PDE/PDF-INFORMED ADAPTIVE SAMPLING FOR EFFICIENT NON-INTRUSIVE SURROGATE MODELLING

Y. VAN HALDER [*], B. SANDERSE [†], AND B. KOREN [†]

Abstract. A novel refinement measure for non-intrusive surrogate modelling of partial differential equations (PDEs) with uncertain parameters is proposed. Our approach uses an empirical interpolation procedure, where the proposed refinement measure is based on a PDE residual and probability density function of the uncertain parameters, and excludes parts of the PDE solution that are not used to compute the quantity of interest. The PDE residual used in the refinement measure is computed by using all the partial derivatives that enter the PDE separately. The proposed refinement measure is suited for efficient parametric surrogate construction when the underlying PDE is known, even when the parameter space is non-hypercube, and has no restrictions on the type of the discretisation method. Therefore, we are not restricted to conventional discretisation techniques, e.g., finite elements and finite volumes, and the proposed method is shown to be effective when used in combination with recently introduced neural network PDE solvers. We present several numerical examples with increasing complexity that demonstrate accuracy, efficiency and generality of the method.

Key word. PDE residual, interpolation, uncertainty quantification, non-intrusiveness

1. Introduction.

Uncertainty Quantification (UQ) has become increasingly important for complex engineering applications. Determining and quantifying the influence of parametric and model-form uncertainties is essential for a wide range of applications: from turbulent flow phenomena [1, 2], aerodynamics [3], biology [4, 5] to design optimisation [6, 7, 8].

Numerical methods in UQ are often divided in two groups; intrusive and non-intrusive. We focus on non-intrusive sampling methods, as they do not change the deterministic model and allow for the usage of black-box solvers. One commonly used non-intrusive method is *stochastic collocation* [9], which uses a black-box to sample the deterministic model several times in stochastic space, and interpolates these samples to construct a surrogate model. Commonly used sets of interpolation nodes are the Gauss nodes [9] and Clenshaw-Curtis nodes [10]. The Gauss nodes possess a high polynomial exactness, but are not nested, which makes them less attractive when the surrogate needs to be refined. On the other hand, Clenshaw-Curtis nodes are nested and are suited for accurate surrogate modelling. However, the number of samples increases exponentially with the number of dimensions of the stochastic space, i.e., the number of uncertainties in the model. This phenomenon is known as the *curse of dimensionality* and limits the applicability of stochastic collocation when the black-box is computationally expensive to sample from. As a remedy, alternatives to the tensor based stochastic collocation were introduced, e.g., Leja-node stochastic collocation [11, 12], empirical interpolation [13, 14, 15, 16], 'best' interpolation [17], and Smolyak sparse grids [18, 19]. These alternatives sample the model adaptively in order to reduce the number of samples. Furthermore, empirical interpolation enhances adaptive sampling placement by incorporating knowledge from the underlying model in terms of the Partial Differential Equation (PDE) residual, which is a measure of how well an approximation satisfies the model. Even though this results in a significant decrease in the number of samples when compared to methods that do not take the model into account, the method is still intractable for uncertainty propagation with a large number of uncertainties, as the empirical interpolation bases sample placement on the entire solution, rather than focusing on the Quantity of Interest (QoI). Furthermore, when interested in the statistical properties of the QoI, e.g., mean and variance, using only the residual as a measure for adaptive sampling placement is not efficient, as it does not utilise the Probability Density Function (PDF), which is used in the calculation of these quantities.

In this work, an empirical interpolation procedure related to [14, 16, 17] is proposed, with the main differences that: probability information is included in the sampling algorithm, fewer restrictions on the type of the PDE are imposed, and the residual is based solely on the QoI and not on the entire PDE solution. A relation between the error in the surrogate and the PDE residual is given, which justifies the residual as a refinement measure for surrogate construction. Furthermore an alternative refinement measure, which incorporates the PDF, is proposed when interested in the statistical properties of the QoI. Using both the residual and PDF as a measure for defining new sampling

---

[*]Centrum Wiskunde & Informatica (CWI), Science Park 123, 1098 XG, Amsterdam, the Netherlands
[†]Eindhoven University of Technology, P.O. Box 513, 5600 MB, Eindhoven, the Netherlands

locations, leads to accurate statistical properties of the QoI, which converge significantly faster than the procedures in [14, 16, 17]. Additionaly, it is shown that the proposed method is suited for efficient surrogate construction on complex topologies, which is a common problem in the case of dependent input uncertainties. Our method does not require a specific type of PDE discretisation, e.g., finite elements or finite volumes, and can therefore be used in combination with new state-of-the-art neural network solvers [20]. A key part of our approach is the use of the PDE residual, which is discussed later in more detail. In order to compute this residual, the black-box solver needs to give not only the solution values, but also derivatives with respect to spatial/temporal coordinates. This introduces a small degree of intrusiveness in the approach, although no changes to the model equations are necessary and our approach is therefore still referred to as a non-intrusive approach. Finally, new methods for solving PDEs [20] can be used in combination with our method without altering the black-box solver. As our approach is still considered to be non-intrusive and uses a combination of the PDE residual and PDF of the uncertain parameters as a refinement measure, we refer to the proposed method as Non-Intrusive PDE/PDF-informed Adaptive Sampling (NIPPAS).

This paper is outlined as follows: section 2 introduces the problem, section 3 introduces the new method and proves that the proposed sampling procedure is suitable for accurate surrogate construction. After introducing the proposed method, section 4 shows the individual steps of the method in more detail. Implementation details are discussed in section 5, and section 6 demonstrates efficiency and accuracy of our method when applied to several test-cases and compares the results with sparse grid interpolation and classical empirical interpolation.

2. Problem Description.

Quantifying the effects of parametric uncertainties in computational engineering typically is a three-step process [21]; the input uncertainties are characterised in terms of a Probability Density Function (PDF); the uncertainties are propagated through the model; and the outputs are post-processed, where the Quantity of Interest (QoI) is expressed in terms of its statistical properties. In the present work we focus on the propagation step, and the input distributions are assumed to be given. The underlying model is a PDE, which is assumed to be of the form

$$\sum_{l=1}^{n_l} g_l(\mathbf{z},\mathbf{x}) L_l^e(v^e;\mathbf{x}) = S(\mathbf{z},\mathbf{x}) , \quad (\mathbf{x},\mathbf{z}) \in D \times I_{\mathbf{z}} , \tag{2.1}$$

which is supplemented with proper initial and boundary conditions. In (2.1), n_l denotes the number of differential operators in the PDE, $\mathbf{z} \in I_{\mathbf{z}} \subset \mathbb{R}^d$ a d-dimensional vector containing uncertain parameters with corresponding joint PDF $\rho(\mathbf{z})$, $\mathbf{x} \in D$ a vector containing spatial and/or temporal coordinates, L_l^e are differential operators, g_l are known functions, S a source term, and $v^e(\mathbf{z},\mathbf{x})$ the exact solution of the PDE. This particular PDE-form assumes that the uncertainties enter the equation via the source term $S(\mathbf{z},\mathbf{x})$ and the functions $g_l(\mathbf{z},\mathbf{x})$ as parameters in front of differential operators and allows the definition of a non-zero residual in the random space $I_{\mathbf{z}}$. This PDE-form does not comprise all possible PDEs, but many PDEs with parametric uncertainties, e.g., isotropic diffusion equations, Newtonian Navier-Stokes equations and advection-diffusion equations, may be rewritten in this form:

$$v_t^e = z \Delta v^e , \tag{2.2}$$

$$v_t^e + (v^e \cdot \nabla) v^e = -\frac{\nabla p^e}{\rho} + z \Delta v^e , \tag{2.3}$$

$$v_t^e + z_1 v_x^e = z_2 v_{xx}^e . \tag{2.4}$$

Because the exact solution of (2.1) is often not available, a discrete solution vector $\mathbf{v}(\mathbf{z})$ is computed, e.g., via a finite-difference method or finite-volume method, which satisfies a discretised form of (2.1) for $\mathbf{z} \in I_{\mathbf{z}}$:

$$\sum_{l=1}^{n_l} G_l(\mathbf{z},X) L_l(\mathbf{v}(\mathbf{z})) = \mathbf{S}(\mathbf{z},X) , \quad X \subset D, \mathbf{v} \in \mathbb{R}^{N_{PDE}} , \tag{2.5}$$

where $L_l : \mathbb{R}^{N_{\mathrm{PDE}}} \to \mathbb{R}^{N_{\mathrm{PDE}}}$ are the discretised PDE operators, $X = (\mathbf{x}_1,...,\mathbf{x}_{N_{\mathrm{PDE}}})$ is the computational grid in space and time consisting of N_{PDE} grid points, $G_l \in \mathbb{R}^{N_{\mathrm{PDE}} \times N_{\mathrm{PDE}}}$ are diagonal matrices with diagonal entries $G_{l,ii}(\mathbf{z},X) = g_l(\mathbf{z},X_i)$,

and $\mathbf{S}(\mathbf{z},X) \in \mathbb{R}^{N_{\text{PDE}}}$ is the source term evaluated on the computational grid. The initial and boundary conditions are comprised in the source term $\mathbf{S}$. The uncertainties enter the equations via the source term and the matrices G_l, which comprises the function g_l evaluated on the computational grid X.

We are interested in the dependence of the QoI on the parameters $\mathbf{z}$. The QoI $\mathbf{u}(\mathbf{z})$ is assumed to be a set of linear combinations of the solution vector $\mathbf{v}$, i.e., $\mathbf{u}(\mathbf{z}) = Q\mathbf{v}(\mathbf{z})$, where

$$(2.6) \qquad\qquad Q : \mathbb{R}^{N_{\text{PDE}}} \to \mathbb{R}^{N_{\text{QoI}}} ,$$

is a matrix that maps the solution vector $\mathbf{v}$ to the QoI $\mathbf{u}$. This assumption allows for a suitable refinement measure for sampling, which is introduced later. By assuming linearity of Q we limit the space of possible QoIs, but this limitation is not too severe as many quantities, e.g., integral quantities and mean quantities, can be written in this form.

The goal in this work is twofold: either construct an accurate surrogate for $\mathbf{u}(\mathbf{z})$ or calculate statistical properties of the QoI with respect to the uncertain parameters $\mathbf{z}$. Calculating statistical properties is achieved by constructing a surrogate, which is used in combination with Monte-Carlo sampling to extract the statistical quantities. However if we are only interested in the statistical properties of the QoI, the surrogate does not need to be accurate everywhere, as some areas of the random space contribute little when calculating these statistical properties. Nevertheless, whether we are interested in constructing an accurate surrogate to study the dependency of the QoI on the parameters $\mathbf{z}$, or whether we are interested in the statistical properties of the QoI, a surrogate needs to be constructed. We construct a surrogate by applying a PDE-solver to (2.5) and by sampling values from the unknown $\mathbf{u}(\mathbf{z})$. We sample the QoI $\mathbf{u}(\mathbf{z}_i)$ at $N+1$ locations $\{\mathbf{z}_i\}_{i=0}^N$ in the random space $I_{\mathbf{z}}$. The QoI evaluations $\mathbf{u}(\mathbf{z}_i)$ are calculated as follows

$$(2.7) \qquad \underbrace{\sum_{l=1}^{n_l} G_l(\mathbf{z},X) L_l(\mathbf{v}(\mathbf{z})) = \mathbf{S}(\mathbf{z},X)}_{\text{solve PDE for } \mathbf{z}=\mathbf{z}_i} \Rightarrow \ \mathbf{u}(\mathbf{z}_i) = Q\mathbf{v}(\mathbf{z}_i) .$$

The PDE-solver is assumed to be a black-box, which means that we supply inputs and receive outputs, without the possibility to observe intermediate steps. After sampling, a surrogate model $\tilde{\mathbf{u}}(\mathbf{z})$ is constructed by means of polynomial interpolation on the samples $\{(\mathbf{z}_i, \mathbf{u}(\mathbf{z}_i))\}_{i=0}^N$. The interpolant is constructed individually for each element of $\mathbf{u}$, such that:

$$(2.8) \qquad \tilde{u}_j(\mathbf{z}) \approx u_j(\mathbf{z}), \ \text{ for all } \ \mathbf{z} \in I_{\mathbf{z}} , \ \text{ with } \tilde{u}_j(\mathbf{z}_i) = u_j(\mathbf{z}_i) \ i = 0,...,N ,$$

where u_j corresponds to the j-th element of the vector $\mathbf{u}$. The element-wise approximation for the entire QoI vector $\mathbf{u}$ is denoted as $\tilde{\mathbf{u}}$. Polynomial interpolation is used instead of polynomial regression as it allows for more efficient adaptive refinement and has extensive theoretical grounding. When interpolating, placing a new sample ensures that the new surrogate model is accurate in a neighbourhood around the newly added sample and has therefore immediate impact on the surrogate in the sampled area. This ensures improved accuracy near the new sample location, something which is not necessarily the case when using regression. Choosing proper sample locations $\mathbf{z}_i$ is crucial for stable and accurate interpolation and this will be the main focus of this paper.

Many UQ methods focus either on the PDF [9] or on the PDE residual [14] for adaptive sample placement. The PDF indicates which values for $\mathbf{z}$ are likely to happen and are therefore important to sample. The PDE residual however gives an indication where surrogate refinement is needed in order for the surrogate to satisfy the underlying PDE. In this paper we propose a novel strategy, in which both the importance of the PDE and PDF is taken into account in determining the sample locations. Finding a set of sample locations, which resembles the importance of both the underlying PDE and the PDF, is the goal of this paper.

3. Non-Intrusive PDE/PDF-informed Adaptive Sampling.

The locations of the interpolation samples determine the accuracy and stability of the surrogate model. We construct a set of interpolation nodes adaptively, by using knowledge from the underlying PDE as refinement measure.

Residual definition.

For this purpose we define the PDE residual, which indicates how well an approximate solution satisfies the discretised PDE in the random space I_z. An intuitive definition of the residual can be obtained by first constructing approximations $\widetilde{L_l(\mathbf{v}(\mathbf{z}))}$ in the random space based on evaluations $L_l(\mathbf{v}(\mathbf{z}_i))$, and substituting these approximations in the discretised PDE (2.5):

$$(3.1) \qquad \mathbf{R}_v(\mathbf{z}) := \sum_{l=1}^{n_l} G_l(\mathbf{z}, X) \widetilde{L_l(\mathbf{v}(\mathbf{z}))} - \mathbf{S}(\mathbf{z}, X) .$$

The quantity $\mathbf{R}_v$ indicates how well the approximations $\widetilde{L_l(\mathbf{v}(\mathbf{z}))}$ satisfy the discretised PDE in the random space $I_{\mathbf{z}}$.

Relation between residual and surrogate error.

The residual $\mathbf{R}_v(\mathbf{z})$ is a quantity that indicates the quality of a surrogate by substituting the surrogate into the discretised PDE and by calculating the error. The next theorem states a relation between the residual $\mathbf{R}_v$ and the error in the surrogate $\tilde{\mathbf{u}}$ for linear PDEs, which is used later to define a suitable refinement measure for placing new samples in the random space.

THEOREM 3.1. *Assume the following:*
- *Bounded 1D random space $I_z = [z_{lb}, z_{rb}]$.*
- *Well-posed discretised linear PDE of the form*

$$(3.2) \qquad \sum_{l=1}^{n_l} G_l(z, X) L_l(\mathbf{v}(z)) = \mathbf{S}(z, X) , \quad X \subset D, \mathbf{v} \in \mathbb{R}^{N_{PDE}} .$$

- *L_l are discretised linear differential operators, i.e., matrices satisfying:*

$$(3.3) \qquad L_l(\mathbf{v} + \mathbf{w}) = L_l \mathbf{v} + L_l \mathbf{w} .$$

- *$G_l(z, X)$ are known matrices as functions of z and X.*
- *QoI vector can be written as $\mathbf{u} = Q\mathbf{v}$, where $Q \in \mathbb{R}^{N_{QoI} \times N_{PDE}}$.*

Then the following holds:

$$(3.4) \qquad Q \left(\sum_{l=1}^{n_l} G_l(z, X) L_l \right)^{-1} (\mathbf{R}_v(z)) = \tilde{\mathbf{u}}(z) - \mathbf{u}(z) ,$$

where $\tilde{\mathbf{u}}$ is the interpolant in the random space I_z for each element of the vector $\mathbf{u}$, based on the samples $\{(z_i, \mathbf{u}(z_i)\}_{i=0}^N$.

Proof. Interpolation in 1D is unique and for convenience we choose the Lagrange basis $\ell_i(z)$, where ℓ_i is the i-th Lagrange basis polynomial defined as:

$$(3.5) \qquad \ell_i(z) = \prod_{\substack{k=0 \\ k \neq i}}^{N} \frac{z - z_k}{z_i - z_k} .$$

The approximations $\widetilde{L_l(\mathbf{v}(z))}$ are then given by

$$(3.6) \qquad \widetilde{L_l(\mathbf{v}(z))} = \sum_{i=0}^{N} \ell_i(z) L_l(\mathbf{v}(z_i)) ,$$

which can be rewritten due to the linearity of L as

$$(3.7) \qquad \widetilde{L_l(\mathbf{v}(z))} = L_l(\sum_{i=0}^{N} \ell_i(z) \mathbf{v}(z_i)) = L_l(\tilde{\mathbf{v}}(z)) .$$

4

Substitution into (3.1) gives

$$(3.8) \qquad \mathbf{R}_v(z) = \sum_{l=1}^{n_l} G_l(z,X) L_l(\tilde{\mathbf{v}}(z)) - \mathbf{S}(\mathbf{z},X) \ .$$

Subtracting equation (3.2), results in

$$(3.9) \qquad \mathbf{R}_v(z) = \sum_{l=1}^{n_l} G_l(z,X) L_l(\tilde{\mathbf{v}}(z)) - \underbrace{\sum_{l=1}^{n_l} G_l(z,X) L_l(\mathbf{v}(z))}_{\mathbf{S}(\mathbf{z},X)} \ .$$

This can be rewritten by using linearity of L as

$$(3.10) \qquad \mathbf{R}_v(z) = \sum_{l=1}^{n_l} G_l(z,X) L_l(\tilde{\mathbf{v}}(z) - \mathbf{v}(z)) \ .$$

Using well-posedness of the underlying discretised PDE, we can write

$$(3.11) \qquad \left(\sum_{l=1}^{n_l} G_l(z,X) L_l \right)^{-1} (\mathbf{R}_v(z)) = \tilde{\mathbf{v}}(z) - \mathbf{v}(z) \ ,$$

which can be multiplied by the matrix Q to obtain the desired result

$$(3.12) \qquad Q \left(\sum_{l=1}^{n_l} G_l(z,X) L_l \right)^{-1} (\mathbf{R}_v(z)) = \tilde{\mathbf{u}}(z) - \mathbf{u}(z) \ ,$$

$$\square$$

Theorem 3.1 states a relation between the residual and the error in the QoI surrogate. From this relation we can derive the error bound stated in the following corollary.

COROLLARY 3.2. *Assume the following:*

- *The conditions of theorem 1 are satisfied, such that* (3.4) *holds.*

Then the error $\|\tilde{\mathbf{u}}(z) - \mathbf{u}(z)\|_2$ *satisfies*

$$(3.13) \qquad \|\tilde{\mathbf{u}}(z) - \mathbf{u}(z)\|_2 \leq \|Q\|_2 \left\| \left(\sum_{l=1}^{n_l} G_l(z,X) L_l \right)^{-1} \right\|_2 \|\mathbf{R}_v(z)\|_2 \ ,$$

where $\| \cdot \|_2$ *is the vector 2-norm or its induced matrix norm.*

Corollary 3.1 states an upper bound for the error in the surrogate in terms of the residual and the discretised differential operator, which gives an indication where the surrogate $\tilde{\mathbf{u}}$ deviates significantly from the exact solution. Equation (3.13) holds for any induced matrix norm. In this paper we adopt the 2-norm.

Refinement measure for adaptive sampling.

The goal is to sample the QoI in the random space $I_\mathbf{z}$ such that we can construct an accurate surrogate model for the QoI by means of polynomial interpolation. Theorem 1 shows that a suitable refinement measure for a linear underlying PDE, is given by

$$(3.14) \qquad \mathbf{R}^*(\mathbf{z}) = Q \left(\sum_{l=1}^{n_l} G_l(\mathbf{z},X) L_l \right)^{-1} \mathbf{R}_v(\mathbf{z}) \ ,$$

If samples are placed such that $\mathbf{R}^*$ converges to zero, then the error in the surrogate also converges to zero. Using polynomial interpolation for the approximations $\widetilde{L_l(\mathbf{v}(\mathbf{z}))}$ in $\mathbf{R}_\nu$ ensures that $\mathbf{R}_\nu$ is close to zero in the neighbourhood of the interpolation samples. The quantity $\left(\sum_{l=1}^{n_l} G_l(\mathbf{z},X)L_l\right)^{-1}$ is a function of $\mathbf{z}$, but does not depend on the choice of interpolation samples and therefore acts as a scaling function for the residual $\mathbf{R}_\nu$. This justifies using a greedy approach for placing new samples as follows:

$$\text{(3.15)} \qquad \mathbf{z}_{\text{new}} = \arg\max_{\mathbf{z}\in I_{\mathbf{z}}} \|\mathbf{R}^*(\mathbf{z})\|_2 \ .$$

The construction of $\widetilde{L_l(\mathbf{v}(\mathbf{z}))}$ in $\mathbf{R}_\nu$ is a crucial step in the NIPPAS method and is therefore discussed in section 4 in detail. However, when using $\mathbf{R}^*$ as a refinement measure, there are two issues:

- The term $\left(\sum_{l=1}^{n_l} G_l(\mathbf{z},X)L_l\right)^{-1}$ is a-priori unknown and expensive to compute.
- Equation (3.14) holds for linear PDEs only.

The inverse of the full discretisation matrix is unknown in general and cannot be used for adaptive sample placement, as it is expensive to compute as a function of $\mathbf{z}$. However, as this term only acts as a scaling function, omitting this term may result in a compact refinement measure for the QoI, which is then computed as follows:

$$\text{(3.16)} \qquad \mathbf{R}(\mathbf{z}) := Q\mathbf{R}_\nu(\mathbf{z}) \ .$$

Notice that in comparison to $\mathbf{R}^*$, $\mathbf{R}$ can be computed for both linear and non-linear PDEs. Intuitively, we might expect that the refinement measure $\mathbf{R}(\mathbf{z})$ produces accurate and stable interpolants in combination with the greedy sample placement, because large errors and/or instabilities in the surrogate would lead to large errors in the residual, which then triggers new sample placement due to the greedy approach. A commonly used quantity that is used to indicate the quality of a set of sample locations is the Lebesgue constant [22]. We note that our proposed greedy approach does not necessarily result in sample locations with an optimal Lebesgue constant, but effectively uses the model information to sample regions of interest, similar to other approaches [11, 14, 17, 23].

To summarise, $\mathbf{R}$ is our proposed refinement measure, as it is less expensive to compute and more generally applicable when compared to $\mathbf{R}^*$. A numerical comparison for both refinement measures is given in section 6 for a case where $\mathbf{R}^*$ can still be computed. Furthermore, the effectiveness of using $\mathbf{R}$ as a refinement measure for the case of non-linear PDEs is demonstrated numerically in section 6.

Incorporating PDF in refinement measure.
Although using the residual $\mathbf{R}(\mathbf{z})$ as a refinement measure for adaptive sampling may result in stable and accurate surrogate models, the sampling procedure may put too much resources in areas that contribute little to the statistical quantities. Such areas are of little interest when we compute statistical quantities like the mean

$$\text{(3.17)} \qquad \mathbb{E}[\mathbf{u}] := \int_{I_{\mathbf{z}}} \rho(\mathbf{z})\mathbf{u}(\mathbf{z})\mathrm{d}\mathbf{z} \ .$$

These statistical quantities are calculated by weighing the QoI with the PDF and integrating the resulting quantity over the parameter space $I_{\mathbf{z}}$. Areas of $I_{\mathbf{z}}$ where both the PDF and QoI are low, contribute little to the integral in (3.17) and therefore an alternative refinement measure is proposed, which is especially suited when one is interested in accurate statistical quantities rather than an accurate surrogate model. The proposed refinement measure is based on the following theorem:

THEOREM 3.3. *Assume the following:*
- *The conditions of theorem 1 are satisfied, such that* (3.4) *holds.*
- $\mathbb{E}[\tilde{\mathbf{u}}]$ *and* $\mathbb{E}[\mathbf{u}]$ *are finite.*

Then the following holds:

$$\text{(3.18)} \qquad \|\mathbb{E}[\tilde{\mathbf{u}}-\mathbf{u}]\|_2 \leq (z_{rb}-z_{lb}) \sup_{z\in[z_{lb},z_{rb}]} s(z)\rho(z)\|Q\|_2\|\mathbf{R}_\nu(z)\|_2 \ ,$$

where

$$(3.19) \qquad s(z) = \left\| \left(\sum_{l=1}^{n_l} G_l(z,X) L_l \right)^{-1} \right\|_2 .$$

The proof of the theorem follows the proof of theorem 3.1 and bounds the integral by multiplying the domain length $(z_{\mathrm{rb}} - z_{\mathrm{lb}})$ with a bound for the integrand. The full proof is not shown to avoid repetitiveness.

Theorem 3.3 states a relation between the residual and the error in the mean of the QoI. The theorem shows that a proper refinement measure again includes the inverse of the full differentiation matrix. As stated before, this value is unknown in general as a function of $\mathbf{z}$ and is expensive to sample. Therefore the proposed refinement measure for calculating statistical quantities is given by

$$(3.20) \qquad \mathbf{R}_\rho(\mathbf{z}) := \rho(\mathbf{z}) Q \mathbf{R}_v(\mathbf{z}) = \rho(\mathbf{z}) \mathbf{R}(\mathbf{z}) ,$$

and new samples are placed using the greedy approach

$$(3.21) \qquad \mathbf{z}_{\mathrm{new}} = \arg\max_{\mathbf{z} \in I_{\mathbf{z}}} \| \mathbf{R}_\rho(\mathbf{z}) \|_2 .$$

Refinement measure (3.20) will not place new samples in parts of the random space where the PDF is zero, but is still able to place samples at "rare events" in the random space, i.e., parts where the PDF is small but where the PDE residual is large.

A comparison of the refinement measures (3.16) and (3.20) for convergence of both surrogate construction and statistical quantity calculation are given in section 6.

Overview of method.
Residual definitions (3.16) and (3.20) are used in the NIPPAS method to adaptively refine a surrogate model $\tilde{\mathbf{u}}(\mathbf{z})$. In detail, the NIPPAS method comprises the following steps:
1. initial sample placement
2. refinement loop:
    (a) compute residual $\mathbf{R}(\mathbf{z})$ or $\mathbf{R}_\rho(\mathbf{z})$
    (b) check stopping criterion
    (c) find $\mathbf{z}_{\mathrm{new}} = \arg\max_{\mathbf{z} \in I_{\mathbf{z}}} \| \mathbf{R}(\mathbf{z}) \|_2$ or $\| \mathbf{R}_\rho(\mathbf{z}) \|_2$
    (d) sample model at $\mathbf{z}_{\mathrm{new}}$
3. interpolate resulting samples

A schematic representation of the methodology is shown in figure 1. The choice of using either refinement measure (3.16) or (3.20) depends on whether the interest is in constructing an accurate surrogate or calculating accurate statistical quantities. The individual steps of the proposed surrogate model construction are discussed in detail in the next section. Before discussing the NIPPAS steps, the connection is shown between the NIPPAS method and general empirical interpolation method.

The refinement measures were derived from theory, which holds for linear PDEs. We will show in section 6 that the proposed refinement measures also work for a non-linear underlying PDE. Stable polynomial interpolating surrogates show clustering samples at the boundary of the domain [22]. Even though stability is not proven for the proposed method, it follows intuitively from using the greedy approach in combination with the residual. If samples do not cluster at the boundaries, then the approximations used to calculate the residual become unstable. These unstable interpolants produce large errors near the boundary of the domain because of the Runge-phenomenon. This results in large residual values, which leads to sample placement near the boundary of the domain. Lastly, note that the sample locations from the NIPPAS are in general scattered, which complicates constructing an interpolant on these samples. This aspect will be discussed in more detail in section 5

Connection with empirical interpolation.
Our NIPPAS procedure has similarities to the classical empirical interpolation procedure [14]. The main difference
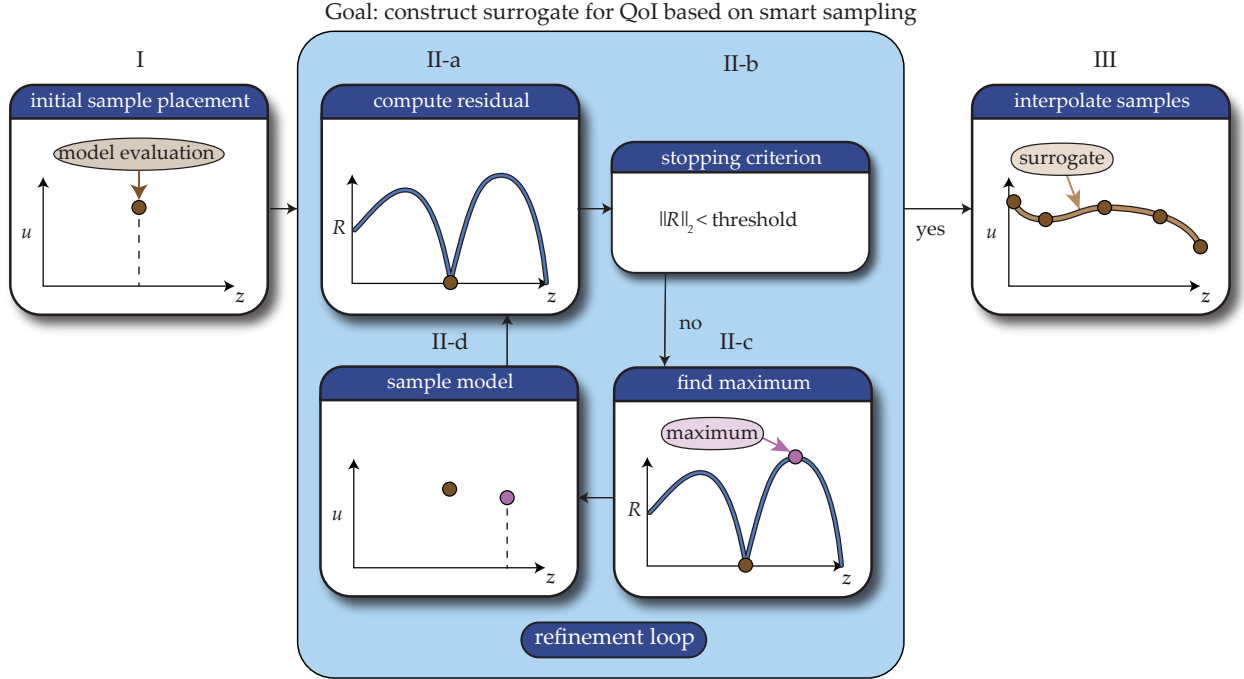
Fig. 1: *Schematic representation of the methodology.*

between general empirical interpolation and NIPPAS is the way in which the residual is constructed. In empirical interpolation the residual is based on the entire spatial/temporal surrogate $\tilde{\mathbf{v}}$, rather than focussing on the QoI $\tilde{\mathbf{u}}$. Consequently, the residual in empirical interpolation is defined as

$$(3.22) \qquad \mathbf{R}_{EI}(\mathbf{z}) = \sum_{l=1}^{n_l} \mathbf{g}_l(\mathbf{z},X) \circ L_l(\widetilde{\mathbf{v}}(\mathbf{z})) \ ,$$

which has the advantage that one only needs to approximate a single term, i.e., $\tilde{\mathbf{v}}(\mathbf{z})$, instead of constructing several approximations $\widetilde{L_l(\mathbf{v}(\mathbf{z}))}$. However, the disadvantage is that the refinement measure $\mathbf{R}_{EI}$ adds a degree of intrusiveness as it assumes that the operators L_l are known completely [15], which is not always the case and restricts the applicability.

An advantage of the NIPPAS is that it focuses the sample placement on regions where $\tilde{\mathbf{u}}$ needs to be refined, rather than where $\tilde{\mathbf{v}}$ needs to be refined. The NIPPAS therefore focuses on areas in the parameter space that are most relevant for constructing an accurate surrogate for $\mathbf{u}$, which results in faster convergence. The disadvantage is that the NIPPAS requires a small alteration of the black-box, which will be discussed in section 4.

4. NIPPAS surrogate construction.

The NIPPAS procedure when using refinement measure $\mathbf{R}$ is shown in this section; the procedure when using $\mathbf{R}_\rho$ is analogous. A schematic overview of the NIPPAS procedure is shown in figure 1.

I. Placement of one random initial sample.

The adaptive algorithm starts by performing initial sampling in the random space. Multiple initial sample configurations can be considered. It is advised to start the method with a single Monte-Carlo sample in the random space. This initialisation is used in the remainder of this paper. The initial sample location and corresponding model evaluation is denoted as $(Z_0, U_0) = (z_0, \mathbf{u}(z_0))$. More generic, a subscript i indicates the number of adaptively placed samples, i.e., $Z_i = \{\mathbf{z}_j\}_{j=0}^{j=i}$ and $U_i = \{\mathbf{u}(\mathbf{z}_j)\}_{j=0}^{j=i}$.

8

II-a. Non-intrusive computation of residual.

The refinement loop starts with computing the residual (3.16).

The residual requires the approximations $\widetilde{L_l(\mathbf{v}(\mathbf{z}))}$. These approximations can be easily constructed for a given sample set Z_i by applying an interpolation operator P (to be discussed in section 5) on the values $\{L_l(\mathbf{v}(\mathbf{z}_i))\}_{j=0}^i$. However, *the black-box solver in general only returns the solution values $\mathbf{v}(\mathbf{z}_i)$, and should be altered such that it also returns the values of the individual discretised differential operators $L_l(\mathbf{v}(\mathbf{z}_i))$. These discretised partial derivatives are used to compute the solution values within the black-box and may be output alongside the solution when sampling the black-box.* This requires only a slight alteration of the black-box, without changing the underlying PDE, and therefore keeps the methodology non-intrusive. To summarise, the approximations of the differential operator terms are given by

$$(4.1) \qquad L_l(\mathbf{v}(\mathbf{z})) \approx \widetilde{L_l(\mathbf{v}(\mathbf{z}))} = P[(Z_i, (L_l V)_i] , \;\; l = 1, ..., n_l ,$$

where

$$(4.2) \qquad (L_l V)_i = \{L_l(\mathbf{v}(\mathbf{z}_0)), ..., L_l(\mathbf{v}(\mathbf{z}_i))\} , \;\; l = 1, ..., n_l ,$$

which are the values that should be returned by the black-box solver alongside the solution values $\mathbf{v}$. After approximating each differential operator term in the random space through interpolation, the interpolants are substituted into (3.16), which returns a function in the variable $\mathbf{z}$. Notice that if the black-box solves (2.5) with negligible round-off and iteration error, then we satisfy

$$(4.3) \qquad \mathbf{R}(\mathbf{z}_i) = 0, \; i = 0, ..., i ,$$

which attains local maxima between the samples, as is shown in figure 1.

II-b. Stopping criterion based on the residual.

The refinement loop has to be terminated after a number of iterations.

For this purpose we need a stopping criterion, which reflects the quality of the surrogate model. Equation (3.13) can be used as a stopping criterion. However, (3.13) does not hold for non-linear PDEs and computing the required norms would be intractable in general. Ideally, the residual will show a similar convergence as the error in the surrogate. If this is the case, the magnitude of the residual is not only useful to adaptively sample the black-box, but it also serves as a reliable indication for the quality of the surrogate. Therefore, the refinement loop is stopped when the following criterion is met:

$$(4.4) \qquad \|\mathbf{R}(\mathbf{z})\|_2 < \varepsilon ,$$

where ε is a specified threshold.

II-c. Find the location where the residual attains maximum.

If criterion (4.4) is not met, the surrogate is refined by placing an extra sample according to (3.16). In other words, we need to solve the following global optimisation problem in $I_{\mathbf{z}}$:

$$(4.5) \qquad \mathbf{z}_{\max} = \arg\max_{\mathbf{z} \in I_z} \|\mathbf{R}(\mathbf{z})\|_2 .$$

The complexity of this global optimisation problem depends on characteristics of the objective function $\mathbf{R}(\mathbf{z})$. The residual is in general not smooth and has multiple local maxima. Therefore, in order to solve (4.5), a particle swarm method from the Global optimisation toolbox in Matlab is used, which is able to solve non-smooth global optimisation problems.

The solution of the global optimisation problem (4.5) is denoted $\mathbf{z}_{\max}$ and is used in the next step to refine the surrogate.

II-d. Sample the model at the new sample location.

To refine the surrogate model, we add the new sample $\mathbf{z}_{max}$ to our current sample set Z_i:

$$(4.6) \qquad\qquad Z_{i+1} = Z_i \cup \mathbf{z}_{max} \; ,$$

$$(4.7) \qquad\qquad U_{i+1} = U_i \cup \mathbf{u}(\mathbf{z}_{max}) \; ,$$

$$(4.8) \qquad\qquad (L_l V)_{i+1} = (L_l V)_i \cup L_l(\mathbf{v}(\mathbf{z}_{max})) \; , \quad l = 1, ..., n_l \; .$$

The new sample set $(L_l V)_{i+1}$ is suited for constructing an improved approximation $\widetilde{L_l(\mathbf{v})}$, see equation (4.1), which is used in the next iteration of the refinement loop.

III Final surrogate is constructed by interpolation.

After the stopping criterion (4.4) is met, the refinement loop terminates and we end up with a sample set Z_n and an evaluation set U_n. In order to construct the final surrogate, we apply the interpolation operator P on the final sample set, leading to

$$(4.9) \qquad\qquad \widetilde{\mathbf{u}}(\mathbf{z}) = P[(Z_n, U_n)] \; ,$$

where $\widetilde{\mathbf{u}}$ is expected to be an accurate approximation to $\mathbf{u}$. If wanted, statistical quantities like (3.17) can be computed by using this surrogate to compute the desired integrals.

5. Implementation details.

In this section some implementation details are discussed.

Surrogate modelling by polynomial interpolation for scalar QoIs.

Interpolation for a scalar QoI is discussed here. Interpolation for a vector QoI is achieved by applying the interpolation operator to each element of the QoI vector individually.

Assume we have a sample set $Z = \{\mathbf{z}_i\}_{i=0}^N$ and corresponding QoI values $U = \{u(\mathbf{z}_i)\}_{i=0}^N$. As mentioned in section 4, we denote by P the interpolation operator which acts on the set (Z, U). Polynomial interpolation aims to find a polynomial $\tilde{u}$, such that:

$$(5.1) \qquad\qquad \tilde{u}(\mathbf{z}_i) = P[(Z, U)](\mathbf{z}_i) = u(\mathbf{z}_i) \; .$$

The polynomial $\tilde{u}$ serves as a surrogate for $u(\mathbf{z})$. Finding $\tilde{u}$ is straightforward in a 1D random space, but interpolation in multi-dimensional random spaces is not unique, and the result is influenced by the choice of interpolation basis. Additionally, the sample locations from our adaptive sampling are scattered, which further complicates the interpolation procedure. In order to make the interpolation basis unique, graded lexicographic ordered Chebyshev polynomials $\{\phi_i(\mathbf{z})\}_{i=0}^N$ are used, which are known for their stability when using them for interpolation [22]. The interpolant is found by solving a Vandermonde system [9]:

$$(5.2) \qquad \underbrace{\begin{pmatrix} \phi_0(\mathbf{z_0}) & \dots & \phi_N(\mathbf{z_0}) \\ \vdots & & \vdots \\ \phi_0(\mathbf{z_N}) & \dots & \phi_N(\mathbf{z_N}) \end{pmatrix}}_{A} \begin{pmatrix} c_0 \\ \vdots \\ c_N \end{pmatrix} = \begin{pmatrix} u(\mathbf{z_0}) \\ \vdots \\ u(\mathbf{z_N}) \end{pmatrix} \; ,$$

which results in the interpolant

$$(5.3) \qquad\qquad \tilde{u}(\mathbf{z}) = \sum_{i=0}^N c_i \phi_i(\mathbf{z}) \; .$$

The Vandermonde matrix A can become ill-conditioned when increasing the number of samples [24], which makes solving (5.2) difficult. Nevertheless, other approaches that circumvent solving (5.2) [25, 26, 27] do not have the

flexibility to reuse the inverse computations of previous iterations for solving the larger system at the current iteration. This leads to a significant increase in computational expense as the adaptive sampling placement is done iteratively. Therefore, Vandermonde interpolation is used in the NIPPAS method, and a robust procedure for solving the possibly ill-conditioned system (5.2) should be used.

Several methods exist for solving such ill-conditioned systems of equations: regularisation [28], singular value decomposition [29], pivoted LU-factorisation [30], and pseudo-inversion [31]. In practice, computing the pseudo-inverse is often not advised due to its computational cost, which is $O(N^3)$. However, our adaptive sampling algorithm requires the solution of multiple linear systems with the same Vandermonde matrix A at each iteration, which makes the use of a pseudo-inverse beneficial. Nevertheless, computing the full pseudo-inverse at each iteration would be inefficient and therefore Greville's algorithm [32] is used to update the pseudo-inverse after adding a new row/column to A. Consequently, interpolating polynomials can be constructed without solving the full linear system (5.2) and without computing the full pseudo-inverse each time the interpolation operator P is applied. The implementation of Greville's algorithm is discussed in more detail in the next subsection.

Rank-one update of pseudo-inverse.
As mentioned before, when adding a new sample to the sample set, an extra row and column are appended to the existing Vandermonde matrix (5.2). Therefore, a new pseudo-inverse needs to be determined for this new Vandermonde matrix [32]. Instead of calculating the new entire inverse, Greville's algorithm is used to iteratively compute the pseudo-inverse of a given matrix A.

Assume we have a matrix $A \in \mathbb{R}^{N \times N}$ and its corresponding pseudo-inverse G. When a column is added to A, i.e.,

$$(5.4) \qquad A' = [A\ a] \, ,$$

then Greville's algorithm computes the pseudo-inverse of the new matrix A' recursively. In more detail, the new pseudo-inverse G' of A' is given by [33]

$$(5.5) \qquad G' = \begin{pmatrix} G - db^T \\ b^T \end{pmatrix} \, ,$$

where

$$(5.6) \qquad a^{(1)} = AGa \, , \ \ a^{(2)} = a - a^{(1)} \, ,$$

$$(5.7) \qquad d = Ga \, ,$$

$$(5.8) \qquad b^T = \begin{cases} \dfrac{\left(a^{(2)}\right)^H}{\left(a^{(2)}\right)^H a^{(2)}} \, , & \text{if } a^{(2)} \neq \mathbf{0} \, , \\ (1 + d^T d)^{-1} d^T G \, , & \text{if } a^{(2)} = \mathbf{0} \, , \end{cases}$$

where the H denotes the conjugate transpose. Notice that our refinement loop adds both a row and a column to the Vandermonde matrix (5.2), each time a new sample is added. Therefore the procedure described above has to be performed twice, first adding a column $A' = [A\ c]$, then adding a column r^T to the transpose of the resulting matrix, i.e., $(A'')^T = ((A')^T\ r^T)$.

A recursive algorithm for calculating the pseudo-inverse is necessary to reduce the algorithmic complexity of the adaptive sampling procedure. For an $N \times N$ matrix, Greville's algorithm performs an inverse update with complexity $O(8N^2)$, while computing the full inverse directly has complexity $O(N^3)$ [32] and becomes infeasible quickly in high dimensional spaces $I_{\mathbf{z}}$, where the number of samples required for constructing an accurate surrogate is high.

Algorithmic complexity scales well for high dimensions.
The algorithmic complexity gives an estimate on how the computational effort scales with the number of samples and the dimension of the random space.

The computational complexity is determined by the complexity of the individual parts; updating pseudo-inverse, interpolation, and particle swarm optimisation. First we discuss the algorithmic complexity of a single iteration in the refinement loop.

Updating the pseudo-inverse is achieved by using Greville's algorithm. Assume a row and column are added to an existing pseudo-inverse of dimensions $i \times i$. Using Greville's algorithm, this can be done in $O(8i^2)$ operations [32] (discussed in section 5).

The interpolation procedure is fairly cheap when the pseudo-inverse is available. In detail, when interpolating on i sample locations, solving (5.2) only requires a matrix-vector multiplication of $O(i^2)$ operations.

The complexity of the particle swarm optimisation is determined by the number of particles used and the maximum number of iterations. In our case we use a particle swarm of N_p particles and a maximum of N_{it} iterations. Typical values for N_p and N_{it} are $100 \dim(I_{\mathbf{z}})$ and $200 \dim(I_{\mathbf{z}})$, respectively. In the worst case, the number of maximum iterations is reached and we have to evaluate the residual at $N_p N_{it}$ locations, without the guarantee to have found an optimum. Moreover, if we have i samples, then the residual is a polynomial of degree $i-1$ and can be evaluated with Horner's [24] method in $O(i)$ operations. This results in a total cost of the particle swarm optimisation for a single iteration of the surrogate construction of $O(i \dim(I_{\mathbf{z}})^2 n_l)$, where n_l is the number of terms in the PDE (2.5).

The total cost of each iteration in the refinement loop is now given by:

$$(5.9) \qquad O(\underbrace{i^2}_{\text{Greville}} + \underbrace{i^2}_{\text{interpolation}} + \underbrace{i \dim(I_{\mathbf{z}})^2 n_l}_{\text{particle swarm optimisation}}) \, .$$

Hence, if we run the refinement loop N times, a conservative upper bound for the total algorithm complexity becomes

$$(5.10) \qquad O(N^3 + N^2 \dim(I_{\mathbf{z}})^2 n_l) \, .$$

Notice that the complexity depends quadratically on the dimension of the random space. However, the number of samples necessary for achieving a specified accuracy in higher-dimensional random space will generally increase with the number of dimensions, and computational cost will therefore increase faster than quadratic with the dimension of the random space.

6. Results.

In this section we present multiple examples that illustrate the efficiency of our method. In order to study and compare convergence properties, two error measures are defined, one for the error in the statistical quantities, i.e., mean, variance, etc., and one for the error in the surrogate.

The error in the k-th statistical moment is the 2-norm of the difference vector between the exact and the approximate statistical moment as computed in (3.17):

$$(6.1) \qquad e_\rho^{(k)} := \| \mathbb{E}[\mathbf{u}^k - \tilde{\mathbf{u}}^k] \|_2 = \left\| \int_{I_{\mathbf{z}}} \rho(\mathbf{z})(\mathbf{u}^k(\mathbf{z}) - \tilde{\mathbf{u}}^k(\mathbf{z})) d\mathbf{z} \right\|_2 \, ,$$

where the integral is calculated using numerical integration with negligible error, i.e., a tensor based Gauss quadrature rule with 100 nodes in each direction. The uncertainties are assumed to be uniformly distributed, $\rho(\mathbf{z}) = 1$, unless stated otherwise.

Secondly, the error in the surrogate is computed as follows:

$$(6.2) \qquad e := \frac{1}{N_{MC}} \sum_{i=1}^{N_{MC}} \| \mathbf{u}(\mathbf{z}_i) - \tilde{\mathbf{u}}(\mathbf{z}_i) \|_2 \, ,$$

where the sum is taken over a large number of uniform Monte-Carlo samples (N_{MC}) in the random space.

The first two test-cases consider a steady and unsteady advection-diffusion equation, respectively. They demonstrate the difference between our approach and conventional empirical interpolation, effect of discretisation on error convergence, difference between refinement measures (3.16) and (3.20), and applicability to non-hypercube domains. The third and last test case considers the non-linear shallow water equations and demonstrates how our method can be used in combination with a new state-of-the-art neural network based PDE solver.

Steady-state advection-diffusion equation.

In this part we consider a test-case such that the assumptions in theorem 1 hold and we study the following:

- Comparison of refinement measure (3.16) and (3.14).
- Asymptotic sample distribution.
- Comparison between NIPPAS and conventional empirical interpolation.

The underlying PDE is the dimensionless steady state advection-diffusion problem given by:

$$(6.3) \qquad Re(z)v_x - v_{xx} = 0 \; , \; v(0,z) = 0, \; v(1,z) = 1, \; x \in [0,1] \; ,$$

where $Re(z)$ is the Reynolds number, which is assumed to be uncertain and a function of z. The equations are discretised using a finite-difference approach on an equidistant grid with a resolution of Δx with N_{PDE} grid points. The solution vector on the computational grid $\mathbf{v}(z) \approx v(\mathbf{x})$, with $x_i = i\Delta x$ for $i = 1,...,N_{PDE}$, is obtained by solving the following linear system:

$$(6.4) \qquad Re(z)L_1\mathbf{v} - L_2\mathbf{v} = \mathbf{S}(z) \; ,$$

where

$$(6.5) \qquad L_1 = \frac{1}{2\Delta x}\begin{pmatrix} 0 & 1 & & & \\ -1 & 0 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 0 & 1 \\ & & 0 & -1 & 0 \end{pmatrix}, \; L_2 = \frac{1}{\Delta x^2}\begin{pmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & 0 & 1 & -2 \end{pmatrix}$$

$$(6.6) \qquad \mathbf{S}(z) = \begin{pmatrix} 0 \\ \vdots \\ \vdots \\ 0 \\ -\frac{Re(z)}{2\Delta x} - \frac{1}{\Delta x^2} \end{pmatrix} ,$$

where the boundary conditions enter the discretised equation via the vector $\mathbf{S}$. Notice that the discretised PDE satisfies the assumptions of theorem 1. Example solutions for different Reynolds numbers are shown in figure 2.
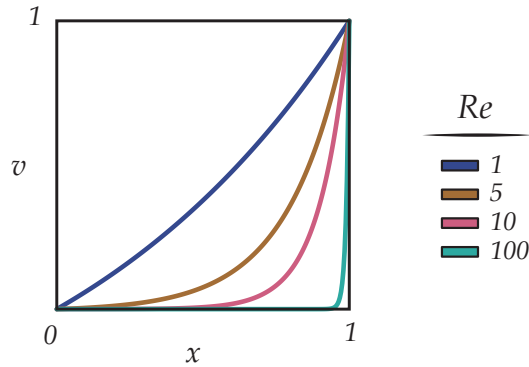

Fig. 2: *Example solutions for different Reynolds number. The solution are computed on a computational grid with $\Delta x = 10^{-3}$.*

In order for the discretisation to produce stable results, the cell Reynolds number $Re_{\Delta x} = Re\Delta x$ should satisfy $Re_{\Delta x} < 2$, which is satisfied by picking Δx sufficiently small, which is $\Delta x = 10^{-3}$ in our case ($N_{PDE} - 1000$).

13

There is no significant difference in convergence between refinement measures $\mathbf{R}$ and $\mathbf{R}^*$

The difference between refinement measures $\mathbf{R}$ and $\mathbf{R}^*$ is the incorporation of the scaling term $\left(\sum_{l=1}^{n_l} G_l(z,X)L_l\right)^{-1}$ in $\mathbf{R}^*$. This scaling term is in general expensive to compute as a function of z and is therefore often infeasible to incorporate in the refinement measure. However, for this simple test-case we can compute this scaling term as a function of z and are able to study its effect on sample placement. In order to compare the two refinement measures $\mathbf{R}$ (3.16) and $\mathbf{R}^*$ (3.14), three different functional forms for $Re(z)$ are used, where z is uniformly distributed between $[0,1]$. The three different functional forms are given by:

$$(6.7) \qquad\qquad\qquad Re_1(z) = 99z + 1 \; ,$$

$$(6.8) \qquad\qquad\qquad Re_2(z) = 10^{2z} \; ,$$

$$(6.9) \qquad\qquad\qquad Re_2(z) = 10^{-2(z-1)} \; ,$$

and are shown in figure 3. All three functions range from 1 to 100 for $z \in [0,1]$, but have completely different shapes, which influence the scaling function and therefore the sample locations when using refinement measure $\mathbf{R}^*$.

A comparison between the two refinement measures $\mathbf{R}$ and $\mathbf{R}^*$ is made by choosing $Q = I$, i.e., we are interested in the entire solution $\mathbf{u} = \mathbf{v}$ as a function of z. The adaptive sample placement starts by placing a single uniformly distributed sample in the random space. The solution values and derivative values $\mathbf{v}_x := L_1\mathbf{v}$ and $\mathbf{v}_{xx} := L_2\mathbf{v}$ are sampled at the sample location. These values are used to construct the interpolants which are required for computing $\mathbf{R}_v$. The convergence of the error in the surrogate $\tilde{\mathbf{v}}(z)$ for both refinement measures and the different Re_i is shown in figure 3.
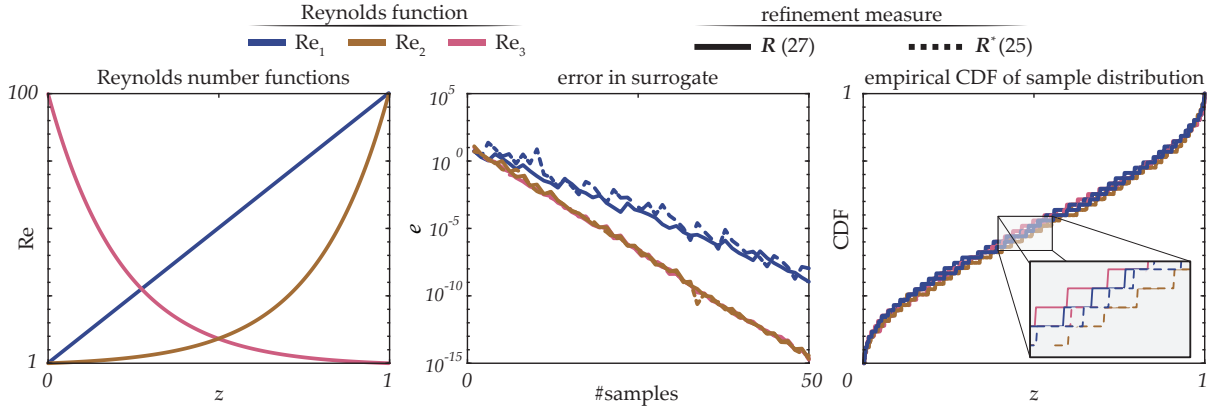

Fig. 3: *The results for the steady-state advection diffusion equation for the two refinement measures $\mathbf{R}$ and $\mathbf{R}^*$ and three different Reynolds functions in the random space. The solutions are computed using $\Delta x = 10^{-3}$.*

The error in the surrogate converges exponentially fast to machine precision for both refinement measures without any significant difference. Samples are placed at similar locations for both refinement measures, which is shown in the empirical CDF plot in figure 3. Regarding stability of the interpolant, a clustering of samples occurs at the boundaries of the random space, which stimulates stable interpolation.

Refinement measure $\mathbf{R}^*$ uses all the terms that depend on z and should be optimal for sample placement, see equations (3.4) and (3.14). However, the error in the surrogate is not only determined by sample placement, but also by the polynomial interpolation procedure. Consequently the best refinement measure does not necessarily lead to the best convergence in the error. Including the scaling term in $\mathbf{R}^*$ appears to have little effect on the sample placement, at least for this simple test-case. In fact, for this test-case one could refine directly on $\|\tilde{\mathbf{u}} - \mathbf{u}\|_2$ (because $\mathbf{u}$ is explicitly known), and this gives the same results as refining based on $\mathbf{R}^*$. As there is no significant difference between the two refinement measures, we use refinement measure $\mathbf{R}$ in the remainder of this paper, as it is more generally applicable.

We compare convergence of the surrogate for the NIPPAS based on (3.16) with the conventional empirical interpolation based on (3.22). The main difference (in absence of incorporation of the PDF) is the fact that empirical interpolation places new adaptive samples based on a residual which is based on the entire solution $\mathbf{v}$, whereas the NIPPAS uses a residual based only on the QoI $\mathbf{u}$. To compare both methods, the QoI is set to $u = (\mathbf{v})_{500}$ (the solution computed at the middle of the computational grid, with $Q = (0,...,0,1,0,...,0)$), and the Reynolds number is given by $Re_1(z)$. The convergence comparison is shown in figure 4.

Fig. 4: *Convergence comparison for NIPPAS method and conventional empirical interpolation for the steady-state advection diffusion equation.*

The NIPPAS enhances the surrogate by focusing on locations that are relevant for the QoI. This leads to faster convergence for the NIPPAS when compared to conventional empirical interpolation. However, in case more solution values from the solution vector $\mathbf{v}$ would be incorporated, i.e., more non-zero entries in the columns of Q, the convergence speed-up will decrease and eventually the convergence coincides with the one from empirical interpolation when the QoI uses the entire solution vector $\mathbf{v}$.

Unsteady advection-diffusion equation.
In the previous example we applied NIPPAS to a steady-state PDE for a 1D random space in order to compare refinement measures and to compare with conventional empirical interpolation. In this section we study the following:
- The effect of the discretisation method applied to the PDE.
- The accuracy of NIPPAS for approximating statistical moments in a 2D random space.
- The construction of a surrogate model on non-hypercube domains.

Therefore, we thoroughly study the NIPPAS for an unsteady advection-diffusion equation with two parameter uncertainties. We start with a 2D hypercube random space and then gradually increase the complexity of the test-case.

The underlying PDE is the 1D advection-diffusion equation, given by

$$(6.10) \qquad v_t + z_1 v_x = z_2 v_{xx} ,$$

where the advection parameter z_1 and the diffusion parameter z_2 are assumed to be uncertain and uniformly distributed between $[0, 2\pi]$. For the problem (6.10) to be well-posed, initial and boundary conditions are required. A spectral spatial discretisation method [34] is used for solving (6.10) and the boundary conditions are taken periodic for $x \in [0, 2\pi]$ and the initial condition is given by

$$(6.11) \qquad v(x,0) = v_0(x) = \sin(x) .$$

The spectral spatial discretisation is performed on an equidistant grid $x_i = (2\pi i)/N_x$ for $i = 0,...,N_x$, where $N_x = 256$.

15

This results in a solution vector $\mathbf{v}(t)$:

$$(6.12) \qquad \mathbf{v}(t) = \begin{pmatrix} v_0(t) \\ \vdots \\ v_{N_x}(t) \end{pmatrix},$$

where $v_i(t)$ is the approximate solution at grid point x_i. Additionally, taking derivatives can be written in terms of a matrix-vector multiplication

$$(6.13) \qquad v_x \approx D_x \mathbf{v},$$

$$(6.14) \qquad v_{xx} \approx (D_x)^2 \mathbf{v},$$

where D_x is the spectral differentiation matrix [34]. As a result, the solution vector $\mathbf{v}(t)$ can be obtained by solving the semi-discrete problem

$$(6.15) \qquad \mathbf{v}_t + z_1 D_x \mathbf{v} = z_2 D_x^2 \mathbf{v},$$

which can be rewritten in the form (2.5) by discretising the time-derivative and formulating the resulting set of equations in matrix form. Notice that this results in a block-diagonal system.

A surrogate is created for the quantity $u(z_1, z_2) = v_{N_x}(1)$, which is the solution at the right boundary of the physical domain at $t = 1$.

Effect of discretisation method is small

As we use a spectral method with a fine resolution for the spatial discretisation and the problem is linear, the time discretisation error is expected to be dominant, and therefore the effect of the time discretisation method is studied. The semi-discrete problem (6.15) is solved using different time discretisation schemes, i.e., backward-Euler, Crank-Nicolson and fourth-order explicit Runge-Kutta (RK4), where we fix the time step at $\Delta t = 10^{-5}$.

To demonstrate the efficiency of the NIPPAS method, convergence is compared to a nested Smolyak grid [35] on the Clenshaw-Curtis nodes. The errors in the surrogate for both the NIPPAS method and the Smolyak solution are computed by using a Monte-Carlo reference solution based on 5000 samples. Notice that the reference solution changes, depending on the time discretisation. The reference solution for a Crank-Nicolson time discretisation is shown in figure 5.
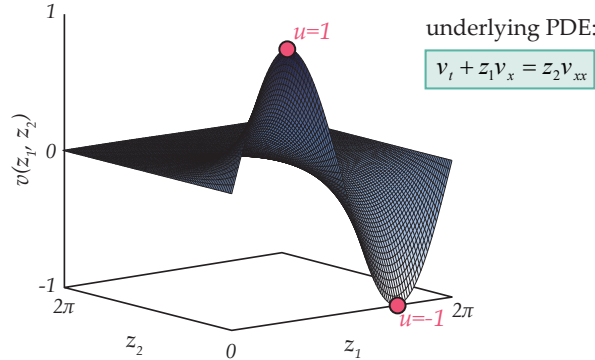

Fig. 5: *Reference solution for the advection-diffusion equation for the quantity $u(z_1, z_2) = v_{N_x}(1)$ based on 5000 samples.*

The choice of the time discretisation method affects the accuracy of the black-box solver, and changes the shape of the

surrogate slightly, as different time discretisations do not give the same QoI at the exact same location in random space. As a result, when changing the discretisation, a new reference solution has to be computed to study convergence. To further clarify, there is a difference between the exact/wanted surrogate, which is obtained by solving (2.1), and the discrete exact surrogate, obtained by solving the discretised equations (2.5). This difference is shown in the following equation:

$$(6.16) \qquad \underbrace{\|u - \tilde{u}_{N_x}\|_2}_{\text{error w.r.t. exact solution}} \quad \leq \quad \underbrace{\|u - u_{N_x}\|_2}_{\text{discretisation error}} \quad + \quad \underbrace{\|u_{N_x} - \tilde{u}_{N_x}\|_2}_{\text{error w.r.t. discrete solution}} \quad ,$$

where u is the exact solution (solution of (2.1)), u_{N_x} is the discrete solution computed using $N_x + 1$ grid points (solution of (2.5)), and $\tilde{u}_{N_x}$ is the surrogate based on the discrete solutions. For this reason, we plot both the error in the surrogate with respect to the exact solution and the discrete solution. The results are shown in figure 6.
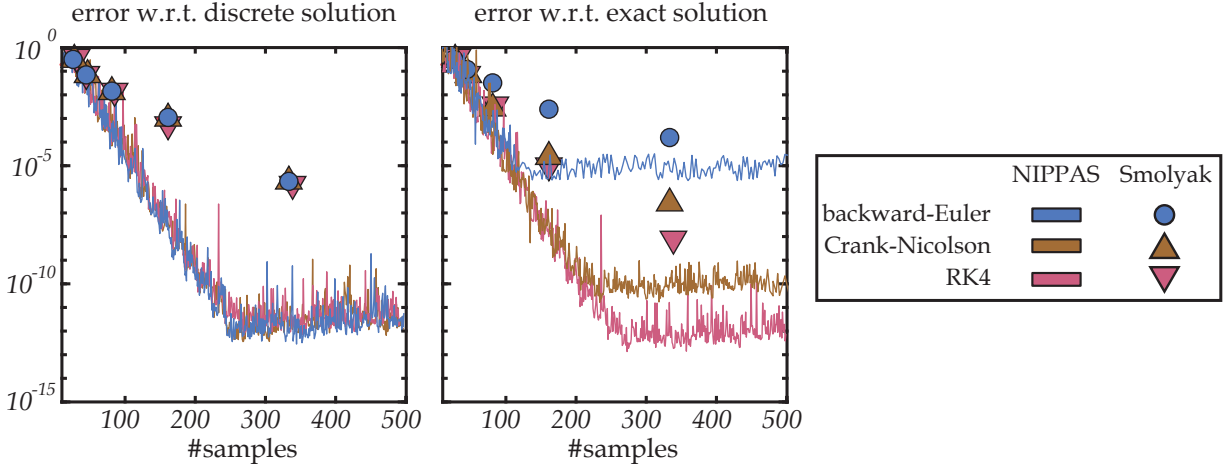

Fig. 6: *Error* (6.2) *comparison between the Smolyak sparse grid surrogate and the NIPPAS surrogate for different time discretisation methods with* $\Delta t = 10^{-5}$ *and* $N_x = 256$. *The errors are plotted with respect to both the discrete solution of* (6.10) *and the exact solution of* (6.15).

The error with respect to the discrete solution converges to zero, with a convergence rate that is significantly faster than the Smolyak sparse grid. Convergence is non-monotonic, which is common for adaptive sampling methods [11]. The error with respect to the exact solution stalls before machine precision is reached, which is due to the discretisation error. To clarify, the error with respect to the discrete solution converges to zero and equation (6.16) therefore states that the observed error after stalling is the discretisation error.

The convergence behaviour of the error with respect to the discrete solution is not dependent on the time discretisation, which indicates that the performance of our method does not depend on the underlying discretisation.

Faster convergence for statistical quantities with PDF weighing

Next, to study the effect of the two different refinement measures (3.16) and (3.20), we now assume independent β-distributed input uncertainties z_1 and z_2:

$$(6.17) \qquad \rho(z_1, z_2) = c \cdot (2\pi z_1)^{\alpha_1} (2\pi z_2)^{\alpha_2} (1 - 2\pi z_1)^{\beta_1} (1 - 2\pi z_2)^{\beta_2} ,$$

where (α_i, β_i) are parameters that characterise the PDF and the constant c is chosen such that the PDF has total probability 1 on $[0, 2\pi]^2$. Convergence behaviour is studied as a function of the shape of the underlying PDF and therefore we vary the PDF parameters as follows

$$(6.18) \qquad \alpha_i \in \{1, 2, 3, 4, 5\}, \ \beta_i \in \{1, 2, 3, 4, 5\}, \ i = 1, 2 ,$$

which results in a total of 625 PDFs with totally different shapes. The convergence statistics of the mean, variance and the entire surrogate are shown in figure 7.

2D independent β-distribution: $\rho(z_1,z_2) = c \cdot (2\pi z_1)^{\alpha_1}(2\pi z_2)^{\alpha_2}(1-2\pi z_1)^{\beta_1}(1-2\pi z_2)^{\beta_2}$, $\alpha_1,\alpha_2,\beta_1,\beta_2 \in \{1,2,3,4,5\}$
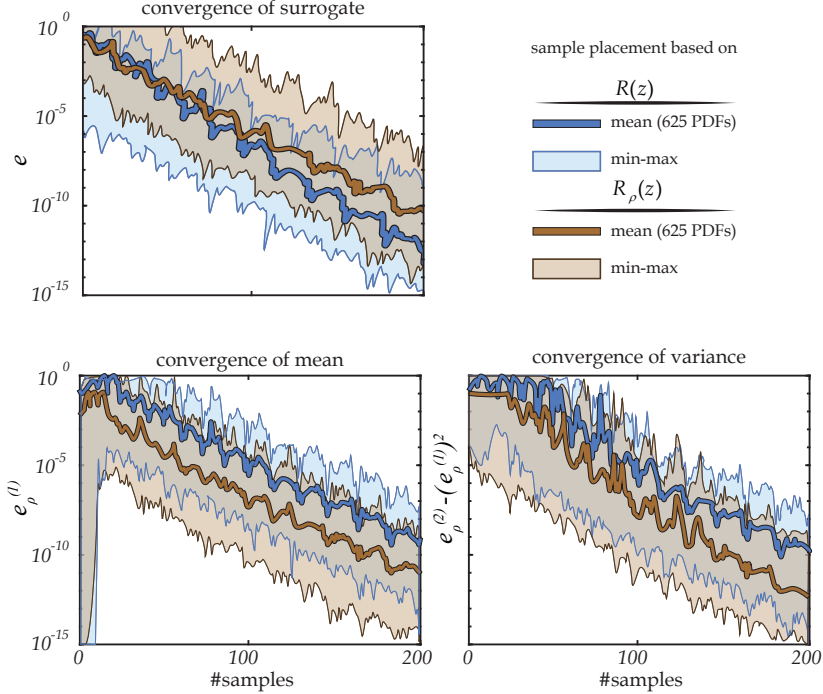

Fig. 7: *Convergence in advection-diffusion surrogate, mean and variance for the two refinement measures* (3.16) *and* (3.20). *The error in the surrogate is computed with* (6.2) *with 5000 Monte-Carlo samples. The errors in the mean and variance are calculated with* (6.1).

Figure 7 shows the average convergence behaviour taken over the 625 different PDFs. From the figure we may conclude that weighing the residual with the PDF results in slower convergence for the surrogate model, see equation (6.2), but leads to faster convergence in both the mean and the variance, which are computed using (6.1). Furthermore, the variation in convergence over all 625 different PDFs, given by the shaded area around the mean line, has similar width for both refinement measures (3.16) and (3.20). The choice of refinement measures thus depends on the type of convergence the user requires, i.e., convergence in the surrogate or convergence in the statistical quantities. Notice that the shaded area for the convergence in the mean has a minimum that starts at 0 initially, which is due to the fact that for some PDFs the exact mean is zero, and as the initial sample is placed at a location in the random space where the response is also 0, we start with an approximate mean that is equal to the exact mean. However, the sample placement does not stop immediately, as the stopping criterion (4.4) is not met.

Method shows fast convergence for non-hypercube domains

The 2D β-distribution (6.17) just discussed, comprises two independent uncertainties and therefore the space in which we construct a surrogate is a hypercube. However, when dealing with dependent uncertainties, the space in which the surrogate is constructed, is in general not a hypercube, and many existing UQ methods fail when dealing with a complex random space.

To study the performance of the NIPPAS method on more general geometries, we assume a non-hypercube domain $D \subset \mathbb{R}^d$ with associated uniform PDF $\rho(\mathbf{z})$. In order to sample the model on the domain D, we restrict the residual

(3.16) to D by multiplying it with an indicator function:

$$(6.19) \qquad R_D(\mathbf{z}) = R(\mathbf{z})\mathbb{I}_D(\mathbf{z}) \,,$$

where

$$(6.20) \qquad \mathbb{I}_D(\mathbf{z}) = \left\{ \begin{array}{ll} 1, & \text{if } z \in D \,, \\ 0, & \text{otherwise} \,. \end{array} \right.$$

After altering the residual definition slightly and placing an initial sample randomly in the non-hypercube random space, the NIPPAS method can be applied straightforwardly. Note that the basis functions are the same for the hypercube case, which are given by the Chebyshev polynomials defined on the smallest hypercube that comprises D. In order to show the efficiency of our method for non-hypercube domains, we again construct a surrogate for the response shown in figure 5, but now on more complexly shaped domains. Three different geometries with different characteristics are chosen:

- sharp corners:

$$(z_2 \geq 0) \cap (z_2 - \sqrt{3}z_1 \leq 0) \cap (\sqrt{3}z_1 + z_2 \leq 2\pi\sqrt{3}) \,,$$

- smooth boundary:

$$(z_1 - \pi)^2 + (z_2 - \pi)^2 \leq \pi^2,$$

- domain with holes:

$$(z_1, z_2) \in [0, 2\pi]^2, \text{ but}$$

$$\neg \left((z_1 - \pi)^2 + (z_2 - \pi)^2 \leq \frac{\pi^2}{4} \right) \cup$$

$$\neg \left((z_1 - \frac{\pi}{3})^2 + (z_2 - \frac{\pi}{3})^2 \leq \frac{\pi^2}{25} \right) \cup$$

$$\neg \left((z_1 - \frac{5\pi}{3})^2 + (z_2 - \frac{5\pi}{3})^2 \leq \frac{\pi^2}{9} \right) \,.$$

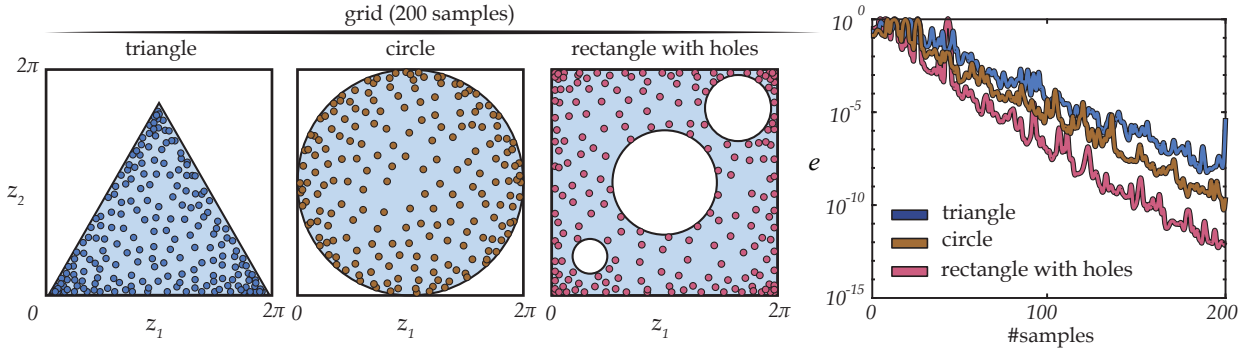The convergence results are shown in figure 8.

Fig. 8: *Convergence of surrogate for three different non-hypercube domains.*

The results show an exponential convergence behaviour for all three different geometries, but the convergence rates slightly differ, which is likely caused by the choice of basis, which is suboptimal for all domains. Furthermore,

weighing the residual with the PDF in case of a non-uniformly distributed uncertainties is again possible, but results are similar to the results shown in figure 7 and are not shown for repetitiveness.

It has to be pointed out that the choice of a lexicographically ordered Chebyshev basis is far from optimal on non-hypercube domains. Nevertheless, good convergence behaviour is still achieved.

Shallow water equations with dependent random inputs.

As last test-case, we study the performance of the NIPPAS method for a hyperbolic non-linear PDE with dependent random inputs in a non-hypercube random space. In this test-case the underlying model is non-linear and comprises three uncertain parameters which lie on a 2D triangular manifold in a 3D space and therefore combines the difficult aspects from all previous test-cases. The underlying model is a system of conservation laws, namely the 1D shallow water equations (SWEs). The SWEs describe inviscid fluid flow with a free surface, under the action of gravity, with the thickness of the fluid layer small compared to the other length scales [36]:

$$(6.21) \qquad \frac{\partial}{\partial t}\begin{pmatrix} h \\ hv \end{pmatrix} + \frac{\partial}{\partial x}\begin{pmatrix} hv \\ hv^2 + gh^2/2 \end{pmatrix} = \mathbf{0},$$

where h is the free surface height (thickness of the fluid layer), v the velocity, and g the acceleration of gravity. Reflective boundary conditions are imposed at $x = \pm 1$ and the initial condition for the system of PDEs is given by a Riemann problem:

$$(6.22) \qquad \begin{pmatrix} h \\ v \end{pmatrix}(x, t = 0) = \begin{cases} \begin{pmatrix} h_l \\ v_l \end{pmatrix}, & x \leq 0, \\ \begin{pmatrix} 1 \\ 0 \end{pmatrix}, & x > 0, \end{cases}$$

leading to a so-called dambreak problem. The three uncertain inputs are $z_1 = g$, $z_2 = h_l$ and $z_3 = v_l$, and are assumed to jointly follow a Dirichlet distribution, which is the multivariate generalisation of the 1D β-distribution [37], and is often used as a prior to a discrete categorical distribution in Bayesian statistics. The PDF of the $3D$ Dirichlet distribution with shape parameters $(\alpha_1, \alpha_2, \alpha_3)$ is given by:

$$(6.23) \qquad \rho(z_1, z_2, z_3; \alpha_1, \alpha_2, \alpha_3) = \frac{\Gamma(\sum_{i=1}^{3}\alpha_i)}{\prod_{i=1}^{3}\Gamma(\alpha_i)}\prod_{i=1}^{3}z_i^{\alpha_i - 1},$$

which is defined on the unit simplex

$$(6.24) \qquad \sum_{i=1}^{3}z_i = 1, \text{ and } z_i \geq 0 \, \forall i \,.$$

For this specific test-case we scale/translate the unit simplex to a triangle with corner points

$$(6.25) \qquad (g, h_l, v_l) = \{(12, 0.5, -0.5), (8, 1.5, -0.5), (8, 0.5, 0.5)\}\,.$$

For testing the efficiency of refinement measure (3.20), we consider the shape parameter set $(\alpha_1, \alpha_2, \alpha_3) = (5, 2, 2)$. This specific shape parameter set corresponds to an asymmetric PDF and is shown in figure 9.

The Dirichlet distribution is defined on a 2D triangular manifold in 3D space and is used for testing the NIPPAS efficiency for non-hypercube random spaces. The interpolation basis is the set of Chebyshev polynomials defined on the smallest hypercube that comprises the 2D manifold. As mentioned before, improvements on the interpolation basis are possible, but are outside the scope of this paper. Furthermore, the QoI $u(z_1, z_2, z_3)$ is the free surface height h at the left boundary $x = -1$ at $t = 1$.

The NIPPAS method samples the solution of the dambreak problem for multiple inputs in order to construct a surrogate. A commonly used method for solving the SWEs is a Riemann solver based finite-volume discretisation [38], which can determine accurate solutions efficiently. In this paper instead we demonstrate the effectiveness of the
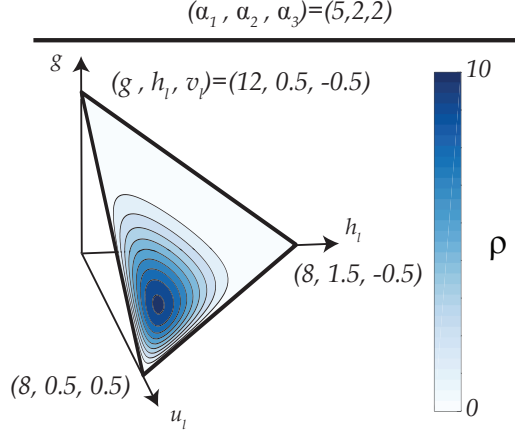
Fig. 9: *Dirichlet PDF on the scaled unit simplex for* $(\alpha_1, \alpha_2, \alpha_3) = (5, 2, 2)$.

NIPPAS method in combination with a more recently developed numerical method. Instead of using a Riemann solver, a neural network is used to solve the SWEs [20]. An advantage of using neural networks for solving PDEs is that *the solution is given in terms of a functional form, from which derivatives can be directly computed analytically*. This functional form allows us to calculate the residual, without requiring alterations to the code output. A multi-output neural network with 7 hidden layers and 40 neurons in each hidden layer is used to solve the SWEs, which is trained on a total of 10^5 collocation nodes in space and time, which are the places where the neural network tries to enforce the PDE. This particular combination of number of hidden layers and neurons showed the best results and is therefore used in this paper. The trained neural network produces a solution of the PDE. After the residual is computed and a new sample location in random space is determined, the neural network is retrained to produce a solution of the PDE for this new set of parameter values. Previously trained neural networks closest to the new sample location are used as initial starting point for training the new neural network in order to significantly speed-up the training process.
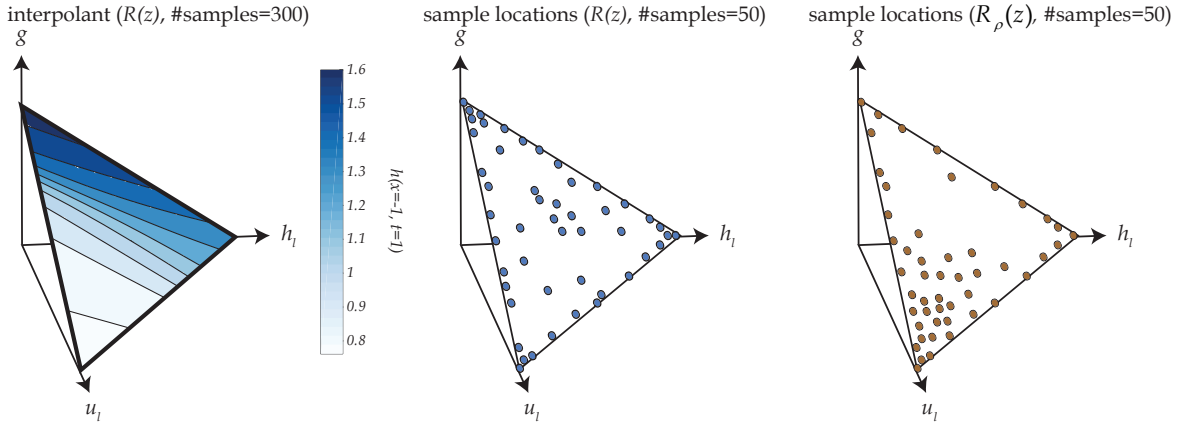

Fig. 10: *(left) Surrogate model based on 300 samples for refinement measure* (3.16). *(right) Sample locations for refinement measures* (3.16) *and* (3.20), *respectively.*

The surrogate and sample locations for both refinement measures (3.16) and (3.20) are shown in figure 10. The sample locations show clustering at the boundaries, to produce a stable interpolant. As mentioned before, if the surrogate tends to become unstable and grows at the boundaries of the random space, the residual becomes large at the boundaries as

well and causes refinement of the surrogate near the boundaries. However, at early stages of the refinement process the surrogate can still show irregular oscillations, which is due to insufficient refinement at the boundaries, but these disappear upon further refinement. When taking the PDF into account, clustering also occurs in the region of high probability, as expected. This clustering deteriorates the accuracy of the surrogate in regions of low probabilities, but leads to improved estimation of statistical quantities. Convergence comparison for refinement measures (3.16) and (3.20) are shown in figure 11.
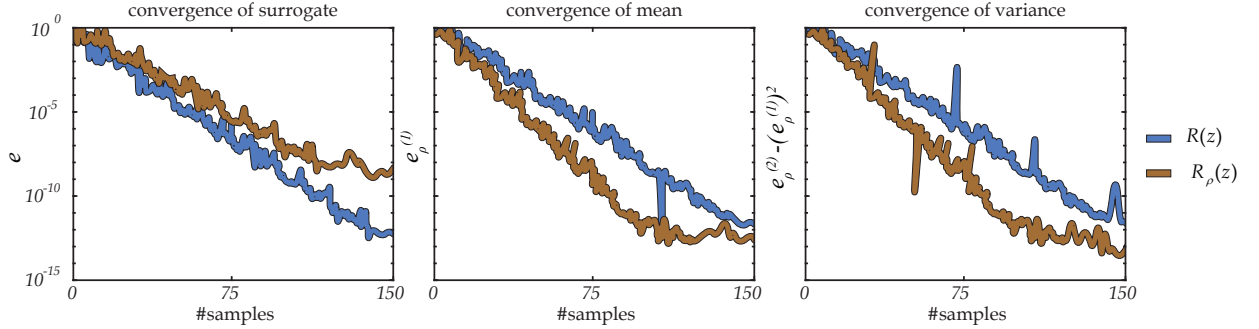

Fig. 11: *Results for the dambreak problem with random inputs. The convergence plot shows a comparison of the two different refinement measure (3.16) and (3.20). The error in the surrogate is computed with (6.2) with 5000 Monte-Carlo samples. The errors in the mean and variance are calculated with (6.1).*

The results show indeed faster convergence in statistical quantities when accounting for the PDF in the refinement measure, which was also shown in figure 7.

7. Conclusion.

In this paper we have presented a novel approach for parametric surrogate construction when the underlying PDE is known. Our technique, the Non-Intrusive PDE/PDF-informed Adaptive Sampling (NIPPAS), is suited for surrogate construction on non-hypercube parametric spaces. Non-hypercube parametric spaces occur when the underlying PDF is dependent, e.g., Dirichlet-distributed, and significantly complicate surrogate construction when using common stochastic collocation methods, e.g., sparse grid interpolation. The key ingredient of the proposed empirical interpolation procedure is refinement which is based on both the PDE-residual and the PDF. Sampling based on the PDE-residual leads to stable interpolation, even on non-hypercube domains, due to sample clustering at the boundaries of the domain. At the same time, the incorporation of the PDF in the refinement procedure samples the parametric space in regions of high probability, which ensures fast convergence of statistical quantities. This combination makes the NIPPAS method an efficient and flexible method that is applicable to a wide range of UQ problems.

The NIPPAS method has been applied to several numerical examples: 1D and 2D surrogate construction on a hypercube with linear underlying PDE, 2D surrogate construction on complex domains, and 3D surrogate construction with non-linear underlying PDE on a complex domain. In all cases, exponential convergence is obtained, leading to an accurate surrogate model fast. This surrogate model can be directly used as a tool for uncertainty quantification (for example with Monte-Carlo type methods), but it is also a great tool for the parametric solution of black-box models.

Currently, the interpolation basis for non-hypercube domains is a Chebyshev basis defined on the smallest hypercube that comprises the parametric space. Several improvements could be made, for instance by constructing a suitable basis based on the sample locations [25]. Furthermore, the global minimisation problem to be solved at each iteration does not scale well to high-dimensional random space, and alternatives for the particle swarm optimisation may be used [39].

22

References.

[1] H. Xiao, J. L. Wu, J. X. Wang, R. Sun, C. J. Roy, Quantifying and reducing model-form uncertainties in Reynolds-averaged Navier-Stokes simulations: a data-driven, physics-informed Bayesian approach, Journal of Computational Physics 324 (2016) 115–136.

[2] W. N. Edeling, R. P. Dwight, P. Cinnella, Simplex-stochastic collocation method with improved scalability, Journal of Computational Physics 310 (2016) 301–328.

[3] F. Simon, P. Guillen, P. Sagaut, D. Lucor, A gPC-based approach to uncertain transonic aerodynamics, Computer Methods in Applied Mechanics and Engineering 199 (2010) 1091–1099.

[4] K.-H. Cho, S.-Y. Shin, W. Kolch, O. Wolkenhauer, Experimental design in systems biology, based on parameter sensitivity analysis using a Monte Carlo method: a case study for the TNF-mediated NF- B signal transduction pathway, Simulation 79 (2003) 726–739.

[5] R. Abagyan, M. Totrov, Biased probability Monte Carlo conformational searches and electrostatic calculations for peptides and proteins, Journal of Molecular Biology 235 (3) (1994) 983–1002.

[6] E. M. Constantinescu, V. M. Zavala, M. Rocklin, S. Lee, M. Anitescu, A computational framework for uncertainty quantification and stochastic optimization in unit commitment with wind power generation, IEEE Transactions on Power Systems 26 (1) (2011) 431–441.

[7] J. Mateos, T. Gonzalez, D. Pardo, V. Hoel, A. Cappy, Monte Carlo simulator for the design optimization of low-noise HEMTs, IEEE Transactions on Electron Devices 47 (10) (2000) 1950–1956.

[8] M. Papadrakakis, N. D. Lagaros, Reliability-based structural optimization using neural networks and Monte Carlo simulation, Computer Methods in Applied Mechanics and Engineering 191 (32) (2002) 3491–3507.

[9] D. Xiu, Numerical Methods for Stochastic Computations: A Spectral Method Approach, Princeton University Press, 2010.

[10] J. P. Boyd, Chebyshev and Fourier Spectral Methods: Second Revised Edition, Courier Corporation, 2013.

[11] A. Narayan, J. Jakeman, Adaptive Leja sparse grid constructions for stochastic collocation and high-dimensional approximation, SIAM Journal on Scientific Computing 36 (6) (2014) A2952–A2983.

[12] L. M. M. van den Bos, B. Sanderse, W. A. A. M. Bierbooms, G. J. W. van Bussel, Bayesian model calibration with interpolating polynomials based on adaptively weighted Leja nodes, arXiv e-prints (2018) arXiv:1802.02035.

[13] M. Barrault, Y. Maday, N. C. Nguyen, A. T. Patera, An empirical interpolation method: application to efficient reduced-basis discretization of partial differential equations, Comptes Rendus Mathematique 339 (9) (2004) 667 – 672.

[14] J. S. Hesthaven, G. Rozza, B. Stamm, The Empirical Interpolation Method, in: Certified Reduced Basis Methods for Parametrized Partial Differential Equations, Springer Briefs in Mathematics, Springer, Cham, 2016, pp. 67–85.

[15] J. S. Hesthaven, B. Stamm, S. Zhang, Efficient greedy algorithms for high-dimensional parameter spaces with applications to empirical interpolation and reduced basis methods, ESAIM: Mathematical Modelling and Numerical Analysis 48 (1) (2014) 259–283.

[16] M. Ohlberger, S. Rave, Reduced Basis Methods: Success, Limitations and Future Challenges, arXiv:1511.02021 [math].

[17] N. C. Nguyen, A. T. Patera, J. Peraire, A best points interpolation method for efficient approximation of parametrized functions, International Journal for Numerical Methods in Engineering 73 (4) (2008) 521–543.

[18] F. Nobile, R. Tempone, C. Webster, A sparse grid stochastic collocation method for partial differential equations with random input data, SIAM Journal on Numerical Analysis 46 (5) (2008) 2309–2345.

[19] F. Nobile, R. Tempone, C. Webster, An Anisotropic Sparse Grid Stochastic Collocation Method for Partial Differential Equations with Random Input Data, SIAM Journal on Numerical Analysis 46 (5) (2008) 2411–2442.

[20] M. Raissi, P. Perdikaris, G. E. Karniadakis, Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations, arXiv:1711.10561 [cs, math, stat].

[21] T. Chantrasmi, Pade-legendre Method for Uncertainty Quantification with Fluid Dynamics Applications, Stanford University, 2011.

[22] L. N. Trefethen, Approximation Theory and Approximation Practice, SIAM, 2013.

[23] D. Loukrezis, U. Rmer, H. De Gersem, Numerical Comparison of Leja and Clenshaw-Curtis Dimension-Adaptive Collocation for Stochastic Parametric Electromagnetic Field Problems, arXiv:1712.07223 [cs].

[24] N. Higham, Accuracy and Stability of Numerical Algorithms, Other Titles in Applied Mathematics, Society for Industrial and Applied Mathematics, 2002.

[25] A. Narayan, D. Xiu, Stochastic collocation methods on unstructured grids in high dimensions via interpolation, SIAM Journal on Scientific Computing 34 (3) (2012) A1729–A1752.

[26] C. d. Boor, A. Ron, On multivariate polynomial interpolation, Constructive Approximation 6 (3) (1990) 287–302.

[27] T. Sauer, Polynomial interpolation of minimal degree, Numerische Mathematik 78 (1) (1997) 59–85.

[28] A. Neumaier, Solving Ill-Conditioned and Singular Linear Systems: A Tutorial on Regularization, SIAM Review 40 (3) (1998) 636–666.

[29] J. Varah, On the Numerical Solution of Ill-Conditioned Linear Systems with Applications to Ill-Posed Problems, SIAM Journal on Numerical Analysis 10 (2) (1973) 257–267.

[30] N. Higham, The Accuracy of Solutions to Triangular Systems, SIAM Journal on Numerical Analysis 26 (5) (1989) 1252–1265.

[31] A. Klinger, Approximate pseudoinverse solutions to ill-conditioned linear systems, Journal of Optimization Theory and Applications 2 (2) (1968) 117–124.

[32] S. Mohideen, V. Cherkassky, On Recursive Calculation of the Generalized Inverse of a Matrix, ACM Transactions on Mathematical Software 17 (1) (1991) 130–147.

[33] N. Shinozaki, M. Sibuya, K. Tanabe, Numerical algorithms for the Moore-Penrose inverse of a matrix: Direct methods, Annals of the Institute of Statistical Mathematics 24 (1) (1972) 193–203.

[34] C. Canuto, M. Y. Hussaini, A. M. Quarteroni, T. A. Zang, Jr, Spectral Methods in Fluid Dynamics, Springer Science & Business Media, 2012.

[35] A. Klimke, B. Wohlmuth, Algorithm 847: Spinterp: Piecewise multilinear hierarchical sparse grid interpolation in MATLAB, ACM Transactions on Mathematical Software 31 (2005) 561–579.

[36] C. B. Vreugdenhil, Numerical Methods for Shallow-Water Flow, Springer Verlag, 2013.

[37] M. Hazewinkel, Encyclopaedia of Mathematics, Springer Science & Business Media, 1990.

[38] R. J. LeVeque, Finite Volume Methods for Hyperbolic Problems, Cambridge University Press, 2002.

[39] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016.