 depicts the mean weighted predictive variance and L_2 norm of the error in variance as a function of the number of observations. Here, we also compare the results obtained from the ALWPR with those of the Adaptive Sparse Grid Collocation Method (ASGC) [14] and Monte Carlo method. As shown in the figure, for this KO-2 problem, for the same number of samples, ALWPR leads to higher accuracy than ASGC. Fig. 5 plots the predictive mean and variance of $y_3(t)$ as a function of time t and compares it with the MC prediction. Finally, using 10^5 samples, Fig. 6 provides the kernel density estimation of the PDF of $y_2(t=10)$ and $y_3(t=10)$. This example is run in parallel with 24 processors ($q = 20$ and $n = 4$).

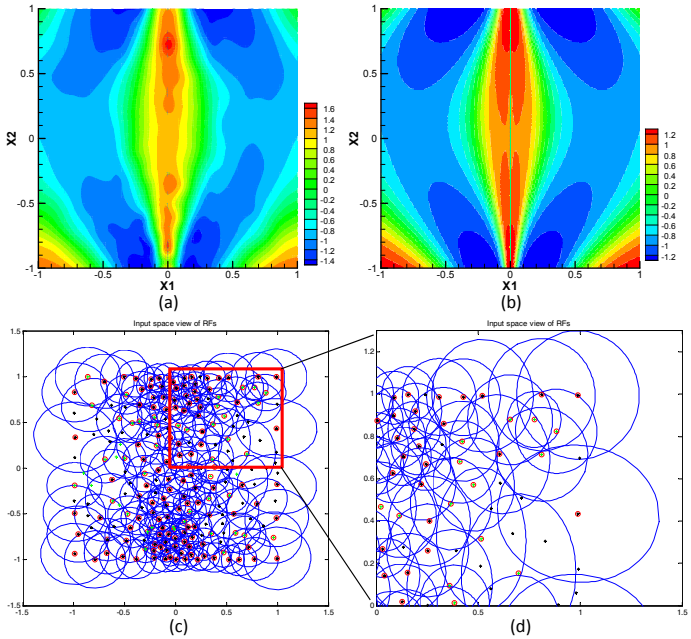

Figure 1: KO-2: (a) The prediction of $y_3(t=10)$ at $\delta = 10^{-5}$. (b) The true response of $y_3(t=10)$. (c) Final Receptive fields. (d) Initial data (red squares) and new samples selected by ALWPR (green squares).

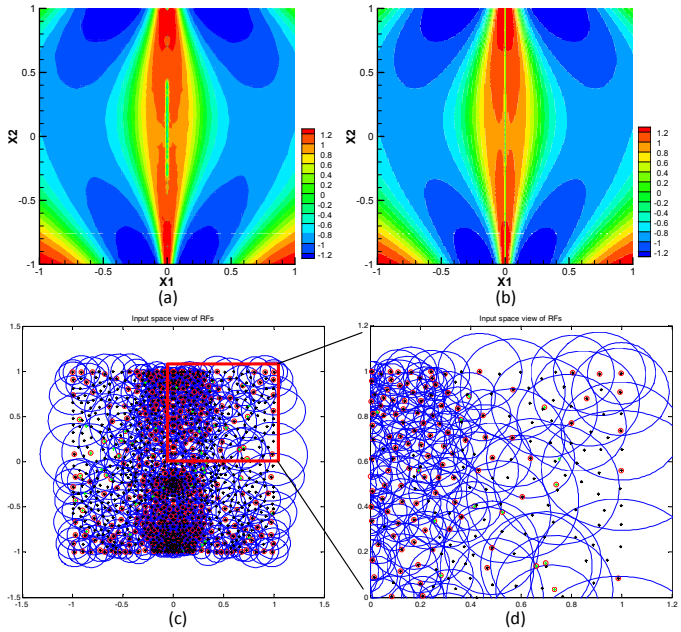

Figure 2: KO-2: (a) The prediction of $y_3(t=10)$ at $\delta=10^{-7}$. (b) The true response of $y_3(t=10)$. (c) Final Receptive fields. (d) Initial data (red squares) and new samples selected by ALWPR (green squares).

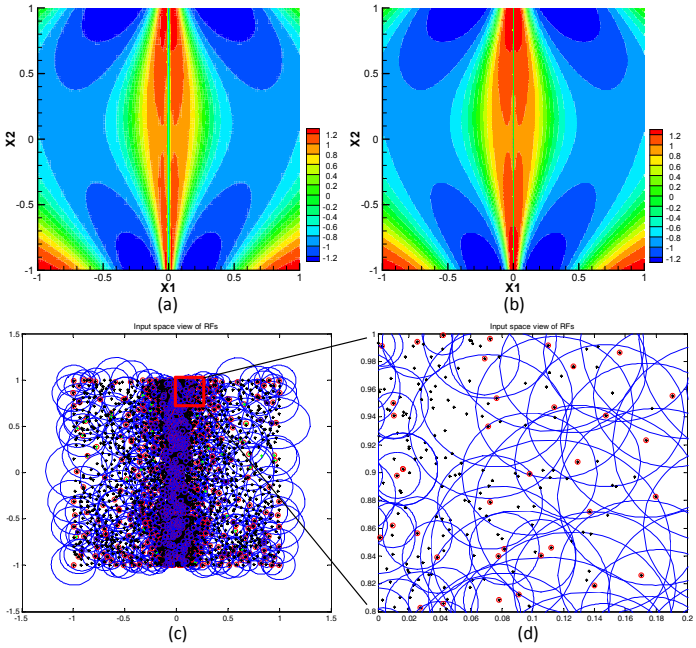

Figure 3: KO-2: (a) The prediction of $y_3(t=10)$ at $\delta=10^{-9}$. (b) The true response of $y_3(t=10)$. (c) Final Receptive fields. (d) Initial data (red squares) and new samples selected by ALWPR (green squares).

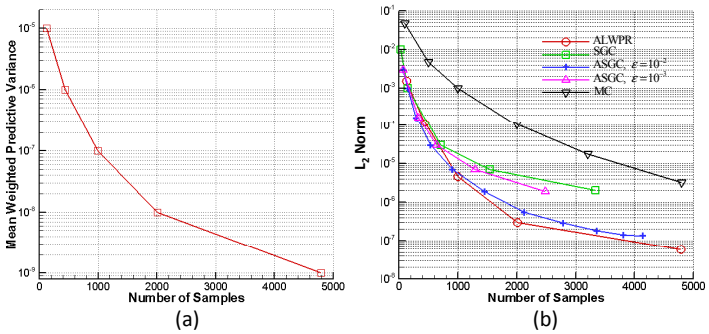

Figure 4: KO-2: (a) The mean weighted predictive variance as a function of the number of samples observed. (b) The L_2 norm of the error in variance as a function of the number of samples observed for ALWPR, Sparse Grid Collocation (SGC), Adaptive Sparse Grid Collocation (ASGC) and Monte Carlo (MC).

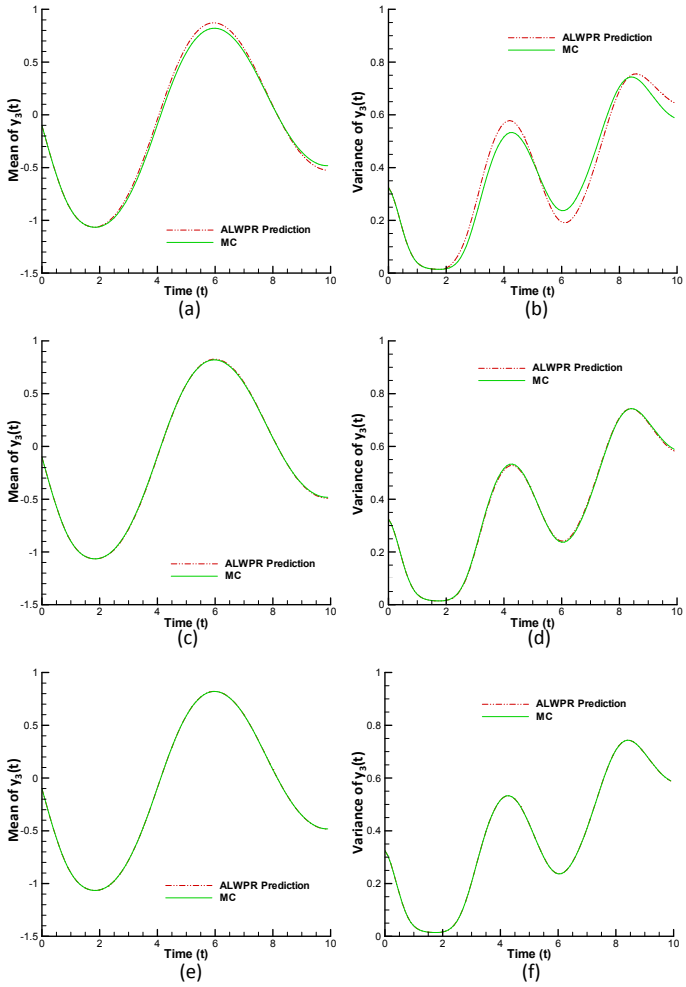

Figure 5: KO-2: Predictive mean (red) versus MC estimate (green) of the mean (left column) and variance (right column) of $y_3(t)$ for $\delta=10^{-5}$, 10^{-7} and 10^{-9} (from top to bottom, respectively).

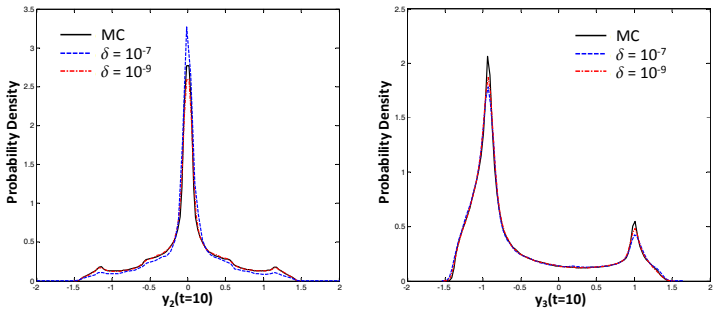

Figure 6: KO-2: Kernel density estimation of the PDF of $y_2(t=10)$ (left) and $y_3(t=10)$ (right) using 10^5 samples.

3.2 Horn problem

In this section, we apply the ALWPR method to the planar acoustic horn problem [34–36], in the form of the two-dimensional Helmholtz equation in random media. The structure of the horn is depicted in Fig. 7. The incoming wave comes from the left end and propagates to the right end through a horn-like tunnel. The walls of the tunnel are built by sound-hard material. The governing equations for the (complex) pressure are:

$$\nabla^2 p(x,y,\omega)+k^2(1+n^2(x,y,\omega))p(x,y,\omega)=0, \tag{3.4}$$

with boundary conditions

$$\begin{aligned}
&\frac{\partial p}{\partial \vec{n}}-ikp=0, && \text{on } \Gamma_1,\\
&\frac{\partial p}{\partial \vec{n}}=0, && \text{on } \Gamma_2,\\
&p(x,y,\omega)=f(x,y), && \text{on } \Gamma_3,
\end{aligned}$$

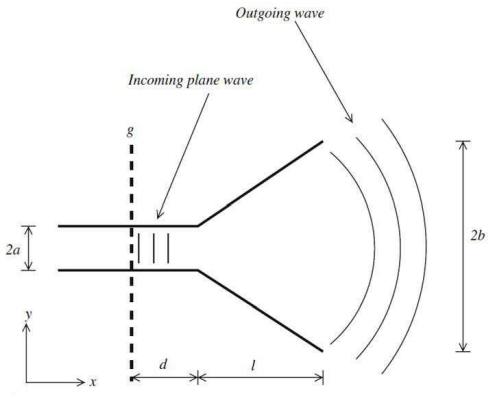

Figure 7: The structure of the horn, the incoming wave comes from the left end and propagates to the right end through a horn-like tunnel, where the walls of the tunnel are built by sound-hard material.

where $\vec{n}$ is the unit outer-pointing normal of the boundary, k is the wave number and $n^2(x,y,\omega)$ is the random reflectivity of the media. Γ_1 is the outer boundary, Γ_2 is the boundary of the tunnel (horn) and Γ_3 is the source boundary for the incoming wave (inlet of the horn). In this example, the random reflectivity of the media is chosen to be [37]

$$n^2(x,y,\omega)=\sum_{i=1}^{4}\xi_i(\omega)\psi_i(x,y),$$

where $\{\xi_i(\omega)\}$ are i.i.d. uniformly distributed random variables in $[0,1]$ and the functions $\{\psi_i(x,y)\}$ are given by

$$\psi_1(x,y)=\sin^2(2\pi x)\sin^2(2\pi y), \qquad \psi_2(x,y)=\sin^2(4\pi x)\sin^2(4\pi y),$$
$$\psi_3(x,y)=\sin^2(6\pi x)\sin^2(4\pi y), \qquad \psi_4(x,y)=\sin^2(6\pi x)\sin^2(6\pi y).$$

The deterministic problem is solved by using the FreeFEM++ software [38] with $f(x)=1$ and $k=0.7$. We consider a circular domain discretized with triangular elements with totally 3942 nodes, as shown in Fig. 8. In the context of the ALWPR method, this is a regression problem with 3942 outputs. In this example, the geometric parameters (see Fig. 8) are chosen as follows: $a=1.6$, $b=4$, $l=4$, $d=4$, $R=9.6$ (the radius of the circular domain).

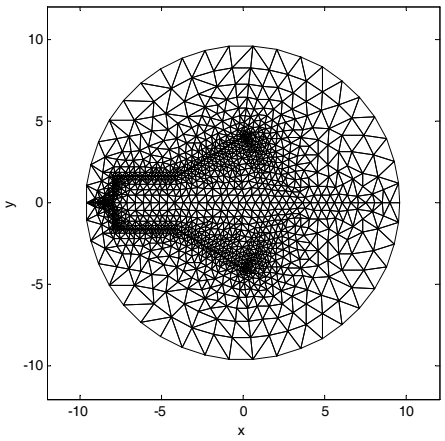

Figure 8: The FEM mesh used in the horn problem (3942 nodes).

The horn problem is studied with a four-dimensional random input using 100 initial samples. The convergence plots of the mean weighted predictive variance and L_2 norm of the error in variance are shown in Fig. 9. The obtained results are compared with the results of ASGC and MC method. Fig. 10 compares the predictive mean and variance given by ALWPR with the corresponding MC estimates obtained with 10^6 samples. Notice that as the threshold δ decreases, the predictive variance is almost identical to the MC estimates. In order to see the predictive capabilities of ALWPR for this problem, we

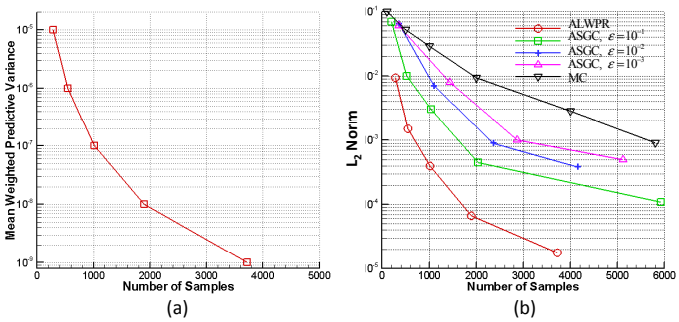

Figure 9: Horn (4 input dimensions): (a) The mean weighted predictive variance as a function of the number of samples observed. (b) The L_2 norm of the error in variance as a function of the number of samples used by ALWPR, ASGC and MC.

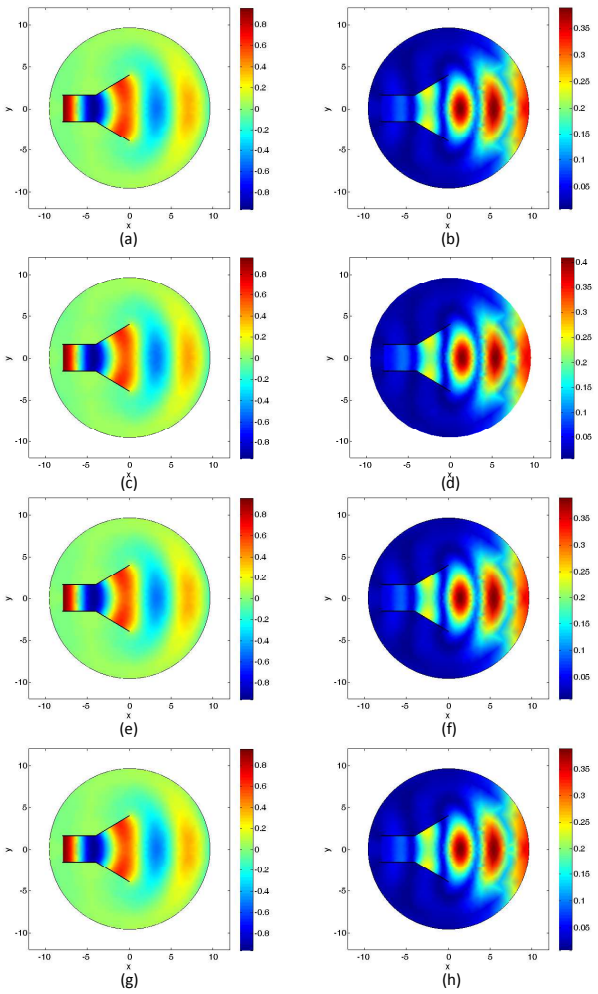

Figure 10: Horn (4 input dimensions): Comparison of the predictive variances using ALWPR with the MC estimates using 10^6 samples. The first row provides the MC mean (a) and the MC std (b). The next three rows are the predicted mean and predicted std with $\delta=10^{-5}$, 10^{-7} and 10^{-9}, respectively.

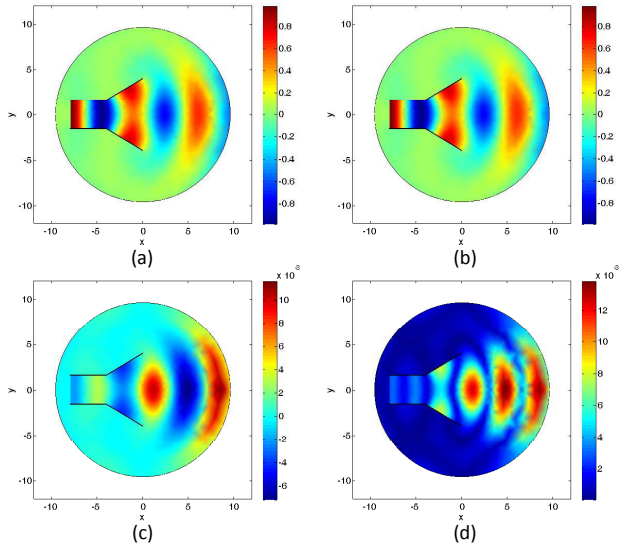

Figure 11: Horn (4 input dimensions): Comparison of the prediction at a random input point with the true response. (a) prediction given by ALWPR, (b) true response, (c) difference between the prediction and the true response and (d) predictive variance given by ALWPR.

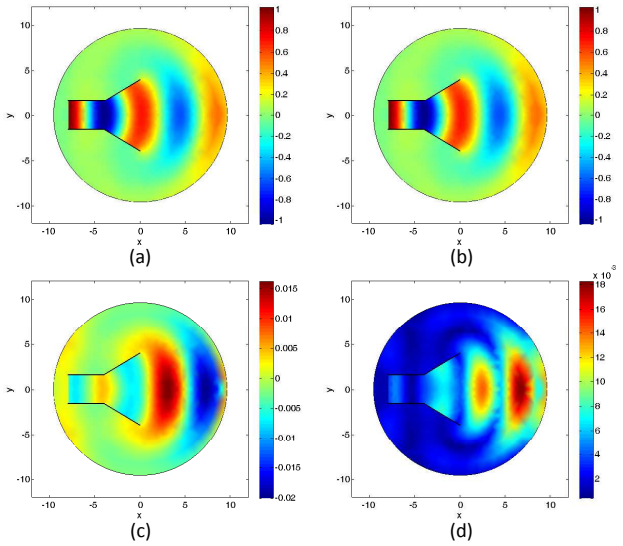

Figure 12: Horn (4 input dimensions): Comparison of the prediction at a random input point with the true response. (a) prediction given by ALWPR, (b) true response, (c) difference between the prediction and the true response and (d) predictive variance given by ALWPR.

plot the prediction at $\delta = 10^{-9}$ on two random input samples and compare them with the true responses, as shown in Figs. 11 and 12. One can notice that the predictions agree very well with the true responses. Also to better examine the performance of ALWPR, we compare the predictive PDFs for the outputs at two specific nodes, $p(-4,1.6)$, the junc-

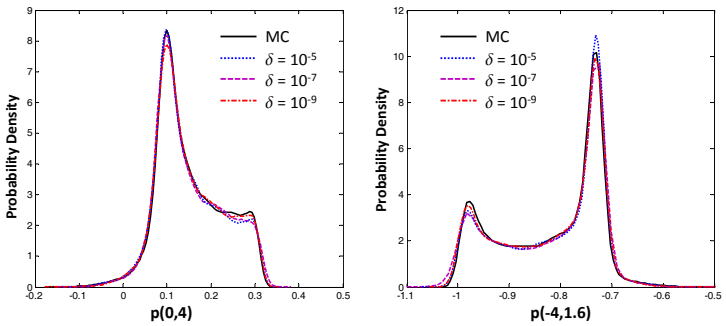

Figure 13: Horn (4 input dimensions): Comparison of the predictive PDF at two different spatial points using ALWPR with the corresponding MC predictions.

tion point of the throat and horn flare, where the incoming wave first enters the divergent region of the horn and $p(0,4)$, the end of the horn flare where the wave leaves the horn. Fig. 13 provides the kernel density estimation of the PDFs at these two points by using 10^5 samples. We can see that the predicted PDFs agree well with the MC estimates. This example is run in parallel with 205 processors ($q=200$ and $n=5$).

3.3 Elliptic problem

In this section, we consider a benchmark stochastic elliptic problem:

$$-\nabla\cdot(a_K(\mathbf{x},\cdot)\nabla u(\omega,\cdot))=f(\cdot), \quad \text{in } D, \tag{3.5a}$$

$$u(\omega,\cdot)=0, \quad \text{on } \partial D, \tag{3.5b}$$

where the physical domain is $D=[0,1]^2$. In order to avoid confusion with the physical dimension $\mathbf{x}=(x,y)$, ω is used to denote the random variables instead of x. We choose a smooth deterministic load

$$f(x,y)=100\cos(x)\sin(y), \tag{3.6}$$

and work with homogeneous boundary conditions. The deterministic problem is solved with the finite element method using a 20×20 grid of bilinear quadrilateral elements. The random diffusion coefficient $a_K(\omega,\mathbf{x})$ is constructed as

$$\log(a_K(\omega,x,y)-0.5)=1+\omega_1\left(\frac{\sqrt{\pi}L}{2}\right)^2+\sum_{k=2}^{K}\xi_k\phi_k(x)\omega_k, \tag{3.7}$$

where

$$\xi_k:=(\sqrt{\pi}L)^{\frac{1}{2}}\exp\left(-\frac{(\lfloor\frac{k}{2}\rfloor\pi L)^2}{8}\right), \quad \text{for } k\geq2,$$

and

$$\phi_k(x) := \begin{cases} \sin\left(\frac{\lfloor \frac{k}{2} \rfloor \pi x}{L_p}\right), & \text{if } k \text{ is } even, \\ \cos\left(\frac{\lfloor \frac{k}{2} \rfloor \pi x}{L_p}\right), & \text{if } k \text{ is } odd. \end{cases} \tag{3.8}$$

We choose ω_k, $k=1,\cdots,K$ to be independent identically distributed random variables

$$\omega_k \sim \mathcal{B}eta([2,5]).$$

While this problem has been studied before with polynomial chaos and sparse grid approaches using uniform random variables, we select variables following the $\mathcal{B}eta$ distribution in order to demonstrate the ability of the algorithm to bias the sample selection based on the input probability distribution. Hence, the stochastic input space is $\Omega=[0,1]^K$. Finally, we set

$$L_p = \max\{1, 2L_c\} \quad \text{and} \quad L = \frac{L_c}{L_p},$$

where L_c is called the correlation length. The expansion Eq. (3.7) resembles the Karhunen-Loève expansion of a two-dimensional random field with stationary covariance

$$Cov[\log(a_K - 0.5)]((x_1,y_1),(x_2,y_2)) = \exp\left\{-\frac{(x_1-x_2)^2}{L_c^2}\right\}.$$

In this example, we set the correlation length to L_c=0.6 and study the problem with K=40 input dimensions. The number of initial samples is chosen to be 1000. The convergence plots of the mean weighted predictive variance and L_2 norm of the error in variance are shown in Fig. 14. A comparison with the results obtained using the ASGC method and

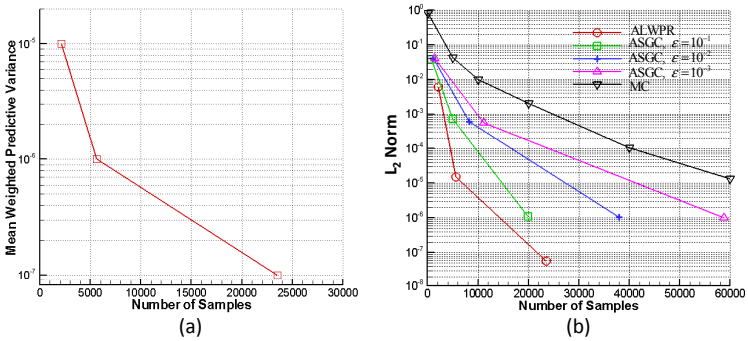

Figure 14: Elliptic example (40 input dimensions): (a) The mean weighted predictive variance as a function of the number of samples observed. (b) The L_2 norm of the error in variance as a function of the number of samples used by ALWPR, ASGC and MC.

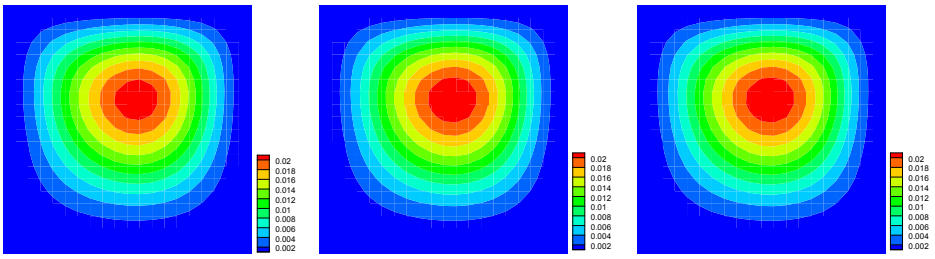

Figure 15: Elliptic example (40 input dimensions): Comparison of the predictive variances using ALWPR with (a) $\delta=10^{-5}$, (b) $\delta=10^{-7}$ and (c) a MC simulation using 10^6 samples.

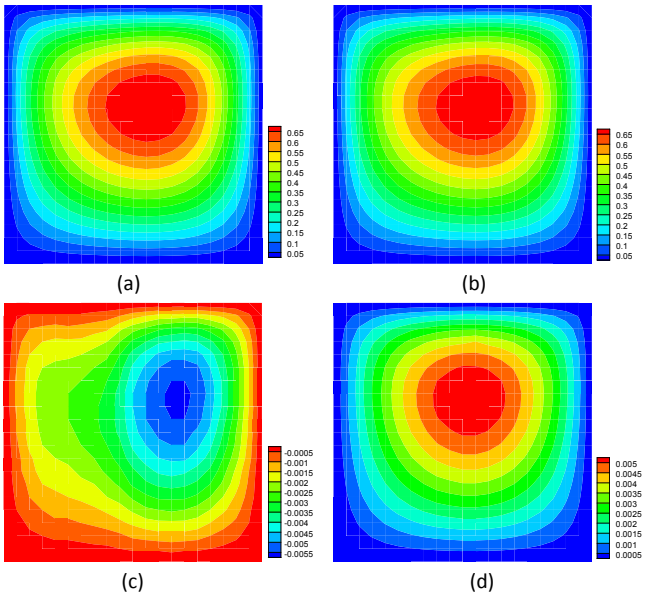

Figure 16: Elliptic example (40 input dimensions): Comparison of the prediction at a random input point with the true response. (a) Prediction given by the ALWPR, (b) True response, (c) Difference between the prediction and the true response and (d) Predictive variance given the ALWPR.

MC method is also shown. Fig. 15 plots the predicted variance of the response for $K=40$ against the MC estimate with 10^6 samples. Notice that as the threshold δ decreases, the predicted variance becomes indistinguishable from the MC estimates. Figs. 16 and 17 show the predictive capabilities of ALWPR for $K=40$ at $\delta=10^{-7}$ on two random input points. The predictions agree very well with the true responses. Also, by using 10^5 samples, we compare the predictive PDFs for two randomly selected outputs, $u(0.4,0.15)$ and $u(0.5,0.5)$, as shown in Fig. 18. It can be seen that the predicted PDFs are in good agreement with those obtained from MC. This example was run in parallel with 300 processors ($q=200$ and $n=100$).

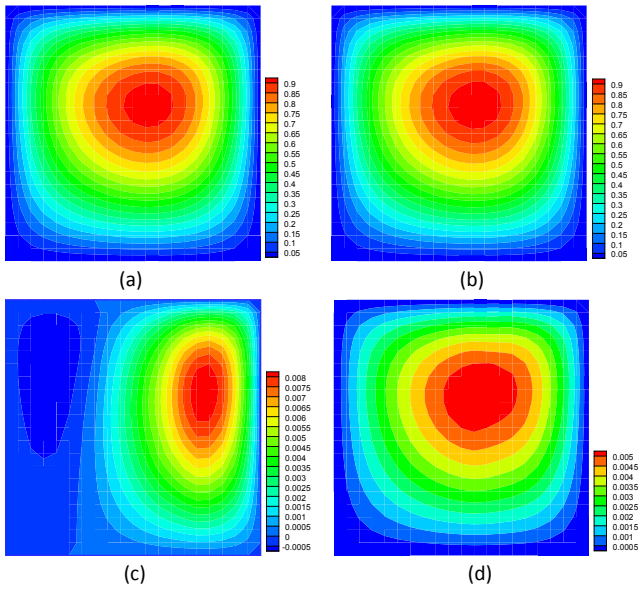

Figure 17: Elliptic example (40 input dimensions): Comparison of the prediction at a random input point with the true response. (a) Prediction given by the ALWPR, (b) True response, (c) Difference between the prediction and the true response and (d) Predictive variance given the ALWPR.

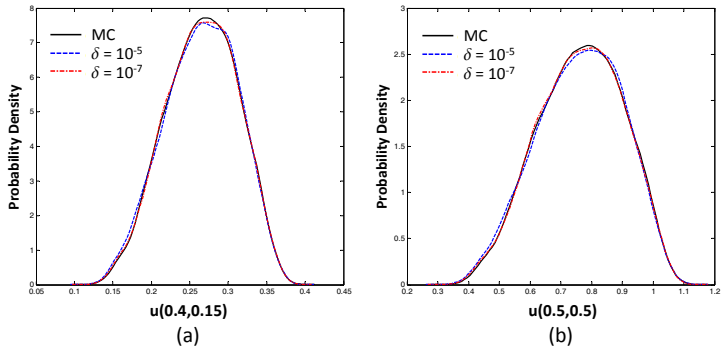

Figure 18: Elliptic example (40 input dimensions): Comparison of the predictive PDF at two different spatial points using ALWPR with the MC predictions.

4 Conclusions

An adaptive implementation of the locally weighted projection regression method was considered and applied to uncertainty quantification problems. The method works for any input distribution and provides predictions with error-bars at any query point. It can deal with multi-outputs and uses active learning in the selection of new sample input points. The selection of new input points is based on the predictive variance and an additional distance penalty term. Also, the method is capable of assigning a proper initial value of the distance metric for each local model depending on the local environment.

Once the model is successfully built, it can provide rapid predictions at new query points thus making the ALWPR framework an inexpensive surrogate of the direct solver.

Various examples were considered to study the accuracy and efficiency of the developed ALWRP method. It was shown that the method is capable of predicting the correct statistics in the presence of discontinuities in the stochastic space. In the high-dimensional elliptic problem considered, the scheme captured well the first- and second-order statistics and also provided reasonable predictions of the PDFs of the outputs. It is clear that at higher dimensions the performance of the method will be limited from issues related to the curse-of-dimensionality. The presented methodology treats multiple outputs in an independent fashion thus it cannot accurately predict correlations among them. This certainly can be a promising direction for expanding the framework. Finally, a complete Bayesian treatment of locally weighted progression regression if of current interest and work in this area will be reported in future works.

Acknowledgments

This research was supported by an OSD/AFOSR MURI09 award on uncertainty quantification, the U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research and the Computational Mathematics program of the National Science Foundation (NSF) (award DMS-0809062). This research used resources of the National Energy Research Scientific Computing Center, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231. Additional computing resources were provided by the NSF through TeraGrid resources provided by NCSA under grant number TG-DMS090007.

Appendices

A Update of the distance metric

In Section 2.1.3, the penalized cross-validation cost function J_s (Eq. (2.13)) is defined as the leave-one-out cross-validation error J augmented with a penalty term. We can write the first term J in Eq. (2.13) as:

$$J=\frac{1}{\sum_{i=1}^{n}w_i}\sum_{i=1}^{n}w_i\|y_i-\tilde{y}_{i,-i}\|^2=\frac{1}{\sum_{i=1}^{n}w_i}\sum_{i=1}^{n}w_ie_{i,-i}^2, \tag{A.1}$$

where $\tilde{y}_{i,-i}$ denotes the prediction at the input location $\mathbf{x}_i$ of the i-th data point calculated from training the model with the $i-th$ data point $(\mathbf{x}_i,y_i)$ excluded from the training set. Also, $e_{i,-i}$ denotes the leave-one-out error. In the following, we use the subscript $()_{i,-i}$ to denote the corresponding variables to the model without using the i-th data point.

In the weighted linear regression system, the parameter $\boldsymbol{\beta}$ can be calculated by:

$$\boldsymbol{\beta}=(\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}\mathbf{y}=\mathbf{P}\mathbf{X}^T\mathbf{W}\mathbf{y}, \tag{A.2}$$

where $\mathbf{P}$ is the inverted weighted covariance matrix of the input. For mathematical convenience, let $\mathbf{C}=\mathbf{X}^T\mathbf{W}\mathbf{X}=\mathbf{P}^{-1}$ and $\mathbf{d}=\mathbf{X}^T\mathbf{W}\mathbf{y}$. In order to obtain $\tilde{y}_{i,-i}$, we should first compute the regression coefficient $\boldsymbol{\beta}_{i,-i}$. Similarly to Eq. (A.2), we can write:

$$\boldsymbol{\beta}_{i,-i}=(\mathbf{X}_{i,-i}^T\mathbf{W}_{i,-i}\mathbf{X}_{i,-i})^{-1}\mathbf{X}_{i,-i}^T\mathbf{W}_{i,-i}\mathbf{y}_{i,-i}=\mathbf{C}_{i,-i}^{-1}\mathbf{d}_{i,-i}, \tag{A.3}$$

where

$$\mathbf{C}_{i,-i}=\mathbf{C}-\mathbf{x}_i^T w_i\mathbf{x}_i, \qquad \mathbf{d}_{i,-i}=\mathbf{d}-w_i\mathbf{x}_i^T y_i. \tag{A.4}$$

To obtain the inverse of $\mathbf{C}_{i,-i}$, we use the Sherman-Morrison-Woodbury Theorem [18]. A special case for the theorem is given below:

Theorem A.1 (Sherman-Morrison-Woodbury Theorem [39]). *Given an invertible matrix* $\mathbf{A}$ *and column vector* $\mathbf{v}$, *then assuming* $1+\mathbf{v}^T\mathbf{A}^{-1}\mathbf{v}\neq 0$,

$$(\mathbf{A}+\mathbf{v}\mathbf{v}^T)^{-1}=\mathbf{A}^{-1}-\frac{\mathbf{A}^{-1}\mathbf{v}\mathbf{v}^T\mathbf{A}^{-1}}{1+\mathbf{v}^T\mathbf{A}^{-1}\mathbf{v}}. \tag{A.5}$$

Using Theorem A.1, we can thus write:

$$\mathbf{C}_{i,-i}^{-1}=(\mathbf{C}-\mathbf{x}_i^T w_i\mathbf{x}_i)^{-1}=\mathbf{C}^{-1}+\frac{\mathbf{C}^{-1}\mathbf{x}_i^T w_i\mathbf{x}_i\mathbf{C}^{-1}}{1-w_i\mathbf{x}_i\mathbf{C}^{-1}\mathbf{x}_i^T}=\mathbf{P}+\frac{\mathbf{P}\mathbf{x}_i^T w_i\mathbf{x}_i\mathbf{P}}{1-w_i\mathbf{x}_i\mathbf{P}\mathbf{x}_i^T}. \tag{A.6}$$

Using this result and Eq. (A.3), we can express $\boldsymbol{\beta}_{i,-i}$ as:

$$\begin{aligned}
\boldsymbol{\beta}_{i,-i}&=\Big[\mathbf{P}+\frac{\mathbf{P}\mathbf{x}_i^T w_i\mathbf{x}_i\mathbf{P}}{1-w_i\mathbf{x}_i\mathbf{P}\mathbf{x}_i^T}\Big]\Big[\mathbf{d}-w_i\mathbf{x}_i^T y_i\Big],\\
&=\mathbf{P}\mathbf{d}-\mathbf{P}w_i\mathbf{x}_i^T y_i+\frac{\mathbf{P}\mathbf{x}_i^T w_i\mathbf{x}_i\mathbf{P}\mathbf{d}}{1-w_i\mathbf{x}_i\mathbf{P}\mathbf{x}_i^T}-\frac{\mathbf{P}\mathbf{x}_i^T w_i\mathbf{x}_i\mathbf{P}w_i\mathbf{x}_i^T y_i}{1-w_i\mathbf{x}_i\mathbf{P}\mathbf{x}_i^T}\\
&=\boldsymbol{\beta}+\frac{1}{1-w_i\mathbf{x}_i\mathbf{P}\mathbf{x}_i^T}[\mathbf{P}w_i\mathbf{x}_i^T y_i w_i\mathbf{x}_i\mathbf{P}\mathbf{x}_i^T\\
&\quad -\mathbf{P}w_i\mathbf{x}_i^T y_i+\mathbf{P}\mathbf{x}_i^T w_i\mathbf{x}_i\mathbf{P}\mathbf{d}-\mathbf{P}\mathbf{x}_i^T w_i\mathbf{x}_i\mathbf{P}w_i\mathbf{x}_i^T y_i]\\
&=\boldsymbol{\beta}-\frac{\mathbf{P}w_i\mathbf{x}_i^T}{1-w_i\mathbf{x}_i\mathbf{P}\mathbf{x}_i^T}(y_i-\mathbf{x}_i\boldsymbol{\beta}).
\end{aligned} \tag{A.7}$$

Here, $\mathbf{x}_i\boldsymbol{\beta}=\tilde{y}_i$ is the prediction of the linear model. By multiplying the above equation by $\mathbf{x}_i$ and subtracting by y_i, we obtain,

$$\begin{aligned}
e_{i,-i}=y_i-\mathbf{x}_i\boldsymbol{\beta}_{i,-i}&=y_i-\mathbf{x}_i\boldsymbol{\beta}+\frac{\mathbf{x}_i\mathbf{P}w_i\mathbf{x}_i^T}{1-w_i\mathbf{x}_i\mathbf{P}\mathbf{x}_i^T}(y_i-\mathbf{x}_i\boldsymbol{\beta})\\
&=\frac{1}{1-w_i\mathbf{x}_i\mathbf{P}\mathbf{x}_i^T}(y_i-\tilde{y}_i).
\end{aligned} \tag{A.8}$$

Hence, we can write the following:

$$J=\frac{1}{\sum_{i=1}^{n}w_i}\sum_{i=1}^{n}w_i\|y_i-\tilde{y}_{i,-i}\|^2=\frac{1}{\sum_{i=1}^{n}w_i}\sum_{i=1}^{n}w_i\frac{\|y_i-\tilde{y}_i\|^2}{(1-w_i\mathbf{x}_i\mathbf{P}\mathbf{x}_i^T)^2}. \tag{A.9}$$

In ALWPR, the above cost function can be formulated in terms of the PLS projected inputs $\mathbf{z}_i$ as

$$J=\frac{1}{\sum_{i=1}^{n}w_i}\sum_{i=1}^{n}w_i\frac{\|y_i-\tilde{y}_i\|^2}{(1-w_i\mathbf{z}_i\mathbf{P}_z\mathbf{z}_i^T)^2}, \tag{A.10}$$

where $\mathbf{P}_z$ corresponds to the inverse covariance matrix computed from the projected inputs $\mathbf{z}_i$. The proof of $\mathbf{x}_i\mathbf{P}\mathbf{x}_i=\mathbf{z}_i\mathbf{P}\mathbf{z}_i^T$ can be found in Appendix A in [19].

B Combined prediction variance

In LWPR, for each individual local model, we assume that the local prediction is a noisy observation of the true response with two independent noise processes [19]:

$$\tilde{y}^{(s)}(\mathbf{x}_q)=y(\mathbf{x}_q)+\epsilon_1+\epsilon_{2,s}, \tag{B.1}$$

where $\epsilon_1\sim\mathcal{N}(0,\sigma^2/w_s(\mathbf{x}_q))$, $\epsilon_{2,s}\sim\mathcal{N}(0,\sigma^2_{pred,s}/w_s(\mathbf{x}_q))$ and $y(\mathbf{x}_q)=f_r(\mathbf{x}_q)$ is the true response for the output r. Recall from Eq. (2.11) that:

$$\tilde{y}(\mathbf{x}_q)=\frac{1}{\sum_s w_s(\mathbf{x}_q)}\sum_s w_s(\mathbf{x}_q)\tilde{y}^{(s)}(\mathbf{x}_q). \tag{B.2}$$

To simplify the notation, we denote in the following $w_s(\mathbf{x}_q)$ simply as w_s and similarly $\tilde{y}(\mathbf{x}_q)$ as $\tilde{y}$ and $\tilde{y}^{(s)}(\mathbf{x}_q)$ as $\tilde{y}^{(s)}$. The combined predictive variance can now be derived as

$$\begin{aligned}
\sigma^2_{\text{pred}}&=E[\tilde{y}^2]-(E[\tilde{y}])^2=E\left[\left(\frac{\sum_s w_s\tilde{y}^{(s)}}{\sum_s w_s}\right)^2\right]-(E[\tilde{y}])^2\\
&=\frac{1}{(\sum_s w_s)^2}E\left[\left(\sum_s w_s y\right)^2+\left(\sum_s w_s\epsilon_1\right)^2+\left(\sum_s w_s\epsilon_{2,s}\right)^2\right]-(\tilde{y})^2\\
&=\frac{1}{(\sum_s w_s)^2}E\left[\left(\sum_s w_s\epsilon_1\right)^2+\left(\sum_s w_s\epsilon_{2,s}\right)^2\right]\\
&=\frac{1}{(\sum_s w_s)^2}\left[var\left(\sum_s w_s\epsilon_1\right)+var\left(\sum_s w_s\epsilon_{2,s}\right)\right]\\
&=\frac{1}{(\sum_s w_s)^2}\left[\sum_s w_s^2\frac{\sigma^2}{w_s}+\sum_s w_s^2\frac{\sigma^2_{pred,s}}{w_s}\right]\\
&=\frac{\sum_s w_s\sigma^2}{(\sum_s w_s)^2}+\frac{\sum_s w_s\sigma^2_{pred,s}}{(\sum_s w_s)^2}\\
&=\frac{\sigma^2}{\sum_s w_s}+\frac{\sum_s w_s\sigma^2_{pred,s}}{(\sum_s w_s)^2}.
\end{aligned} \tag{B.3}$$

References

[1] M. Grigoriu, Stochastic Systems: Uncertainty Quantification and Propagation (Springer Series in Reliability Engineering Reliability Engineering), 2012.

[2] Y. Cao and T. A. Zang, An efficient Monte Carlo method for optimal control problems with uncertainty, Comput. Opt. Appl., 26(3) (2003), 219–230.

[3] L. Mathelin and T. A. Zang, Stochastic approaches to uncertainty quantification in CFD simulations, Numer. Algorithms, 38 (2005), 209–236.

[4] L. Mathelin, T. A. Zang and F. Bataille, Uncertainty propagation for a turbulent compressible nozzle flow using stochastic methods, AIAA J., 42(8) (2004), 1669–1676.

[5] S. V. Poroseva and J. Letschert, Application of evidence theory to quantify uncertainty in hurricane/typhoon track forecasts, Meteorology and Atmospheric Physics, 97 (2007), 149–169.

[6] R. G. Ghanem and P. D. Spanos, Stochastic Finite Elements: A Spectral Approach, 2003.

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

[8] X. Wan and G. E. Karniadakis, An adaptive multi-element generalized polynomial chaos method for stochastic differential equations, J. Comput. Phys., 209(2) (2005), 617–642.

[9] X. Wan and G. E. Karniadakis, Multi-element generalized polynomial chaos for arbitrary probability measures, SIAM J. Sci. Comput., 28(3) (2006), 901–928.

[10] J. Foo and G. E. Karniadakis, Multi-element probabilistic collocation method in high dimensions, J. Comput. Phys., 229(5) (2010), 1536–1557.

[11] I. Babuska, F. Nobile and R. Tempone, A stochastic collocation method for elliptic partial differential equations with random input data, SIAM Rev., 52 (2) (2010), 317–355.

[12] 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) (2008), 2309–2345.

[13] S. A. Smolyak, Quadrature and interpolation formulas for tensor products of certain classes of functions, Soviet Mathematics, Doklady, 4 (1963), 240–243.

[14] X. Ma and N. Zabaras, An adaptive hierarchical sparse grid collocation algorithm for the solution of stochastic differential equations, J. Comput. Phys., 228(8) (2009), 3084–3113.

[15] I. Bilinois and N. Zabaras, Multi-output local Gaussian process regression: application to uncertainty quantification, J. Comput. Phys., 231 (2012), 5718–5746.

[16] R. B. Gramacy and H. K. H. Lee, Bayesian treed Gaussian Process models with an application to computer modeling, 103 (2008), 1119–1130.

[17] C. G. Atkeson, A. W. Moore and S. Schaal, Locally weighted learning, Artificial Intelligence Rev., 11(1) (1997), 11–73.

[18] S. Schaal and C. G. Atkeson, Constructive incremental learning from only local information, Neural Computation, 10(8) (1998), 2047–2084.

[19] S. Vijayakumar, A. DSouza and S. Schaal, Incremental online learning in high dimensions, Neural Computation, 17(12) (2005), 2602–2634.

[20] H. Hoffman, S. Schaal and S. Vijayakumar, Local dimensionality reduction for non-parametric regression, Neural Process Lett., 29 (2009), 109–131.

[21] Y. Nievergelt, Total least squares: state-of-the-art regression in numerical analysis, SIAM Rev., 36(2) (1994), 258–264.

[22] D. M. Hawkins, On the investigation of alternative regressions by principal component analysis, 22(3) (1973), 275–286.

[23] I. T. Jolliffe, A note on the use of principal components in regression, Journal of the Royal Statistical Society, Series C (Applied Statistics), 31(3) (1982), 300–303.
[24] N. R. Draper and R. C. van Nostrand, Ridge regression and James-Stein estimation: review and comments, Technometrics, 21 (4) (1979), 451–466.
[25] B. F. Swindel, Geometry of ridge regression illustrated, The American Statistician, 35(1) (1981), 12–15.
[26] I. E. Frank and J. H. Friedman, A statistical view of some chemometrics regression tools, Technometrics, 35(2) (1993), 109–135.
[27] S. Wold, A. Ruhe, H. Wold and W. J. III Dunn, The collinearity problem in linear regression. the partial least squares (pls) approach to generalized inverses, SIAM J. Sci. Stat. Comput., 5(3) (1984), 735–743.
[28] H. Abdi, Partial least squares regression and projection on latent structure regression (PLS Regression), Wiley Interdisciplinary Reviews: Computational Statistics, 2(1) (2010), 97–106.
[29] G. E. Box, W. G. Hunter, J. S. Hunter and W. G. Hunter, Statistics for Experimenters: Design, Innovation and Discovery, 2nd Edition, 2005.
[30] N. Rubens, D. Kaplan and M. Sugiyama, Recommender Systems Handbook: Active Learning in Recommender Systems, Springer, 2011.
[31] P. Koutsourelakis and E. Bilionis, Scalable Bayesian reduced-order models for simulating high-dimensional multiscale dynamical systems, Multiscale Modeling and Simulation, 9(1) (2011), 449–485.
[32] D. Maljovec, B. Wang, A. Kupresanin, G. Johannesson, V. Pascucci and P. Bremer. Adaptive Sampling with Topological Scores, Int. J. Uncertainty Quantification, (accepted), 2012.
[33] M. Galass, J. Davies, J. Theiler, B. Gough, G. Jungman, P. Alken, M. Booth and F. Rossi, GNU Scientific Library Reference Manual, 2009.
[34] R. Udawalpola and M. Berggren, Optimization of an acoustic horn with respect to efficiency and directivity, Int. J. Numer. Methods Eng., 73(11) (2008), 1571–1606.
[35] R. Udawalpola, E. Wadbro and M. Berggren, Optimization of a variable mouth acoustic horn, Internat. J. Numer. Methods Eng., (2011), 591–606.
[36] E. Wadbro, R. Udawalpola and M. Berggren, Shape and topology optimization of an acoustic horn–lens combination, J. Comput. Appl. Math., 234 (2010), 1781–1787.
[37] X. Hu, G. Lin, T. Y. Hou and P. Yan, An adaptive ANOVA-based data-driven stochastic method for elliptic PDE with random coefficients, Technical Report, Applied and Computational Mathematics, California Institute of Technology, January 11, 2012.
[38] F. Hecht, O. Pironneau, J. Morice, A. L. Hyaric and K. Ohtsuka, FreeFEM++ Manual.
[39] J. Sherman and W.J. Morrison, Adjustment of an Inverse Matrix Corresponding to Changes in the Elements of a Given Column or a Given Row of the Original Matrix, Annals Math. Statist., 20 (1949), 620–624.