Received: 4 July 2022 | Revised: 18 January 2023 | Accepted: 27 February 2023

DOI: 10.1002/nme.7234

RESEARCH ARTICLE

WILEY

An *hp*-adaptive multi-element stochastic collocation method for surrogate modeling with information re-use

Armin Galetzka[1] | **Dimitrios Loukrezis**[1,2,3] | **Niklas Georg**[1,2,4] | **Herbert De Gersem**[1,2] | **Ulrich Römer**[4]

[1]Institute for Accelerator Science and Electromagnetic Fields, TU Darmstadt, Darmstadt, Germany

[2]Centre for Computational Engineering, TU Darmstadt, Darmstadt, Germany

[3]Technology, Siemens AG, Munich, Germany

[4]Institut für Dynamik und Schwingungen, TU Braunschweig, Braunschweig, Germany

Correspondence
Armin Galetzka, Institute for Accelerator Science and Electromagnetic Fields, TU Darmstadt, Schlossgartenstr. 8, 64289 Darmstadt, Germany.
Email: galetzka@temf.tu-darmstadt.de

Funding information
Deutsche Forschungsgemeinschaft, Grant/Award Numbers: 264883531, 492661287

Abstract

This article introduces an *hp*-adaptive multi-element stochastic collocation method, which additionally allows to re-use existing model evaluations during either *h*- or *p*-refinement. The collocation method is based on weighted Leja nodes. After *h*-refinement, local interpolations are stabilized by adding and sorting Leja nodes on each newly created sub-element in a hierarchical manner. For *p*-refinement, the local polynomial approximations are based on total-degree or dimension-adaptive bases. The method is applied in the context of forward and inverse uncertainty quantification to handle non-smooth or strongly localized response surfaces. The performance of the proposed method is assessed in several test cases, also in comparison to competing methods.

KEYWORDS

hp-adaptivity, multi-element approximation, stochastic collocation, surrogate modeling, uncertainty quantification

1 | INTRODUCTION

Prediction and optimization with computational models is now routinely carried out in many fields of science and engineering. For instance, reliable predictions require the propagation and quantification of uncertainties[1] related to parameter calibration from noisy and limited data. The so-called non-intrusive approach is widely adopted, where computer codes are treated as black box functions from model parameters to quantities of interest (QoIs). In this way, restructuring and modifying complex software is avoided. Exemplarily, in forward uncertainty quantification (UQ),[2] also referred to as uncertainty propagation, non-intrusive methods evaluate the model at selected points of the input vector, which are generated according to the underlying probability distribution. Similarly, in inverse UQ,[3] repeated model evaluations are required to build the likelihood function at individual elements of a Markov chain.

Even when large computational resources are available, such multi-query simulations with complex models present a widely acknowledged challenge. This fact has led to significant research efforts in the area of surrogate modeling,[4,5]

for example, by means of neural networks,[6] Gaussian processes,[7] or polynomial chaos expansions (PCEs).[8] Surrogate modeling can be carried out efficiently if the response function is sufficiently smooth and if the function varies moderately over the parameter space. However, there exist many examples in engineering and mechanics where the response is discontinuous or exhibits strong variations and sharp transitions.[9] In this case, most surrogate modeling techniques become too computationally expensive or altogether unreliable, thus, Monte Carlo methods are usually preferred. Nevertheless, an efficient sampling scheme is adapted to a specific simulation task and the computed model responses cannot be re-used in a different context. This is a major bottleneck of pure sampling approaches and a strong motivation to develop flexible and robust surrogate models, which can be applied even if the model response is non-smooth.

Surrogate modeling for response surfaces with a reduced regularity typically involves some sort of local refinement. For instance, the spatially adaptive sparse grid method[10] allows for adaptive and hierarchical local refinement in subregions of the parameter domain. A combination of spatial and dimension-wise adaptivity has also been proposed.[11] Alternatively, the family of multi-element methods[12-16] presents a class of approximation techniques which have been successfully applied in the UQ context for addressing non-smooth stochastic response functions. In a multi-element method, the parameter space is decomposed into subdomains referred to as *elements*, where local polynomial basis functions are used. Here, on the one hand, one can opt for a discontinuity detection method, which first estimates a discontinuity and then splits elements accordingly. For example, the discontinuity detection can be based on spatially adaptive sparse grids and the element-wise restriction of sparse grid points is subsequently used for local polynomial approximation.[17] A difficulty to be addressed here is the missing structure of the elements after splitting, which has previously been handled by the least orthogonal interpolation method.[17] On the other hand, the element splitting can be based on hypercubes or simplices, which does not require any resolution of the discontinuity, however, applications in higher dimensions are more difficult to realize. We note that, local refinement can equally be applied in a Monte Carlo setting, for example, using stratified sampling on simplices.[18]

Here, we focus on the non-intrusive multi-element stochastic collocation method on hypercubes for forward and inverse UQ. Although several proposals for multi-element collocation exist,[16,17,19-23] a number of challenges remain to be addressed. Most notably, computational efficiency considerations require to limit the number of collocation points, which is problematic since existing points do not necessarily fit to a collocation pattern once the subdomains in the parameter space are refined. Another challenge is to efficiently implement dimension adaptivity to avoid using large tensor grids in the parameter domain. Considering the worst-case where each parameter domain is split into two parts in each refinement step, the complexity of multi-element methods scales with $\mathcal{O}\left(2^N\right)$, where the exponent refers to the number of random inputs and the base to the subdomain splitting. This bottleneck, as mentioned above, is known to affect most multi-element methods suggested in the literature.[24,25] However, multi-element collocation methods can greatly benefit from *hp*-adaptivity concepts, which are well established in the finite element literature.[26] In this context, adaptive *h*-refinement, also referred to as *h*-adaptivity, refers to the adaptive decomposition of the computational domain into subdomains where local polynomial approximations are developed, while *p*-adaptivity refers to increasing the local polynomial degree within a subdomain. These concepts have already been successfully applied for UQ purposes in the context of the stochastic Galerkin method.[27] However, multi-element collocation methods so far rely on *h*-adaptivity only. In the best case, the *hp*-adaptive multi-element approach chooses the type of refinement that provides the highest convergence rate. That is, in elements where the function is locally analytic, *p*-refinement is carried out, leading to exponential convergence. Contrarily, *h*-refinement is performed in elements where the function shows reduced regularity, leading to algebraic convergence. For most practically relevant cases, this leads to large elements with a high regularity, where *p*-refinement is performed, and a fine grid around areas with reduced regularity, where *h*-refinement is preferred.

In this work, we present a new *hp*-adaptive multi-element stochastic collocation method, which re-uses existing model evaluations during both *h*- and *p*-refinement. While re-using model data after splitting the parameter domain is easily possible for regression methods, this is not true for approaches relying on structured grids. Our method solves this problem through the use of weighted Leja collocation points.[28] A given sequence of collocation points can be stabilized by adding additional Leja points in a greedy way. Interpolation stability is further ensured by sorting the collocation points according to the Leja ordering. Hence, on each element the collocation scheme can be stabilized by enriching the collocation set obtained after splitting. This information re-use improves the overall efficiency of the method, which is demonstrated with several examples, also in comparison to existing approaches. Although our idea of stabilizing subsets of Leja sequences after multielement refinement is established here in the context of hypercubes, extensions to unstructured elements could be possible as well, for instance, with Leja sequences outlined in Reference 29.

Moreover, the weighted Leja nodes naturally handle arbitrary parameter distributions, which is particularly advantageous in the case of h-refinement with non-uniform distributions. For p-adaptivity, total degree (TD) and dimension-adaptive[30,31] basis representations are utilized, but other choices of basis representations, for example, based on Smolyak sparse grids, can be easily implemented. The dimension-adaptive algorithm relies on the nestedness and granularity of Leja points, which allows to increase the polynomial order in each dimension one-by-one, and exploits possible parameter anisotropies.

Our approach contributes to the growing literature on adaptive surrogate models in the field of UQ and can be used both in forward and inverse UQ analysis, which we illustrate in the numerical examples. As a consequence of the underlying tensor product structure in case of h-refinement, we restrict ourselves to problems with low to moderate ($\mathcal{O}(10)$) dimensionality. In many practically relevant cases, high dimensional problems are reduced to moderate dimensions with the help of engineering experience and a-priori knowledge. Nevertheless, the remaining parameters may have a significant impact on the model response and can result in strong variations or discontinuities. The proposed approach is suitable to handle these types of problems.

The remaining of this article is structured as follows. In Section 2, we introduce the notation, the problem setting and some necessary preliminaries on stochastic collocation methods. Section 3 is concerned with stochastic collocation on Leja grids, based either on hierarchical interpolation or on PCE. In Section 4, we introduce our novel hp-adaptive multi-element stochastic method based on Leja grids, which additionally allows for re-using model evaluations after either h- or p-refinement. Numerical results showcasing the advantages of the proposed multi-element stochastic collocation method are reported in Section 5. Finally, conclusions are drawn in Section 6.

2 | MODEL PROBLEM AND PRELIMINARIES

In this section, we introduce the notation and the stochastic setting in particular. We also present the main ideas of stochastic collocation as a surrogate modeling approach for forward and inverse problems. Details of the collocation method are postponed until Section 3. Finally, we introduce a model problem based on a partial differential equation (PDE).

2.1 | Notation

We consider a parameter-dependent model that receives as input a parameter vector and estimates an output, represented through the map

$$g : \mathbf{x} \mapsto g(\mathbf{x}). \tag{1}$$

In (1), $\mathbf{x} \in \mathbb{R}^N$ denotes the input parameter vector and $g(\mathbf{x}) \in \mathbb{R}^{N_{\text{out}}}$ the model evaluation for the given input parameters. In forward analyses, for example, in the context of uncertainty propagation, $g(\mathbf{x})$ is typically called the QoI. In inverse problem settings, $g(\mathbf{x})$ represents the map from parameters to observations or the likelihood function. In cases where an evaluation $g(\mathbf{x})$ is computationally expensive, for example, the model in question is a high-fidelity numerical solver, an inexpensive approximation $\tilde{g}(\mathbf{x}) \approx g(\mathbf{x})$ that can reliably replace the original model is often desirable. This is particularly true for multi-query tasks in UQ, optimization, or design space exploration. We refer to such an approximation as a *surrogate model*.

In the context of this work, the inputs $\mathbf{x}$ are assumed to be realizations of independent random variables (RVs) X_n, $n = 1, 2, \dots, N$, which are collected in the multivariate RV $\mathbf{X} = (X_1, X_2, \dots, X_N)^\top$, also referred to as a random vector. The random vector $\mathbf{X}$ is defined on the probability space (Θ, Σ, P), where Θ denotes the sample space, Σ the sigma algebra of events, and $P : \Sigma \to [0, 1]$ the probability measure. We further introduce the image space $\Xi \subset \mathbb{R}^N$ such that $\mathbf{X} : \Theta \to \Xi$, equivalently, $\mathbf{X}(\theta) = \mathbf{x}$ with $\theta \in \Theta$ and $\mathbf{x} \in \Xi$, as well as the probability density function (PDF) $\pi_{\mathbf{X}}(\mathbf{x}) : \Xi \to \mathbb{R}_{\geq 0}$. Note that the notation differentiates between a random vector $\mathbf{X}$ and a realization $\mathbf{x} = \mathbf{X}(\theta)$. Recalling that the individual RVs X_n, $n = 1, \dots, N$, are mutually independent, it holds that $\pi_{\mathbf{X}}(\mathbf{x}) = \prod_{n=1}^{N} \pi_{X_n}(x_n)$ and $\Xi = \Xi_1 \times \cdots \times \Xi_N$, where $\pi_{X_n}(x_n)$ and Ξ_n refer to the marginal (univariate) PDFs and image spaces, respectively. Then, the model output is a RV dependent on the input random vector $\mathbf{X}$. Note that the model itself remains purely deterministic, that is, the uncertainty in the model

output is caused only due to the random input parameters. Accordingly, a RV realization $\mathbf{x}$ corresponds to an estimation $g(\mathbf{x})$ of fixed value.

2.2 | Surrogate modeling for forward and inverse problems

The term *stochastic collocation*[32,33] refers to a class of sampling-based methods where a model g is evaluated (sampled) on a set of so-called *collocation points* $\mathcal{X} = \{\mathbf{x}_j\}_{j=1}^{J}$. The pairs of collocation points and corresponding model evaluations $\{\mathbf{x}_i, g(\mathbf{x}_j)\}_{j=1}^{J}$ are then utilized to compute a surrogate model $\tilde{g}(\mathbf{x}) \approx g(\mathbf{x})$, which typically takes the form

$$\tilde{g}(\mathbf{x}) = \sum_{i=1}^{I} \beta_i B_i(\mathbf{x}), \tag{2}$$

where $\beta_i \in \mathbb{R}^{N_{\text{out}}}$ are coefficients, B_i appropriately chosen basis functions, and I the size of the basis. In a collocation approach, the coefficients are determined based on the condition

$$g(\mathbf{x}_j) = \tilde{g}(\mathbf{x}_j), \ \forall j = 1, \ \dots \ , J, \tag{3}$$

where $I = J$, resulting in a linear system of equations for β_i. The well-known Lagrange basis even results in $\beta_i = g(\mathbf{x}_i)$. A major challenge for surrogate models in the form (2) is the rapidly growing size of the basis when the input dimension is large. In this case, constructing a sparse basis has often proven to be useful.[32,34] Even in case of sparse approximation, the construction of the polynomial basis relies on tensor product polynomials, hence, the one-dimensional case serves as the starting point. This approach is adopted in Section 3.

Before discussing details of the *hp*-adaptive collocation approach, we briefly outline the role of surrogate models in inverse and forward UQ. In forward UQ, the goal is to compute moments or a distribution function of the QoI. This can be achieved, either by sampling the inexpensive surrogate model, or by choosing a suitable basis from which the sought quantities can be readily obtained. The polynomial chaos basis, for instance, allows to infer the moments directly from the basis coefficients without involving any additional sampling routine.[8,35] In inverse UQ, an important object is the likelihood function which, in the common case of Gaussian noise, reads

$$L(\mathbf{x}|\mathbf{b}) = \exp\left(-\frac{1}{2}\|\mathbf{b} - \mathcal{G}(\mathbf{x})\|_{\mathbf{\Sigma}^{-1}}\right), \tag{4}$$

where $\mathbf{b} \in \mathbb{R}^D$ denotes the observation vector and $\mathbf{\Sigma}$ the noise covariance matrix. Then, $\mathcal{G} : \mathbb{R}^N \to \mathbb{R}^D$ represents the nonlinear relationship between input parameters and observations. In (4), the model-data mismatch is measured in the discrete Euclidean norm, weighted with the inverse covariance matrix. When running a Markov chain Monte Carlo analysis, the likelihood function and hence, the model included in $\mathcal{G}$, needs to be evaluated for each entry of the chain. To reduce the large computational workload, a surrogate model $\tilde{g}(\mathbf{x}) \approx g(\mathbf{x}) = \mathcal{G}(\mathbf{x})$ can be employed.[36] A more recent approach constructs a surrogate for the likelihood as $\tilde{g}(\mathbf{x}) \approx g(\mathbf{x}) = L(\mathbf{x}|\mathbf{b})$, which allows to avoid any sampling-based analysis.[37] However, in the large data—small noise regime, the likelihood function is highly concentrated and global surrogate modeling is difficult. Adaptive[38] and possibly multilevel[39] approaches are a popular remedy in this case. Another possibility is to use the spectral stochastic embedding (SSE) method.[37] Other approaches are based on transport maps.[40,41] We will show in Section 5, that our multi-element stochastic collocation method is also capable of handling such highly concentrated likelihood functions.

2.3 | A PDE-based model problem

For the sake of concreteness, we consider the Helmholtz equation with a stochastic inhomogeneous refractive index as a model problem. However, the formulation derived in this subsection will be general enough to cover other problems as well. Let $D = [0,1]^2$ denote the computational domain with boundaries

$$\Gamma_1 = \{(r_1, r_2) \mid r_1 = 0, r_2 \in [0,1]\}, \quad \Gamma_2 = \{(r_1, r_2) \mid r_1 = 1, r_2 \in [0.3, 0.7]\}, \quad \Gamma_3 = \partial D \setminus (\Gamma_1 \cup \Gamma_2).$$

Let ∂_ν denote the derivative with respect to the outgoing normal vector. The strong form of the problem reads

$$\Delta u(\mathbf{x};\mathbf{r}) + k^2 n(\mathbf{x};\mathbf{r}) u(\mathbf{x};\mathbf{r}) = 0 \qquad \mathbf{r} \in D, \tag{5a}$$

$$\partial_\nu u(\mathbf{x};\mathbf{r}) = g(\mathbf{r}) \qquad \mathbf{r} \in \Gamma_1, \tag{5b}$$

$$\partial_\nu u(\mathbf{x};\mathbf{r}) = (i\beta - \alpha) u(\mathbf{x};\mathbf{r}) \qquad \mathbf{r} \in \Gamma_2, \tag{5c}$$

$$\partial_\nu u(\mathbf{x};\mathbf{r}) = 0 \qquad \mathbf{r} \in \Gamma_3, \tag{5d}$$

for $\pi_{\mathbf{X}}$-almost all $\mathbf{x} \in \Xi$. The uncertainty in the index n is modeled with a log-Karhunen–Loève expansion of the form

$$n(\mathbf{x};\mathbf{r}) = \exp\left(n_0(\mathbf{r}) + \sum_{j=1}^{N} x_j \Psi_j(\mathbf{r}) \right). \tag{6}$$

Here, the modes $\Psi_j(\mathbf{r}) = \sqrt{\lambda_j}\varphi_j(\mathbf{r})$ are obtained from the squared exponential kernel with correlation length $l = 0.05$ and standard deviation $\sigma = 1$. In the numerical experiments we set $n_0 = 0$, see Section 5.2.4.

We consider a Ξ-strong/D-weak form based on $H^1(D)$, where we seek for $u(\mathbf{x}) \in H^1(D)$ $\pi_{\mathbf{X}}$-almost everywhere in Ξ subject to

$$\underbrace{\int_D \nabla u(\mathbf{x}) \cdot \nabla v^* \, \mathrm{d}r - k^2 \int_D n(\mathbf{x}) u(\mathbf{x}) v^* \, \mathrm{d}r - \int_{\Gamma_2} (i\beta - \alpha) u(\mathbf{x}) v^* \mathrm{d}r}_{=a(\mathbf{x};u,v)} = \underbrace{\int_{\Gamma_1} g v^* \mathrm{d}r}_{=l(v)}, \quad \forall v \in H^1(D), \tag{7}$$

where * refers to the complex conjugate and where $u(\mathbf{x})$ is short for $u(\mathbf{x};\cdot)$. We apply a standard, lowest order, piecewise continuous finite element method on a regular grid with maximum mesh size h, to compute a numerical solution $u_h(\mathbf{x}) \in V_h \subset H^1(D)$, subject to

$$a(\mathbf{x}; u_h, v_h) = l(v_h), \quad \forall v_h \in V_h. \tag{8}$$

The QoI is here obtained as a linear functional of the solution, that is,

$$g(\mathbf{x}) = q(u_h(\mathbf{x})), \quad q \in V_h^*. \tag{9}$$

The PDE-background of the problem provides more structure, which we exploit to derive an error indicator for the *hp*-adaptive method.

3 | STOCHASTIC COLLOCATION ON LEJA GRIDS

Next, we elaborate on the concept of hierarchical Leja collocation in one- and multiple dimensions and its relation to polynomial chaos surrogates. These topics have been previously presented in the literature in detail, nevertheless, we shortly re-iterate the relevant concepts for the ease of the reader.

3.1 | Leja sequences

In their original form, Leja sequences[42] are sequences of points $\{x_i\}_{i\geq 0}$, $x_i \in [-1,1]$, $\forall i \geq 0$, where the initial point x_0 is chosen arbitrarily within $[-1,1]$ and the remaining points are computed by solving the optimization problem

$$x_i = \arg\max_{x\in[-1,1]} \prod_{k=0}^{i-1} |x - x_k| \,. \tag{10}$$

A point sequence produced by formula (10) is an *unweighted* Leja sequence. *Weighted* Leja sequences are constructed by incorporating a continuous and positive weight function in the definition of Leja sequences.[28] Assuming that the weight

function is a PDF $\pi_X(x) : \Xi \to \mathbb{R}_{\geq 0}$, the points of the corresponding weighted Leja sequence are given as

$$x_i = \arg\max_{x \in \Xi} \sqrt{\pi_X(x)} \prod_{k=0}^{i-1} |x - x_i| , \tag{11}$$

where again the initial Leja point x_0 is chosen arbitrarily within the image space Ξ.

Leja points are known to perform well when employed as interpolation or quadrature nodes.[31,43-46] With respect to interpolation in particular, the Lebesgue constant of Leja sequence based interpolation grids is known to grow subexponentially,[47-49] thus resulting in stable interpolations. Additionally, interpolation and quadrature grids with respect to any continuous PDF can be constructed by using weighted Leja sequences.[39,50-52] Moreover, due to the fact that Leja sequences are by definition nested, that is, $\{x_i\}_{i=0}^{j} \subset \{x_i\}_{i=0}^{j+1}$, they allow for re-using readily available Leja points and model evaluations on those points in case the sequence is further expanded. Due to the nestedness property, Leja points are natural candidates for constructing sparse interpolation or quadrature grids.[30,38,53-55] Note that nested interpolation and quadrature grids can be obtained with other types of nodes, such as Clenshaw–Curtis.[56] However, Leja sequences have the additional attractive feature of retaining the nestedness property even if a single point is added to the sequence, thus allowing for arbitrary grid granularity. Nestedness and grid granularity become additionally important when considering efficient *h*-refinement, as discussed in Section 4.

3.2 | Hierarchical interpolation on Leja grids

In interpolation based stochastic collocation methods,[32-34] the surrogate model $\tilde{g}(\mathbf{x}) \approx g(\mathbf{x})$ is a global polynomial approximation computed by means of interpolation. For simplicity, in the following we assume a scalar model output $g(\mathbf{x}) \in \mathbb{R}$, however, the extension to vector-valued outputs is straightforward.

Considering a model $g(x)$ dependent on a single parameter and an interpolation grid $\mathcal{X}_j = \{x_i\}_{i=0}^{j}$, a Lagrange interpolation based approximation is given as

$$\tilde{g}(x) = \mathcal{I}_j[g](x) = \sum_{i=0}^{j} g(x_i)\, l_i^j(x), \tag{12}$$

where l_i^j are Lagrange polynomials of degree j, defined as

$$l_i^j(x) = \prod_{k=0, k\neq i}^{j} \frac{x - x_k}{x_i - x_k}, \qquad l^0 = 1. \tag{13}$$

Assuming that the interpolation grid $\mathcal{X}_j$ coincides with a Leja sequence, a sequence of nested grids can be defined, such that $\mathcal{X}_0 = \{x_0\} \subset \mathcal{X}_1 = \{x_0, x_1\} \subset \cdots \subset \mathcal{X}_j = \{x_0, x_1, \ldots, x_j\}$, where each grid defines an interpolation operator $\mathcal{I}_i[g](x)$, $i = 0, 1, \ldots, j$. Then, replacing the Lagrange polynomials with the hierarchical polynomials[30]

$$h^i(x) = \prod_{k=0}^{i-1} \frac{x - x_k}{x_i - x_k}, \qquad h^0(x) = 1, \tag{14}$$

of polynomial degree i, $i = 0, 1, \ldots, j$, a sequential interpolation formula can be derived, such that

$$\tilde{g}(x) = \mathcal{I}_j[g](x) = \sum_{i=0}^{j} s_i h^i(x) = \sum_{i=0}^{j} \left(g(x_i) - \mathcal{I}_{i-1}[g](x_i)\right) h^i(x), \tag{15}$$

where the coefficients s_i are called the hierarchical surpluses.[57] Note that $\mathcal{I}_{-1}$ is a null operator, that is, $\mathcal{I}_{-1}[g](x) = 0$, accordingly, $s_0 = g(x_0)$. We shall refer to the interpolation format (15) as *hierarchical* interpolation.

The interpolation formats (12) and (15) are in fact equivalent, as the interpolating polynomial is unique for a given interpolation grid irrespective of the choice of the polynomial basis.[58] The hierarchical representation (15) offers the

advantage that, if the Leja sequence based interpolation grid is expanded with new nodes, the already computed basis polynomials remain unchanged. Accordingly, each hierarchical polynomial $h^i, i = 0, 1, \dots, j$, has a degree equal to i which is unique. Irrespective of the choice of basis polynomials, a crucial advantage of using Leja sequences as interpolation grids is that the readily available model evaluations $g(x_i), x_i \in \mathcal{X}_j, i = 0, 1, \dots, j$, that are used to construct the interpolation $\mathcal{I}_j[g](x)$, can be re-used for the interpolation $\mathcal{I}_{j+1}[g](x)$, due to the fact that $\mathcal{X}_j \subset \mathcal{X}_{j+1}$. Accordingly, the model needs to be evaluated only on the new Leja point x_{j+1}, where $\{x_{j+1}\} = \mathcal{X}_{j+1} \setminus \mathcal{X}_j$.

Moving to an N-variate model $g(\mathbf{x})$, we introduce a multi-index notation such that a multi-index $\mathbf{j} = (j_1, \dots, j_N)$ can be uniquely associated with the multivariate Leja node $\mathbf{x_j} = (x_{1,j_1}, \dots, x_{N,j_N})$, the multivariate hierarchical polynomial

$$H_{\mathbf{j}}(\mathbf{x}) = \prod_{n=1}^{N} h_n^{j_n}(x_n), \tag{16}$$

and the tensor grid

$$\mathcal{X}_{\mathbf{j}} = \mathcal{X}_{1,j_1} \times \mathcal{X}_{2,j_2} \times \cdots \times \mathcal{X}_{N,j_N}, \tag{17}$$

where each univariate grid $\mathcal{X}_{n,j_n} = \{x_{n,0}, x_{n,1}, \dots, x_{n,j_n}\}$ is assumed to be a Leja sequence. A multivariate hierarchical interpolation scheme can then be derived as follows. We consider a downward closed multi-index set Λ, such that

$$\forall \mathbf{j} \in \Lambda \Rightarrow \mathbf{j} - \mathbf{e}_n \in \Lambda, \forall n = 1, \dots, N, \tag{18}$$

where $\mathbf{e}_n$ denotes the unit vector in the nth dimension. The associated interpolation grid is given by

$$\mathcal{X}_\Lambda = \bigcup_{\mathbf{j} \in \Lambda} \mathcal{X}_{\mathbf{j}}. \tag{19}$$

Since each multi-index $\mathbf{j} \in \Lambda$ uniquely defines a multivariate Leja node $\mathbf{x_j} \in \mathcal{X}_\Lambda$, it holds that $\#\Lambda = \#\mathcal{X}_\Lambda$, where $\#$ denotes the cardinality of a set. Then, a sequence of nested, downward closed multi-index sets $(\Lambda_w)_{w=1}^W$, $\#\Lambda = W$, can be defined, such that $\Lambda_w = \{\mathbf{j}_1, \mathbf{j}_2, \dots, \mathbf{j}_w\}$, $\Lambda_{w-1} \subset \Lambda_w$ $(\Lambda_0 = \emptyset)$, $\#\Lambda_w = w$, and $\Lambda_w \setminus \Lambda_{w-1} = \{\mathbf{j}_w\}$. This also implies that $\Lambda_1 = \{\mathbf{j}_1 = \mathbf{0} = (0, 0, \dots, 0)\}$. A corresponding grid sequence $(\mathcal{X}_{\Lambda_w})_{w=1}^W$ is similarly defined, where $\mathcal{X}_{\Lambda_w} = \{\mathbf{x}_{\mathbf{j}_1}, \mathbf{x}_{\mathbf{j}_2}, \dots, \mathbf{x}_{\mathbf{j}_w}\}$, $\mathcal{X}_{\Lambda_{w-1}} \subset \mathcal{X}_{\Lambda_w}$ $(\mathcal{X}_{\Lambda_0} = \emptyset)$, $\#\mathcal{X}_{\Lambda_w} = w$, and $\mathcal{X}_{\Lambda_w} \setminus \mathcal{X}_{\Lambda_{w-1}} = \{\mathbf{x}_{\mathbf{j}_w}\}$, that is, a single Leja node is added to the next grid in sequence, as in the univariate case. Simplifying the notation to $\mathbf{x}_{\mathbf{j}_w} = \mathbf{x}_w$ and $H_{\mathbf{j}_w}(\mathbf{x}) = H_w(\mathbf{x})$, the multivariate hierarchical interpolation reads

$$\tilde{g}(\mathbf{x}) = \mathcal{I}_\Lambda[g](\mathbf{x}) = \sum_{\mathbf{j} \in \Lambda} s_{\mathbf{j}} H_{\mathbf{j}}(\mathbf{x}) = \sum_{w=1}^{W} s_w H_w(\mathbf{x}) = \sum_{w=1}^{W} \left(g(\mathbf{x}_w) - \mathcal{I}_{\Lambda_{w-1}}[g](\mathbf{x}_w) \right) H_w(\mathbf{x}), \tag{20}$$

where $s_1 = s_{\mathbf{0}} = g(\mathbf{x}_1) = g(\mathbf{x_0})$, that is, $\mathcal{I}_{\Lambda_0}$ is a null operator, similar to the univariate case.

The hierarchical Leja interpolation (20) can be constructed by means of a so-called dimension-adaptive algorithm, which sequentially adds terms of high impact to the polynomial approximation, while neglecting non-influential terms. This algorithm was first suggested for adaptive, sparse quadrature.[57] Similar algorithms have been employed for Leja based, dimension-adaptive interpolation.[28,30,31] In brief, the dimension-adaptive Leja interpolation algorithm consists of the following steps:

1. A hierarchical interpolation based on a downward closed multi-index set Λ is assumed to exist, along with the corresponding Leja interpolation grid $\mathcal{X}_\Lambda$. Otherwise, the algorithm is initialized with the zero multi-index, such that $\Lambda = \{\mathbf{0}\}$. This set is referred to as the *active* set.
2. The set of *admissible* multi-indices is computed, denoted as Λ^{adm}, such that $\Lambda \cup \Lambda^{\text{adm}}$ is downward closed. The corresponding admissible Leja nodes are also computed.
3. The hierarchical surplus corresponding to each admissible multi-index is computed.
4. The admissible multi-index with the maximum hierarchical surplus in absolute value is added to the active multi-index set. The corresponding Leja node is added to the interpolation grid.

5. The steps 2–5 are repeated until the desired approximation accuracy or the maximum allowed number of model evaluations is reached.

After the termination of the algorithm, the final interpolation is constructed using both the active and the admissible multi-indices, such that no Leja nodes and corresponding model evaluations remain unused.

Alternatively, Leja based interpolations can be constructed using interpolation grids with an a priori defined structure, for example, tensor product, TD, hyperbolic cross, or Smolyak sparse grids.[32] In this work, aside from the dimension-adaptive algorithm described above, TD grids are also employed, in which case the multi-index set is given as

$$\Lambda^{\mathrm{TD}} = \left\{ \mathbf{j} : \|\mathbf{j}\|_1 \leq P, P \in \mathbb{Z}_{\geq 0} \right\}. \tag{21}$$

These two choices regarding the employed multi-index set are mainly motivated by the fact that they greatly simplify the transformation to a PCE basis, see Section 3.3.

3.3 | Polynomial chaos expansion on Leja grids

Similar to the stochastic collocation method presented in Section 3.2, the PCE method[59,60] also produces a global polynomial approximation $\tilde{g}(\mathbf{x}) \approx g(\mathbf{x})$, however, instead of Lagrange or hierarchical polynomials, the polynomial basis consists of orthonormal polynomials with respect to the PDF of the input parameters. Again, in the following, we consider a scalar model output $g(\mathbf{x}) \in \mathbb{R}$ for simplicity. The method can be applied to address vector-valued model outputs as well.

The PCE is given as

$$\tilde{g}(\mathbf{x}) = \sum_{\mathbf{p} \in \Lambda} c_{\mathbf{p}} \Psi_{\mathbf{p}}(\mathbf{x}), \tag{22}$$

where $c_{\mathbf{p}}$ are scalar coefficients and $\Psi_{\mathbf{p}}(\mathbf{x}) = \prod_{n=1}^{N} \psi_n^{p_n}(x_n)$ are multivariate polynomials that satisfy the orthonormality condition

$$\mathbb{E}\left[\Psi_{\mathbf{p}} \Psi_{\mathbf{q}}\right] = \int_{\Xi} \Psi_{\mathbf{p}}(\mathbf{x}) \Psi_{\mathbf{q}}(\mathbf{x}) \pi_{\mathbf{X}}(\mathbf{x}) \, \mathrm{d}\mathbf{x} = \delta_{\mathbf{pq}}, \tag{23}$$

where $\mathbb{E}[\cdot]$ denotes the expectation operator, $\delta_{\mathbf{pq}} = \prod_{n=1}^{N} \delta_{p_n q_n}$, and δ_{ij} is the Kronecker delta. The multi-index set Λ now comprises multi-indices corresponding to the polynomial degrees of the PCE polynomials, that is, $\mathbf{p} = (p_1, \ldots, p_N)$.

Note that, assuming a readily available Lagrange or hierarchical interpolation, an equivalent PCE can be computed with a basis transformation to orthonormal polynomials.[61-63] As previously noted in Section 3.2, in this work, such transformations take place using Leja interpolations based either on TD multi-index sets,[32] which have an a priori defined structure, or multi-index sets constructed with a dimension-adaptive algorithm. In both cases, there exists a one-to-one relation between the hierarchical and orthonormal polynomials in terms of their degrees. Therefore, the interpolation and PCE multi-index sets are identical and the basis transformation is greatly simplified.[64]

An important aspect of the PCE is that it provides UQ metrics regarding the approximated model output, which are obtained with negligible computational cost by simply post-processing the PCE's terms. In particular, exploiting the orthonormality property of the basis polynomials, the mean value and the variance of the output can be estimated as

$$\mu = c_{\mathbf{0}}, \tag{24}$$

$$\sigma^2 = \sum_{\mathbf{p} \in \Lambda \setminus \{\mathbf{0}\}} c_{\mathbf{p}}^2. \tag{25}$$

Note that each PCE coefficient $c_{\mathbf{p}}$, $\mathbf{p} \in \Lambda \setminus \{\mathbf{0}\}$, corresponds to a partial variance that contributes to the total variance of the model output. Based on the known connection between variance based sensitivity analysis metrics and the PCE method,[35,65] all PCE coefficients except for $c_{\mathbf{0}}$ can be interpreted as sensitivity indicators.[53,66]

4 | MULTI-ELEMENT STOCHASTIC COLLOCATION WITH ADAPTIVE REFINEMENT

Global polynomial approximations face major difficulties in terms of accuracy and convergence when applied to problems with discontinuities in the parameter space. Additionally, steep gradients or large variations in the objective function also hinder the convergence of global approximation approaches. To address this bottleneck, several *multi-element* approximation techniques have been proposed,[12,15-17,19,20,67] where the parameter space is decomposed into subspaces of improved regularity. Local polynomial approximations are developed in the subspaces, which are subsequently combined to provide a global surrogate model. Borrowing terminology from the finite element method (FEM) literature,[68-72] we call *h*-refinement the refinement of the parameter domain decomposition with additional subdomains and *p*-refinement the refinement of a local polynomial approximation with additional terms. In the following, we build upon the Leja based hierarchical interpolation and PCE methods presented in Section 3 and develop a multi-element polynomial approximation method which refines adaptively both the parameter domain decomposition and the polynomial approximations according to the particularities of the problem at hand.

4.1 | Multi-element hierarchical interpolation and polynomial chaos expansion

Following the work of Wan and Karniadakis,[15] let the image space of the random vector $\mathbf{X} = (X_1, \dots, X_N)$ be given as $\Xi = [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_N, b_N]$, where $a_n, b_n \in \mathbb{R}$, $n = 1, \dots, N$, with extension to infinity. The image space is decomposed into K non-overlapping hypercubes, subsequently referred to as *elements*, such that $\Xi = \bigcup_{k=1}^{K} d_k$, where each element d_k is defined as $d_k = d_1^{(k)} \times d_2^{(k)} \times \dots \times d_N^{(k)}$ with $d_n^{(k)} = [a_n^{(k)}, b_n^{(k)})$ if $b_n^{(k)} \neq b_n$, respectively $d_n^{(k)} = [a_n^{(k)}, b_n^{(k)}]$ if $b_n^{(k)} = b_n$.

We proceed by introducing conditional expectations and local densities associated to the partition $\{d_k\}$. To that end, let $\mathbb{1}_k(\mathbf{x})$ denote the indicator function defined as

$$\mathbb{1}_k(\mathbf{x}) = \begin{cases} 1, & \mathbf{x} \in d_k, \\ 0, & \text{else.} \end{cases} \tag{26}$$

We write $Y_k = \mathbb{1}_k(\mathbf{X})$ in the following. Then,

$$\mathbb{E}[\mathbf{X}|Y_k = 1] = \frac{\mathbb{E}[\mathbf{X}Y_k]}{P(Y_k = 1)}, \quad k = 1, \dots, K. \tag{27}$$

We then use the definition of a PDF via an expected value

$$\pi_X(x) = \mathbb{E}[\delta(X - x)], \tag{28}$$

where δ denotes the Dirac delta distribution, to obtain conditional PDFs for the partition. In particular, we obtain

$$\pi_{\mathbf{X}}(\mathbf{x}|Y_k = 1) = \frac{\mathbb{E}[\delta(\mathbf{X} - \mathbf{x})Y_k]}{P(Y_k = 1)}, \tag{29}$$

$$\pi_k(\mathbf{x}^{(k)}) = \pi_{\mathbf{X}}(\mathbf{x}^{(k)}|Y_k = 1) = \frac{\pi_{\mathbf{X}}\left(\mathbf{x}^{(k)}\right)}{\varrho_k}, \quad \text{for } \mathbf{x} = \mathbf{x}^{(k)} \in d_k. \tag{30}$$

The normalization factor $\varrho_k = \Pr(Y_k = 1)$ is determined as

$$\varrho_k = \prod_{n=1}^{N} \int_{a_n^{(k)}}^{b_n^{(k)}} \pi_{X_n}(x_n)\, \mathrm{d}x_n. \tag{31}$$

Note that $P(Y_k = 1) \neq 0$ by construction, since $a_n^{(k)}$, respectively $b_n^{(k)}$ are chosen such that $\varrho_k \neq 0$, see Section 4.2.4.

In each element, a local interpolation grid $\mathcal{X}^{(k)}_{\Lambda^{(k)}}$ is generated following a local multi-index set $\Lambda^{(k)}$. The weighted Leja points in each element are obtained using the local PDFs, where the local univariate Leja sequences are defined as

$$x^{(k)}_{n,j_n} = \arg\max_{x \in d^{(k)}_n} \sqrt{\pi_{X_n}(x)/\varrho_k} \prod_{i=0}^{j_n-1} \left| x - x^{(k)}_{n,i} \right|, \tag{32}$$

and correspond to local univariate interpolation grids $\mathcal{X}^{(k)}_{n,j_n} = \left\{ x^{(k)}_{n,0}, x^{(k)}_{n,1}, \dots, x^{(k)}_{n,j_n} \right\} \subset d^{(k)}_n$. Local hierarchical polynomial interpolations $\tilde{g}_k\left(\mathbf{x}^{(k)}\right) = \mathcal{I}_{\Lambda^{(k)}}[g](\mathbf{x}^{(k)})$, $\mathbf{x}^{(k)} \in d_k, k = 1, \dots, K$, are then computed as described in Section 3.2 and are subsequently transformed into local PCEs as discussed in Section 3.3.

Once available, the local surrogate models are combined to form global surrogate models as

$$\tilde{g}(\mathbf{x}) = \sum_{k=1}^{K} \mathbb{1}_k(\mathbf{x}) \tilde{g}_k(\mathbf{x}). \tag{33}$$

Using the PCE based global surrogate model, the mean and variance of the output can be estimated as

$$\mu = \sum_{k=1}^{K} c^{(k)}_{\mathbf{0}} \varrho_k, \tag{34}$$

$$\sigma^2 = \sum_{k=1}^{K} \left(\sigma^2_k + (c^{(k)}_{\mathbf{0}} - \mu)^2 \right) \varrho_k, \tag{35}$$

where the local variances σ^2_k are estimated as in (25).[15] For uniformly distributed parameters, the local RVs $\mathbf{X}^{(k)}$ also follow a uniform distribution which can be easily normalized to the domain d_k, that is, no additional costs arise for computing the normalization factors in (31). This also allows the further use of Legendre polynomials for the PCE basis.[15] If non-uniform distributions are employed, the local orthonormal polynomials must be constructed numerically.[67]

4.2 | *hp*-adaptivity

Adaptive approximation strategies based on *hp*-refinement are widely used for FEM based approximations.[68-72] Typically, these strategies follow the step sequence

$$\textit{Solve} \Rightarrow \textit{Estimate} \Rightarrow \textit{Mark} \Rightarrow \textit{Refine},$$

which is iterated until the desired approximation accuracy has been reached. The flowchart shown in Figure 1 provides a systematic overview of the *hp*-refinement procedure followed in this work for developing polynomial approximations by means of the suggested multi-element stochastic collocation method. The individual steps are explained in detail next.

4.2.1 | SOLVE

Given a decomposition of the parameter (image) space $\Xi = \bigcup_{k=1}^{K} d_k$, a hierarchical interpolation based on the local Leja grid $\mathcal{X}^{(k)}_{\Lambda^{(k)}}$ is computed in each element d_k. Subsequently, the local hierarchical polynomial bases are transformed to orthonormal ones, thus obtaining a local PCE in each element.

4.2.2 | ESTIMATE

We consider two settings for error estimation. On the one hand, we rely on the structure of the PDE model problem of Section 2.3 and employ a duality-based error indicator. However, this approach requires access to the solution and matrices after finite element assembly and is therefore not always applicable. Therefore, we employ error indicators based on partial variances as a generally applicable alternative.

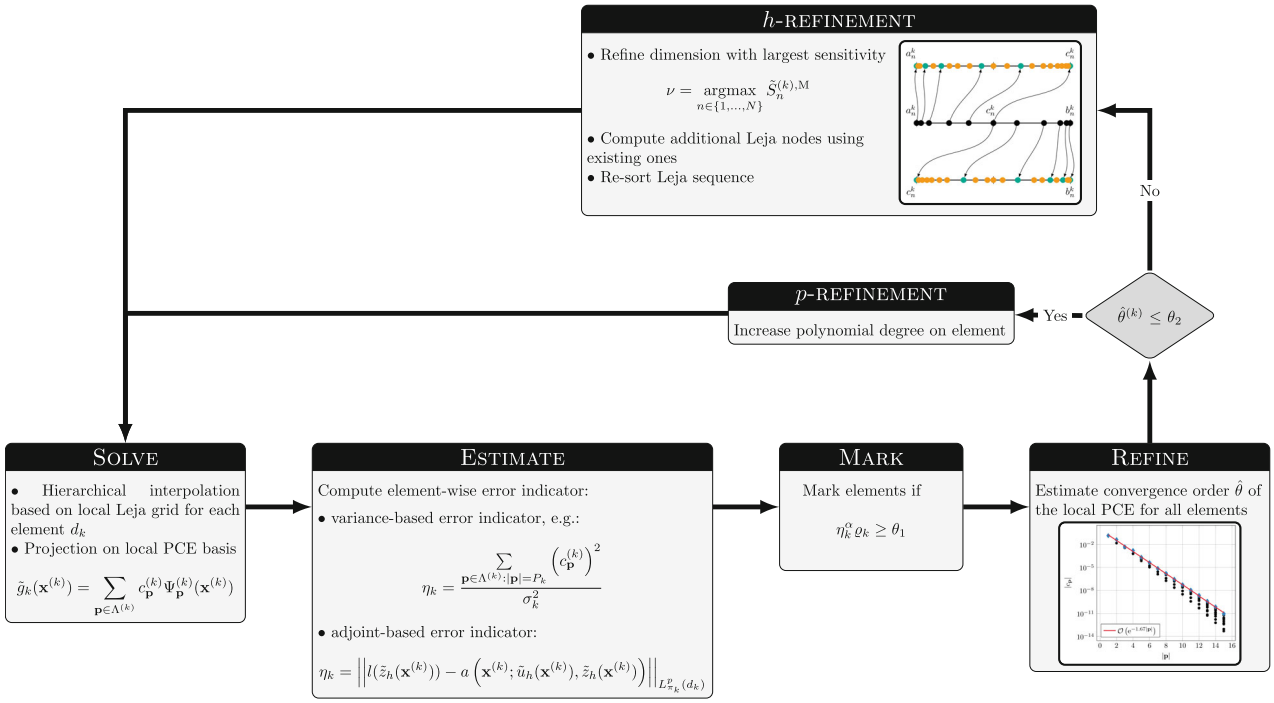

FIGURE 1 Sketch of the *hp*-adaptive multi-element stochastic collocation method.

Adjoint-based error estimation

Similar to previous works,[54,73] we compute the adjoint variable $z_h(\mathbf{x}) \in V_h$, subject to

$$a(\mathbf{x}; v_h, z_h) = q(v_h), \quad \forall v_h \in V_h. \tag{36}$$

Then, we can obtain the pointwise representation of the surrogate error

$$g(\mathbf{x}) - \tilde{g}(\mathbf{x}) = l(z_h(\mathbf{x})) - a(\mathbf{x}; \tilde{u}_h(\mathbf{x}), z_h(\mathbf{x})), \tag{37}$$

with $\tilde{g}(\mathbf{x}) = q(\tilde{u}_h(\mathbf{x}))$. Even if (37) is not computable, because it would require knowledge of the adjoint for every parameter value, an accurate representation can be obtained by employing an adjoint surrogate $\tilde{z}_h$, which results in the error indicator

$$\eta(\mathbf{x}) = l(\tilde{z}_h(\mathbf{x})) - a(\mathbf{x}; \tilde{u}_h(\mathbf{x}), \tilde{z}_h(\mathbf{x})). \tag{38}$$

Finally, we employ local error indicators

$$\eta_k = \|\eta(\mathbf{x}^{(k)})\|_{L^p_{\pi_k}(d_k)} = \left(\int_{d_k} |\eta(\mathbf{x}^{(k)})|^p \, \pi_k(\mathbf{x}^{(k)}) \mathrm{d}\mathbf{x}^{(k)}\right)^{1/p}, \tag{39}$$

where $p = 1, 2$ are common choices in our adaptive algorithm. Note that a Monte Carlo approach is used to estimate the integral, since the error indicator is cheap to evaluate. The surrogate models $\tilde{u}_h(\mathbf{x})$ and $\tilde{z}_h(\mathbf{x})$ are created in addition to the surrogate model of the QoI. However, they share the same multi-index sets $\Lambda^{(k)}$, equivalently, the same polynomial bases employed in $\tilde{g}(\mathbf{x})$.

Variance-based error estimation

Previous works have suggested error estimators for TD and Smolyak bases.[15,16] In case of a TD basis, the error indicator reads

$$\sigma_k^{-2} \sum_{\mathbf{p}\in\Lambda^{(k)}:\|\mathbf{p}\|_1=P_k} \left(c_{\mathbf{p}}^{(k)}\right)^2, \tag{40}$$

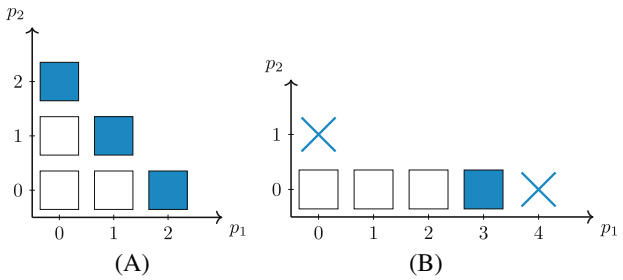

FIGURE 2 Visualization of index sets for $N = 2$ dimensions. The blue indices are employed in the error indicator. Squares show active indices, crosses the admissible indices. (A) Total degree basis. (B) Dimension-adaptive basis.

where $P_k = \max_{\mathbf{p}\in\Lambda^{(k)}} \|\mathbf{p}\|_1$. The error indicator (40) relates the highest-order partial variances to the total variance in an element d_k, see Figure 2A. Thus, if the estimation on the local variance σ_k^2 is sufficiently accurate, the coefficients in the nominator will be small, thus resulting in a small error indicator value as well.

In the case of an anisotropic basis, for example, obtained with a dimension-adaptive algorithm, the only restriction on the local multi-index set $\Lambda^{(k)}$ is that it is downward closed. Then, the error indicator (40) might not be informative regarding the accuracy of the local polynomial approximation, for example, in the case of multi-index sets with a single dominant dimension. Considering the arbitrary shape of the basis, the employed coefficients may originate from different polynomial levels, unlike in (40). It could therefore be beneficial to employ only admissible coefficients. However, this can lead to cases where only a relatively small number of coefficients are employed, which might not represent the actual estimation quality of the local variance. For instance, consider an objective function for which $f(\mathbf{x}) = f(-\mathbf{x})$ and an index set as pictured in Figure 2B. In this case, one dimension is dominant and the objective function demands for even polynomials. Hence, considering only the admissible indices, the error indicator might be misleading and not robust.

To solve this issue, a fixed number of coefficients is employed. Let $\Lambda^{(k)}$ denote the anisotropic multi-index set, including the admissible multi-indices. We then seek for the TD P_k such that

$$\max_{P_k} \left\{ \#\Lambda^{(k)} \geq \#\Lambda^{(k)}_{\mathrm{TD}} = \binom{N+P_k}{N} \right\}, \tag{41}$$

that is, we search for a TD basis with cardinality similar to the anisotropic basis. Then, the error indicator used in the dimension-adaptive case employs the last m_k added multi-indices (including the admissible indices), where m_k is determined with

$$m_k = \binom{N+P_k}{N} - \binom{N+P_k-1}{N}. \tag{42}$$

Note that with Equation (42) we employ as many coefficients as in the case of a TD basis. Assuming that $\#\Lambda^{(k)} = W_k$ and defining a nested sequence $\left\{\Lambda^{(k)}_{w_k}\right\}_{w_k=1}^{W_k}$ where $\Lambda^{(k)}_{w_k}$ is downward closed and $\#\Lambda^{(k)}_{w_k} = w_k$, the last m_k multi-indices are included in the set $\Lambda^{(k)} \setminus \Lambda^{(k)}_{W_k-m_k}$. Then, the modified error indicator is given by

$$\sigma_k^{-2} \sum_{\mathbf{p}\in\Lambda^{(k)}\setminus\Lambda^{(k)}_{W_k-m_k}} \left(c^{(k)}_{\mathbf{p}}\right)^2. \tag{43}$$

Due to the fact that the most recently added partial variances are employed, a similar behavior to the error indicator (40) is to be expected. Note that the error-indicator (43) is purely heuristic, however, it has proven itself a viable choice in many numerical examples, see Section 5.

4.2.3 | MARK

Given the local error indicators η_k, $k = 1, \ldots, K$, an element d_k is marked for either h- or p-refinement if

$$\eta_k^{\alpha}\varrho_k \geq \theta_1, \quad 0 < \alpha < 1, \tag{44}$$

where θ_1, also referred to as the marking parameter, and α are user-defined constants. Regarding the choice of the values of these constants, an extensive discussion can be found in the literature.[15] For all numerical experiments presented in this work, the constant value $\alpha = 0.5$ is used if a variance-based error indicator is employed and $\alpha = 1$ if an adjoint-based error indicator is used. The constant θ_1 is varied according to the problem under investigation. A discussion on the choice of θ_1 is found in Section 5.2.1. Note that marking strategies other than the criterion stated in (44) can be adopted as well, for example, the *fixed energy fraction* or the *maximum strategy* criteria suggested in the literature on *hp*-adaptive FEM.[68,70] In contrast to criterion (44), these strategies mark elements if the local error indicator η_k is large compared to the error indicators of all remaining elements. However, our numerical experiments showed that the proposed marking strategy delivers slightly better results. The algorithm terminates either when all local error indicators η_k are small and/or the elements are small. Note that the local error contribution of an element to the total L_2 error scales with the normalization factor ϱ_k, which is strongly related to the element size.[15] Consequently, *h*-refinement around a discontinuity will eventually terminate if the elements are small enough, since the contribution to the overall error is negligible.

4.2.4 | REFINE

Once the marking procedure is complete, a decision for either *h*- or *p*-refinement for each element d_k is made. To that end, the convergence order of the local PCE is estimated.

In deterministic finite element (FE)-analysis, *p*-refinement is carried out if the solution is locally analytic on the element. In fact, the smoothness properties can be estimated from a Legendre expansion of the solution.[74] In 1D, a linear model is fitted to the logarithmic values of the Legendre series and the slope $\hat{m}$ is extracted. Then, *p*-refinement is carried out if $\hat{\theta} = \mathrm{e}^{-\hat{m}}$ is above a certain threshold, corresponding to a sufficiently large Bernstein ellipse, hence, region of analyticity.

A similar approach can be applied to a (local) PCE approximation

$$\tilde{g}^{(k)}(\mathbf{x}^{(k)}) = \sum_{\mathbf{p}\in\Lambda^{(k)}} c_{\mathbf{p}}^{(k)} \Psi_{\mathbf{p}}^{(k)}(\mathbf{x}^{(k)}).$$

It has been shown[75] that for analytic functions the PCE coefficients satisfy

$$|c_{\mathbf{p}}^{(k)}| \leq C\mathrm{e}^{-\sum_{n=1}^{N} c_n p_n}. \tag{45}$$

In fact, the result was proven for the Legendre basis, but it holds for any PCE basis for which the associated density satisfies $\max_{\mathbf{x}\in\Xi} |\pi_{\mathbf{X}}(\mathbf{x})| \leq C$. The factors c_n depend on the size of the analytic extension of $[a_n, b_n]$ into the complex plane; the larger this extension, the larger the value of c_n.[32,75,76] In elements where the objective function $g(\mathbf{x})$ is smooth, a fast convergence, that is, a large constant $c = \min_n c_n$ is expected, see Figure 3A. Contrarily, in elements where the objective function has less regularity, a small constant c is expected, see Figure 3B. Hence, an estimation on the convergence rate of the PCE coefficients can serve as an indicator for *p*-, respectively *h*-refinement. Hence, we define

$$\hat{\theta} = \mathrm{e}^{-c}, \tag{46}$$

as the convergence indicator. To estimate $\hat{\theta}$, a linear model with slope $\hat{m}$ is fitted to the decaying PCE coefficients. The convergence indicator $\hat{\theta}$ is then estimated by solving

$$\begin{pmatrix} 1 & -1 \\ 1 & -2 \\ \vdots & \vdots \\ 1 & -P_k \end{pmatrix} \begin{pmatrix} b \\ \hat{m} \end{pmatrix} = \begin{pmatrix} \log\left(\max_{\mathbf{p}\in\Lambda^{(k)}:\|\mathbf{p}\|_1=1} |c_{\mathbf{p}}|\right) \\ \log\left(\max_{\mathbf{p}\in\Lambda^{(k)}:\|\mathbf{p}\|_1=2} |c_{\mathbf{p}}|\right) \\ \vdots \\ \log\left(\max_{\mathbf{p}\in\Lambda^{(k)}:\|\mathbf{p}\|_1=P_k} |c_{\mathbf{p}}|\right) \end{pmatrix}, \tag{47}$$

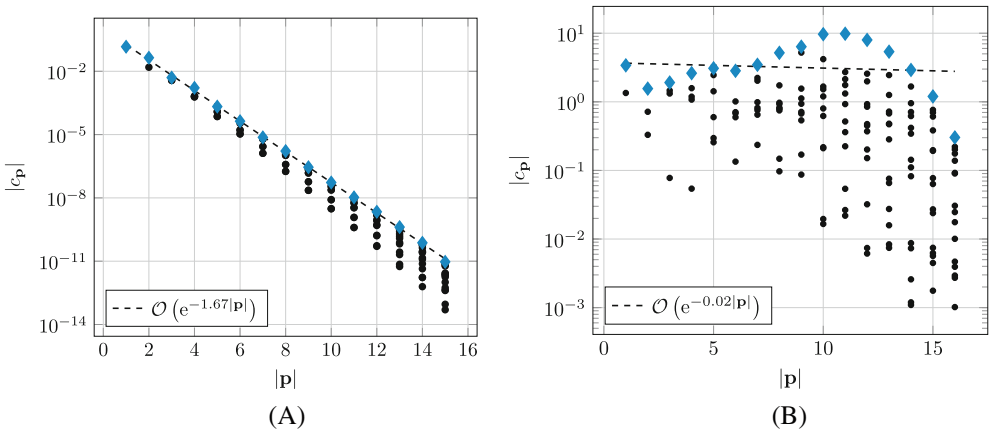

FIGURE 3 Estimating the decay of the PCE coefficients, (A) for a smooth objective function (B), for a function with reduced regularity. The blue markers show the coefficients employed to identify the rate of convergence, whereas the black dots show the remaining PCE coefficients.

in the least-squares sense and determined with $\hat{\theta} = \mathrm{e}^{-\hat{m}}$. The decision for p- or h-refinement is then made as follows:

$$\begin{aligned} &p\text{-refinement for } \ \hat{\theta} \leq \theta_2, \\ &h\text{-refinement for } \ \hat{\theta} > \theta_2, \end{aligned} \tag{48}$$

where θ_2 is a user-defined parameter. A discussion on the choice of θ_2 is available in Section 5.2.1.

p-refinement

In the case of p-refinement, the existing polynomial basis in an element d_k is extended with additional terms. If a TD basis is employed, the total degree P_k is increased by one order and the Leja nodes corresponding to the newly added multi-indices are added to the interpolation grid.

In the case of an anisotropic basis constructed with a dimension-adaptive algorithm, special treatment is necessary. The dimension-adaptive algorithm works sequentially, that is, one node is added to the basis representation in each iteration. Hence, the entire refinement cycle, see Section 4.2, could be estimated after each iteration. However, the improvement of the basis representation by adding one node is expected to be only minor and is in contrast to the computational demand of the entire refinement cycle, for example, computing the PCE coefficients. To avoid unnecessary computational costs, a larger number of nodes is added before we carry on with the next step in the refinement cycle. Following the definition of the corresponding error indicator (43), we estimate again the TD P_k with the given cardinality $\#\Lambda$. Then, $m_k = \binom{N+P_k+1}{N} - \binom{N+P_k}{N}$ basis terms are added, that is, the same number of basis terms as in the TD case. Note that the number of basis terms might be slightly larger than m_k, since the algorithm explores all admissible multi-indices if a multi-index is added to the current set $\Lambda^{(k)}$.

In both cases, the already existing Leja grid points along with the corresponding function evaluations are re-used, hence, the original model must be evaluated only for the newly added Leja nodes.

h-refinement

In the case of h-refinement, for each parameter $X_n^{(k)}$, $n = 1, \ldots, N$, in the element d_k, we estimate the main-effect Sobol sensitivity index[77-79] using the local PCE, such that

$$\tilde{S}_n^{(k),\mathrm{M}} = \frac{1}{\sigma_k^2} \sum_{\mathbf{p} \in \Lambda_n^{(k),\mathrm{M}}} \left(c_{\mathbf{p}}^{(k)}\right)^2, \quad n = 1, \ldots, N, \tag{49}$$

where

$$\Lambda_n^{(k),\mathrm{M}} = \left\{ \mathbf{p} \in \Lambda^{(k)} \,:\, p_n \neq 0 \ \text{ and } \ p_m = 0, \ m \neq n \right\}. \tag{50}$$

The parameter that corresponds to the largest sensitivity index is then selected, that is, the parameter X_ν with

$$\nu = \underset{n\in\{1,\dots,N\}}{\arg\max}\, \tilde{S}_n^{(k),\mathrm{M}}, \tag{51}$$

and the corresponding univariate element $d_\nu^{(k)} = \left[a_\nu^{(k)}, b_\nu^{(k)}\right)$ is split into a "left" and a "right" element, $d_\nu^{(k),\mathrm{L}} = \left[a_\nu^{(k)}, c_\nu^{(k)}\right)$ and $d_\nu^{(k),\mathrm{R}} = \left[c_\nu^{(k)}, b_\nu^{(k)}\right)$, respectively, where the value of $c_\nu^{(k)}$ is chosen such that the two new elements have an equal probability mass, that is,

$$\int_{a_\nu^{(k)}}^{c_\nu^{(k)}} \pi_{X_\nu}\left(x_\nu^{(k)}\right) \varrho_k \, \mathrm{d}x_\nu^{(k)} = \frac{1}{2}. \tag{52}$$

Note that if $b_\nu^{(k)} = b_\nu$, then $d_\nu^{(k)} = \left[a_\nu^{(k)}, b_\nu^{(k)}\right]$. Accordingly, the univariate interpolation grid $\mathcal{X}_{\nu,j_\nu}^{(k)} = \left\{x_{\nu,0}^{(k)}, x_{\nu,1}^{(k)}, \dots, x_{\nu,j_\nu}^{(k)}\right\}$ that exists in the univariate element $d_\nu^{(k)}$ is split into a left grid $\mathcal{X}_{\nu,j_{\nu_\mathrm{L}}}^{(k),\mathrm{L}}$ and a right grid $\mathcal{X}_{\nu,j_{\nu_\mathrm{R}}}^{(k),\mathrm{R}}$, such that $\mathcal{X}_{\nu,j_\nu}^{(k)} = \mathcal{X}_{\nu,j_{\nu_\mathrm{L}}}^{(k),\mathrm{L}} \cup \mathcal{X}_{\nu,j_{\nu_\mathrm{R}}}^{(k),\mathrm{R}}$ and $\mathcal{X}_{\nu,j_{\nu_\mathrm{L}}}^{(k),\mathrm{L}} \cap \mathcal{X}_{\nu,j_{\nu_\mathrm{R}}}^{(k),\mathrm{R}} = \emptyset$. For $n \neq \nu$, the univariate elements $d_n^{(k)}$ and the corresponding univariate interpolation grids $\mathcal{X}_{n,j_n}^{(k)}$ remain unaffected. Therefore, the element d_k is also split into two new elements

$$d_k^{\mathrm{L}} = d_1^{(k)} \times d_2^{(k)} \times \cdots d_\nu^{(k),\mathrm{L}} \times \cdots \times d_N^{(k)}, \tag{53a}$$

$$d_k^{\mathrm{R}} = d_1^{(k)} \times d_2^{(k)} \times \cdots d_\nu^{(k),\mathrm{R}} \times \cdots \times d_N^{(k)}. \tag{53b}$$

We now shift our attention to $d_\nu^{(k)} = d_\nu^{(k),\mathrm{L}} \cup d_\nu^{(k),\mathrm{R}}$, since it is the only univariate element affected by the refinement of the element d_k. It can easily be observed that the interpolation grids $\mathcal{X}_{\nu,j_{\nu,\mathrm{L}}}^{(k),\mathrm{L}}$ and $\mathcal{X}_{\nu,j_{\nu,\mathrm{R}}}^{(k),\mathrm{R}}$ that result from splitting the original element, do not coincide with the Leja sequences that would be constructed in the elements $d_\nu^{(k),\mathrm{L}}$ and $d_\nu^{(k),\mathrm{R}}$, respectively, using the Leja sequence definition in (10) or (11). Therefore, the attractive properties of Leja sequences are mostly lost, especially their suitability as interpolation grids due to the subexponential growth of the Lebesgue constant.[49] Nevertheless, interpolation stability can be recovered by using the existing points to initialize a Leja sequence and then add new Leja points computed as in the definitions (10) and (11), a procedure called *stabilization*.[43] Additionally, the stabilized interpolation grid is sorted according to the Leja ordering, which is necessary to achieve improved Lebesgue constant values.[43,46]

The importance of sorting can also be understood in the case of a uniform distribution, where (32) is recast as

$$x_{n,j_n}^{(k)} = \underset{x\in d_n^{(k)}}{\arg\max} \prod_{i=0}^{j_n-1} \left|x - x_{n,i}^{(k)}\right| = \underset{x\in d_n^{(k)}}{\arg\max} |\det V(x_{n,0}, \dots, x_{n,j_n-1}, x; h_n^0, \dots, h_n^{j_n})|, \tag{54}$$

that is, as determinant maximization over the Vandermonde-like matrix with points $(x_{n,0}, \dots, x_{n,j_n-1}, x)$ and polynomials $\{h_n^0, \dots, h_n^{j_n}\}$. The connection of this greedy maximization procedure to D-optimal experimental design has already been observed.[28] Moreover, it has been shown that the Leja-sorting procedure is equivalent to a partial pivoting LU-decomposition applied to the Vandermonde-like matrix, which can also be used to efficiently compute the Leja sequence.[29]

To the knowledge of the authors, there is no theoretical guarantee regarding the additional Leja nodes that are necessary to achieve a stable Lebesgue constant, respectively, to reduce the interpolation error to a certain level. Numerical tests in a prior work,[43] as well as our own numerical results, indicate that doubling the number of existing nodes leads to a significant improvement in both Lebesgue constant and interpolation accuracy. Note that for symmetric parameter distributions, Leja points are almost symmetrically distributed. Therefore, after splitting, approximately half of the Leja points are available in each new element, see Figure 4A. Hence, to obtain a TD basis of the same polynomial degree, the same number of additional stabilization points must be computed. Considering that, we ensure that the number of Leja nodes is doubled for both TD and dimension-adaptive bases. Now, the multivariate interpolation grid is constructed with

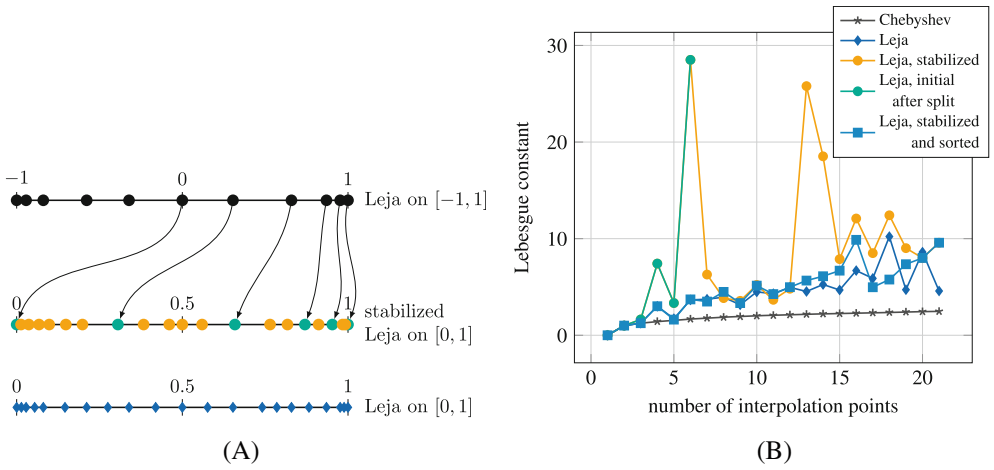

FIGURE 4 (A) Domain splitting and stabilization of the Leja points. The black points have been calculated for a uniform RV $X \sim \mathcal{U}[-1,1]$. The green points show the readily available interpolation nodes within $[0,1]$ after splitting the initial image space $[-1,1]$, whereas the orange points are the new points which are calculated for the uniform RV $X \sim \mathcal{U}[0,1]$ by considering the available (green) points as initialization for the Leja sequence. For comparison, the Leja sequence calculated for $X \sim \mathcal{U}[0,1]$ without considering the readily available nodes are shown in blue. (B) Lebesgue constant for different interpolation grids.

the desired basis representation and reads

$$\mathcal{X}^{(k),\mathrm{L}}_{\Lambda^{(k)}} = \bigcup_{\mathbf{j}\in\Lambda^{(k)}} \left(\mathcal{X}^{(k)}_{1,j_1} \times \mathcal{X}^{(k)}_{2,j_2} \times \cdots \times \mathcal{X}^{(k),\mathrm{L}}_{\nu,j_{\nu_\mathrm{L}}} \times \cdots \times \mathcal{X}^{(k)}_{N,j_N} \right), \tag{55a}$$

$$\mathcal{X}^{(k),\mathrm{R}}_{\Lambda^{(k)}} = \bigcup_{\mathbf{j}\in\Lambda^{(k)}} \left(\mathcal{X}^{(k)}_{1,j_1} \times \mathcal{X}^{(k)}_{2,j_2} \times \cdots \times \mathcal{X}^{(k),\mathrm{R}}_{\nu,j_{\nu_\mathrm{R}}} \times \cdots \times \mathcal{X}^{(k)}_{N,j_N} \right). \tag{55b}$$

Considering the case of a TD basis, all nodes are re-used by construction if we employ the multivariate grid $\mathcal{X}^{(k)}_{\Lambda^{(k)}}$ for interpolation. However, this is not true if the univariate grid $\mathcal{X}^{(k)}_{\nu,j_\nu}$ is sorted. Then, the grids $\mathcal{X}^{(k)}_{\Lambda^{(k)}} \cap \mathcal{X}^{(k),\mathrm{sorted}}_{\Lambda^{(k)}} \neq \emptyset$, but $\mathcal{X}^{(k)}_{\Lambda^{(k)}} \not\subset \mathcal{X}^{(k),\mathrm{sorted}}_{\Lambda^{(k)}}$. Depending on the sorting, several nodes and function evaluations may be lost. However, the numerical results on various examples showed that a significant amount of nodes is re-used.

The dimension-adaptive basis is once more a special case. After h-refinement, the new elements most likely feature a different anisotropic behavior than the original element. Therefore, the multi-index sets in the new elements are re-initialized with the multi-index set $\Lambda^{(k)} = \{\mathbf{0}\}$. The dimension-adaptive algorithm then utilizes the old univariate grids and builds the multivariate grid on demand. The grid defined in (55) then acts as a depot, where Leja nodes are picked if the adaptive Leja algorithm demands for the corresponding nodes. The univariate grids are only extended if all already existing nodes have been utilized. The different anisotropy may lead to the case where a new dimension is now dominant, which then leads to less re-use of Leja nodes. However, the dimension-adaptive basis exploits the anisotropic behavior in the new elements and is expected to be beneficial even if not all nodes are preserved.

The stabilization procedure and its relation to the Lebesgue constant is illustrated in Figure 4. We consider an initial domain $[-1,1]$ with a corresponding Leja sequence consisting of eleven points, which is split into $[-1,0)$ and $[0,1]$, and focus on the latter subdomain. Figure 4A shows the initial Leja sequence (black nodes) and the six points that are re-used within subdomain $[0,1]$ (green nodes). The re-used points are used to initialize a Leja sequence and compute new points (orange nodes) until the interpolation grid is stabilized. For comparison, the Leja points calculated in the interval $[0,1]$ without prior initialization are shown (blue nodes), where it is obvious that they do not coincide with the points of the stabilized Leja grid. Figure 4B shows the Lebesgue constant for an increasing number of interpolation points for the stabilized grid in $[0,1]$, where the initial six points (denoted with green) are obtained from the readily available Leja sequence within $[-1,1]$ prior to splitting. For comparison, the Lebesgue constant for Chebyshev points and for Leja points without prior initialization within $[0,1]$ are shown. It can easily be observed that adding new nodes according to the Leja sequence definition results in the reduction and stabilization of the Lebesgue constant. It is also clear that sorting the stabilized grid according to the Leja ordering is crucial for interpolation stability.

5 | NUMERICAL EXPERIMENTS

In the following, the performance and approximation capabilities of the proposed *hp*-adaptive stochastic collocation method is tested on several test cases. A rigorous discussion on the *h*-convergence of the multi-element collocation method, in particular with isotropic and equidistant grids, is available in the literature.[16] Since the main advantage of using Leja sequences is the ability to re-use readily available Leja points and thus significantly reduce the computational cost due to the model evaluations on the collocation points, we focus on the computational gains provided by the *h*-adaptive and *hp*-adaptive refinement strategies described in Section 4. In particular, we first consider fixed polynomial bases, thus disregarding *p*-adaptivity, and compare the *h*-adaptive multi-element stochastic approach developed in this work against other *h*-adaptive methods found in the literature.[10,15,16] Unless stated otherwise, a discrete root mean square (RMS) error is used to evaluate the convergence behavior of the tested methods. The error is given by

$$\epsilon_{\text{RMS}} = \sqrt{\mathbb{E}_Q\left[(\tilde{g}(\mathbf{x}) - g(\mathbf{x}))^2\right]} = \sqrt{\frac{1}{Q}\sum_{q=1}^{Q}\left(\tilde{g}(\mathbf{x}_q) - g(\mathbf{x}_q)\right)^2}, \tag{56}$$

which is computed using a validation sample $\{\mathbf{x}_q\}_{q=1}^{Q}$ drawn randomly from the joint input PDF.

5.1 | *h*-refinement test cases

5.1.1 | Benchmark problems

In this test case, we discuss the cost savings that are obtained when re-using the Leja points in the context of *h*-refinement. Note that, due to the nestedness of the Leja nodes, all Leja nodes are re-used when *p*-refinement is performed. We consider the continuous and discontinuous Genz functions[80]

$$g_1(\mathbf{x}) = \exp\left(-\sum_{n=1}^{N} 10 \cdot 2^{-n} \left|x_n - 0.5\right|\right), \tag{57a}$$

$$g_2(\mathbf{x}) = \begin{cases} 0, & \text{if } x_1 > 0.5 \text{ or } x_2 > 0.5, \\ \exp\left(\sum_{n=1}^{N} 10 \cdot 2^{-n} x_n\right), & \text{otherwise}, \end{cases} \tag{57b}$$

where the coefficients are chosen such that an anisotropic behavior in the function response is achieved. In the following, x_n denotes realizations of the RVs $X_n \sim \mathcal{U}[0,1]$. The *h*-convergence is examined for varying values of the marking parameter θ_1. Additionally, a comparison against the same methodology, that is, also using *h*-adaptivity only, but based on Gauss–Legendre nodes is performed. In the latter case, the nodes cannot be re-used after *h*-refinement. A further comparison is performed against spatially adaptive sparse grids,[10] where the latter is based on the implementation provided by the SG++ toolbox*. All results have been computed with a validation set of $Q = 10^6$.

First, we consider only the discontinuous Genz function $g_2(\mathbf{x})$ for $N = 2$. For Leja and Gauss–Legendre nodes, a TD polynomial basis with maximum polynomial degrees $P_k = 4$ in each element is used. Figure 5A shows the RMS error (56) in dependence to the number of function calls after the *h*-refinement procedure has converged for a given value of the marking parameter θ_1, with $\theta_1 \in \{10^{-1}, 10^{-3}, 10^{-5}, 10^{-6}, 10^{-7}\}$. The amount of re-used function evaluations is shown next to the marks for the case where nodes are re-used and sorted. As can be observed, the Leja based stochastic collocation method outperforms the Gauss–Legendre based one, irrespective of whether the Leja nodes are re-used, re-used but not sorted, or not re-used. By re-using the Leja nodes, an additional gain in performance is obtained. For the given example, not sorting the Leja nodes leads to a negligible improvement in the RMS error. As mentioned in Section 4.2.4, 100% of the Leja nodes are re-used after element splitting if the nodes are not sorted. However, without sorting the nodes, it is unclear if the Lebesgue constant remains stable, see Figure 4B. Therefore, for the remaining numerical examples, the Leja nodes will always be sorted.

*https://sgpp.sparsegrids.org/. The adaptivity of the method is controlled through the surpluses of the hierarchical interpolation polynomials.

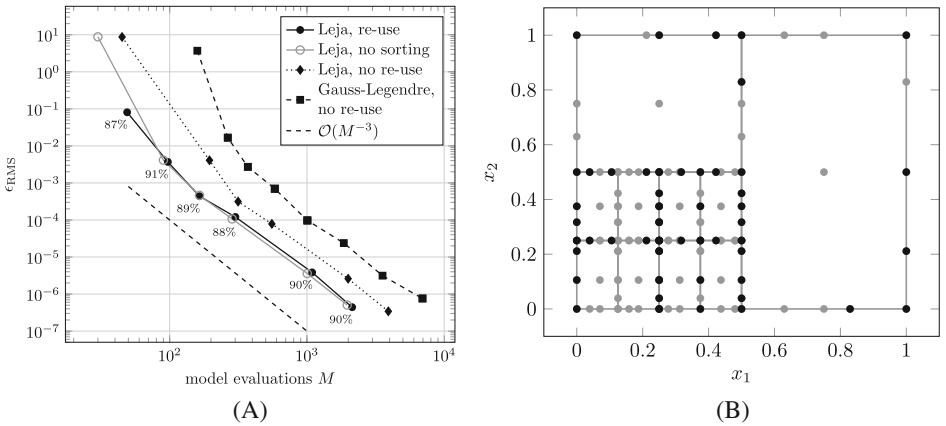

FIGURE 5 (A) RMS error versus model evaluations (M) for the multi-element stochastic collocation method based on Leja points with and without information re-use, as well as with Gauss–Legendre nodes. The percentage of re-used Leja nodes is shown next to the mark. (B) Leja interpolation grid after the convergence of the h-refinement procedure for $\theta_1 = 10^{-4}$. The gray dots show the nodes that have been newly computed in the last iteration of the algorithm. The black dots show the nodes that have been re-used.

All employed approaches reach an asymptotic h-convergence rate of $\mathcal{O}(M^{-3})$ once the discontinuity of the function has been isolated. In the case of sorted Leja nodes, a re-use rate of approximately 90% is obtained. Figure 5B shows the interpolation grid and the corresponding Leja nodes after the h-adaptive algorithm has converged for $\theta_1 = 10^{-4}$. The gray dots show the new nodes that have been computed latest by the algorithm, whereas the black dots show the nodes that have been re-used. It is clearly visible that the adaptive h-refinement results in element splitting only in the regions of the parameter domain where the objective function is not zero, thus verifying that the h-refinement strategy and the corresponding refinement criteria described in Section 4.2 perform as expected. Note that, in this case, the discontinuity of the objective function coincides with the parameter axes. Therefore, the elements can resolve the discontinuity exactly. If this is not the case, a grid with dense elements around the discontinuity will appear and the convergence rate of the RMS error is hindered.[16] The algorithm stops once the criterion (51) is not met. This happens if the error indicator (40) and/or the elements are sufficiently small, equivalently, ϱ_k is sufficiently small.

In the following we discuss the remaining results for both test functions (57) for $N = 2$ and $N = 5$. Figure 6 shows the RMS error over function evaluations calculated with the h-adaptive algorithm based on Leja and Gauss–Legendre nodes. The TD basis is constructed for a maximum polynomial degree $P_k = 4$. Furthermore, results computed with the SG++ toolbox for both test functions are shown. Irrespective of the utilized nodes, the h-adaptive algorithm shows a similar convergence behavior for all considered functions. The results show clearly that re-using the Leja nodes leads to a gain in the approximation accuracy. In the continuous case, the SG++ is superior for $N = 5$, but inferior for $N = 2$. This can be attributed to the underlying tensor product structure of the proposed multi-element approach, which is not inherent in the spatially adaptive sparse-grids approach. Contrarily, the convergence of the SG++ method is hindered when considering the discontinuous example. This behavior is consistent with previous results[10] and is related to the regularity conditions on the objective function that are here violated.

5.2 | *hp*-refinement test cases

5.2.1 | Introductory example

To showcase the computational benefits obtained due to the *hp*-adaptivity of the proposed method, an academic test case featuring a one-dimensional function is considered first. In particular, the analytical function

$$g(x) = -x + 0.1\sin(30x) + \mathrm{e}^{-(50(x-0.65))^2}, \tag{58}$$

is employed, which has previously been used to validate the SSE method.[81] The function is depicted in Figure 7A, where the values of the parameter x refer to realizations of the uniform RV $X \sim \mathcal{U}[0, 1]$. The observed sharp spike is the main

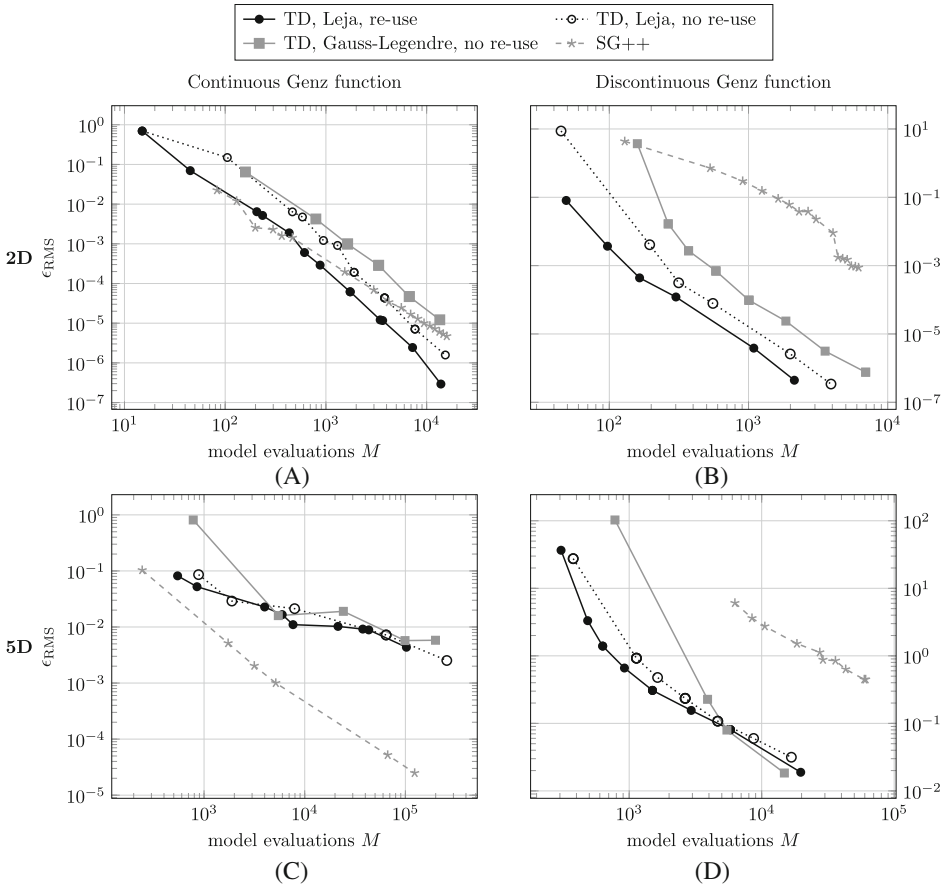

FIGURE 6 RMS error over model evaluations M. The results have been computed with adaptive h-refinement. Left column: Continuous Genz function $g_1(\mathbf{x})$, right column: Discontinuous Genz function $g_2(\mathbf{x})$, top row: $N = 2$, bottom row: $N = 5$.

reason why a rather slow convergence rate is to be expected if a global approximation is applied, for example, a global interpolation on Leja or Chebyshev nodes. The *hp*-adaptive Leja based multi-element collocation method should significantly improve the convergence rate by dividing the domain into smooth sub-domains (h-refinement), in which local polynomial approximations with increasing degree are developed (p-refinement), as presented in Section 4.

Following the SSE paper,[81] the accuracy of a surrogate model $\tilde{g}(x) \approx g(x)$ is evaluated using the relative root mean squared error

$$\epsilon_{\text{RMS,rel}} = \sqrt{\frac{\mathbb{E}\left[(g(x) - \tilde{g}(x))^2\right]}{\mathbb{V}\left[g(x)\right]}}, \tag{59}$$

which is estimated using a random sample with 10^5 points. The results for different surrogate models are shown in Figure 7A. The results for the *hp*-adaptive Leja based multi-element collocation method are obtained with $\theta_1 = 10^{-12}$ and $\theta_2 = 0.8$. Using h-refinement alone and a constant polynomial degree $P_k = 3$ leads to a convergence rate of $\mathcal{O}(M^{-4})$. For comparison, the results obtained with p-refined global interpolations on Leja and Chebyshev nodes, as well as with the SSE method[81] are presented.

All proposed methods show a converging behavior, however, it is clearly evident that the *hp*-adaptive multi-element collocation method outperforms all other surrogate modeling options. Based on a least-squares estimate on the decay of the error for the *hp*-adaptive approach, a geometric convergence rate of $\mathcal{O}\left(\mathrm{e}^{-0.32M}\right)$ is obtained.[82] Note that the *hp*-adaptive Leja based multi-element stochastic collocation method and the SSE method tackle different types of problems. The method proposed in this work handles problems with low to moderate dimensions, whereas the SSE method is intended for high dimensional problems. Therefore, the comparison in Figure 7A is not to be understood in a competing manner.

Next we analyze the impact of the parameters θ_1 and θ_2 on the *hp*-refinement procedure. Figure 8 shows the domain decomposition and the polynomial degree of each element for different values of θ_1 and θ_2. The left column figures

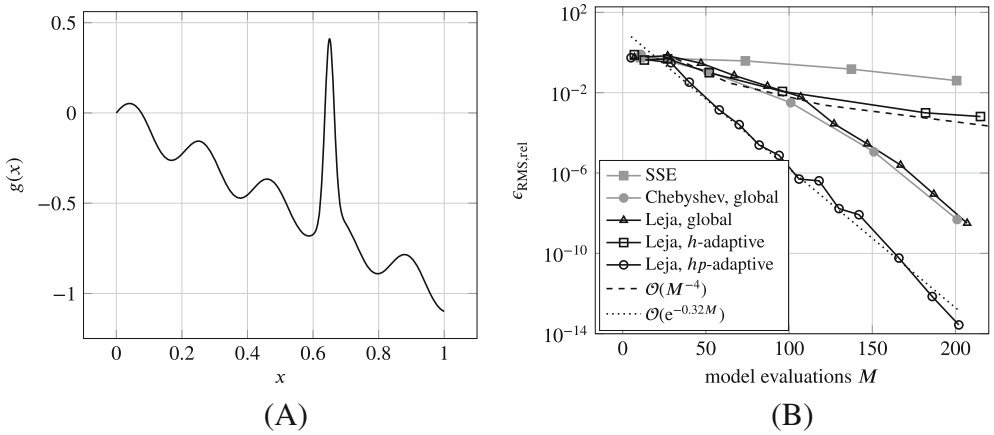

FIGURE 7 (A) One-dimensional analytical function. (B) Relative mean square error η in dependence of the number of model evaluations for SSE, global Chebyshev/Leja, and *hp*-adaptive Leja surrogates.

correspond to $\theta_1 = 10^{-2}$, while the right column figures to $\theta_1 = 10^{-4}$. Accordingly, the top row figures correspond to $\theta_2 = 0.2$, in which case *h*-refinement is preferred over *p*-refinement. Contrarily, *p*-refinement is prioritized in the bottom row figures, which correspond to $\theta_2 = 0.8$. An initial polynomial degree of $P_k = 2$ has been used for all four cases. All four choices for θ_1 and θ_2 generate interpolation grids that isolate the spike. As would be expected, more elements are generated for $\theta_2 = 0.2$, in particular in the region where the spike is observed. Further decreasing θ_1 leads to a more finely resolved spike area, where the polynomial degrees in each element remain at moderate values. As would also be expected, larger elements with higher polynomial degrees are generated for $\theta_2 = 0.8$. If θ_1 is further decreased in this case, *p*-refinement with a focus on the spike region is carried out.

Remark: In the following we discuss the choice of the constants θ_1 and θ_2. The marking parameter θ_1 determines the accuracy of the solution. However, this accuracy is estimated either with variance-based error indicators or, if available, with the adjoint-based error indicator (38), see Section 4.2.2. Parameter θ_1 is chosen according to the expected behavior of the error indicator. Numerical tests have shown that the variance-based error indicator is often too optimistic, that is, if a desired accuracy needs to be reached, a smaller value for θ_1 is necessary. In case of a fixed computational budget, θ_1 can be decreased sequentially until the budget is reached. The constant θ_2 is chosen according to the objective function. If we expect the function to have areas that are not easily resolvable, *h*-refinement is preferred and a smaller constant θ_2 is chosen. Contrarily, if the function has areas that can easily be resolved or the function is analytic but has sharp transitions, see for example the function (58), a larger θ_2 should be chosen. In all considered numerical examples, the value of θ_2 is chosen such that $\theta_2 \in [0, 1]$.

5.2.2 | Benchmark problems

The full capabilities of the proposed method, that is, *hp*-adaptive refinement with different basis representations, is now discussed for the case of the Genz test functions (57). To assess the performance of the proposed method, we compare it against a spatially adaptive sparse grid combination technique, in the following referred to as sparseSpACE.[11] Numerical tests are carried out for $N = 2$ and $N = 8$ parameter dimensions. In all cases, the error (56) is determined with a validation set of size $Q = 10^6$.

Figure 9 shows the RMS error for both test functions for the cases $N = 2$ and $N = 8$. First, considering the case of $N = 2$ dimensions, the *hp*-adaptive approach with a TD basis is found to be superior against all other approaches for both test functions. This result is not surprising, due to the fact that the objective functions are almost isotropic for $N = 2$. The results obtained with the sparseSpACE toolbox show only a moderate convergence behavior. Next, we consider the case of $N = 8$ dimensions for the continuous Genz function, see Figure 9C. We observe that all methods show a converging behavior. In this higher dimensional case, the dimension-adaptive basis performs significantly better than the TD basis. This can be attributed to the anisotropic objective function. Nevertheless, the tensor-product structure of the proposed multi-element approach hinders its ability to compete with sparseSpACE in this test case, similar to the results obtained when considering *h*-refinement only, see Section 5.1.1. However, looking at the results obtained for the discontinuous

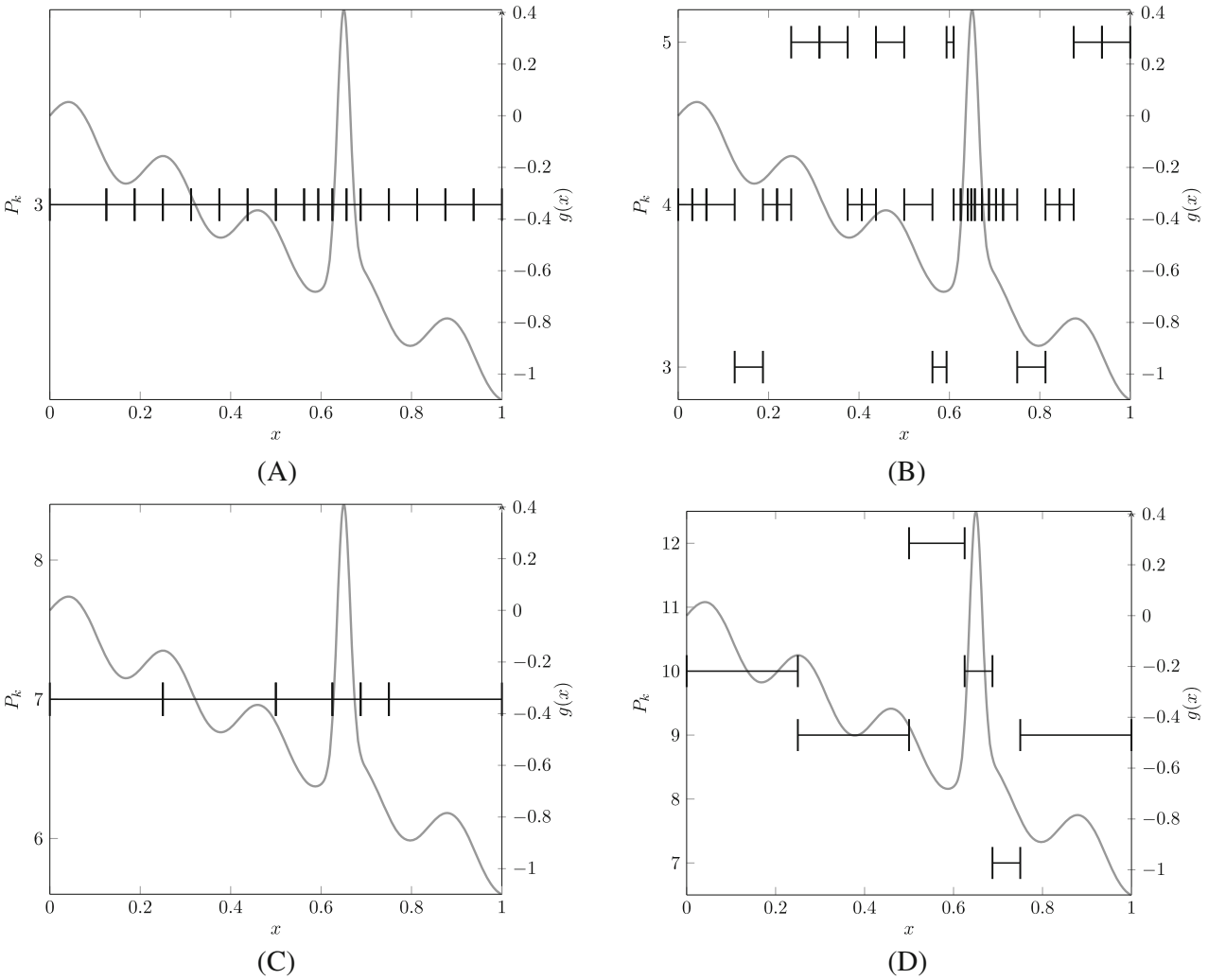

FIGURE 8 Adaptively generated interpolation grids and corresponding polynomial degrees for different values of θ_1 and θ_2. (A) $\theta_1 = 10^{-2}$, $\theta_2 = 0.2$. (B) $\theta_1 = 10^{-4}$, $\theta_2 = 0.2$. (C) $\theta_1 = 10^{-2}$, $\theta_2 = 0.8$. (D) $\theta_1 = 10^{-4}$, $\theta_2 = 0.8$.

Genz function, see Figure 9D, the dimension-adaptive basis performs significantly better. In particular, the TD basis and the dimension-adaptive basis show a similar convergence behavior, whereby the dimension-adaptive basis outperforms the TD basis by two orders of magnitude. In this case, the sparseSpACE toolbox does not converge, which is consistent with previously reported results.[11]

For the discontinuous Genz function (57b), two h-refinement steps are necessary to resolve the discontinuity. All proposed hp-approaches achieve this minimum domain splitting for a certain parameter choice of θ_2. Under this parameter configuration, the hp-adaptive algorithm achieves the best results. The minimum number of h-refinements leads to large elements within which only p-refinement is carried out afterwards. Then, the basis representation in the elements is of even more importance for the computational costs, respectively the accuracy of the approximation.

The relative amount of re-used Leja nodes in case of h-refinement can be found in Table 1. Note that the discontinuous Genz function with $N = 8$ dimensions results in comparatively low re-use percentages, particularly for the TD and dimension-adaptives bases. However, in those cases, the hp-adaptive algorithm performed only two h-refinements and thus the overall costs are dominated by the following p-refinements.

5.2.3 | Posterior estimation

As stressed in Section 1, surrogate models for the likelihood function, respectively for the posterior distribution of inverse problems, can be an alternative to expensive Markov chain Monte Carlo analyses. To showcase the benefits of using the

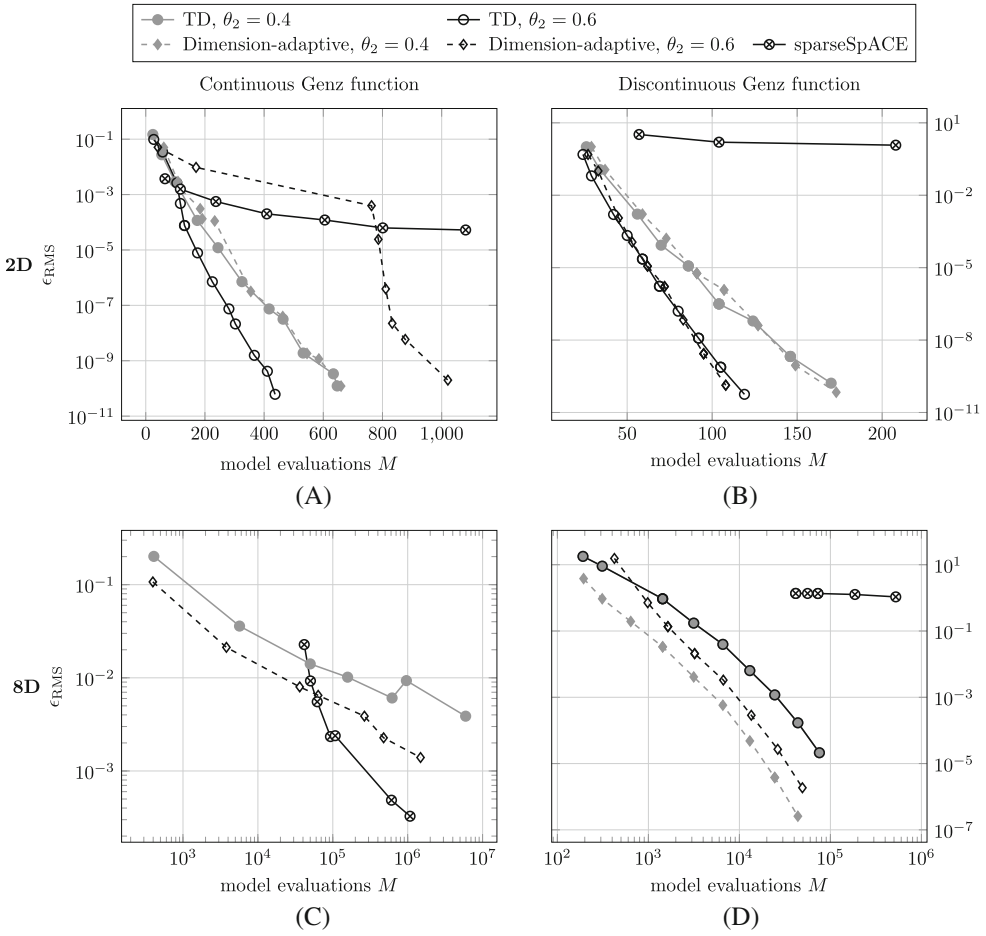

FIGURE 9 RMS error over model evaluations M. The results have been computed with adaptive *hp*-refinement. Left column: Continuous Genz function $g_1(\mathbf{x})$, right column: Discontinuous Genz function $g_2(\mathbf{x})$, top row: $N = 2$, bottom row: $N = 8$.

TABLE 1 Relative amount of re-used Leja nodes over the amount of available Leja nodes.

		Continuous Genz		Discontinuous Genz	
Basis	θ_2	$N = 2$	$N = 8$	$N = 2$	$N = 8$
TD	0.4	93%	93%	89%	38%
	0.6	87%	88%	83%	38%
Dimension-adaptive	0.4	94%	69%	100%	78%
	0.6	66%	86%	58%	50%

Note: Only savings during *h*-refinement are counted. The results were obtained upon convergence of the algorithm.

method proposed in this work, a statistical model with additive noise is considered,[3] which reads

$$\mathcal{G}(\mathbf{x}) + \boldsymbol{\epsilon} = \mathbf{b}, \tag{60}$$

where $\mathbf{b} \in \mathbb{R}^D$ denotes the observations or measurements, $\mathbf{x} \in \mathbb{R}^N$ the unknown model parameters, $\mathcal{G} : \mathbb{R}^N \to \mathbb{R}^D$ the forward model, and $\boldsymbol{\epsilon} \in \mathbb{R}^D$ the Gaussian noise, such that $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2\mathbf{I})$, with $\mathbf{I} \in \mathbb{R}^{D\times D}$ being the identity matrix. In the simplest case, the inverse problem (60) represents a linear regression problem, where $\mathcal{G}$ is the discretized model operator and $\mathbf{x}$ the vector of regression coefficients, accordingly, it holds that $\mathcal{G}(\mathbf{x}) = \mathcal{G}\mathbf{x}$. In the same context, deconvolution problems pose significantly greater challenges.

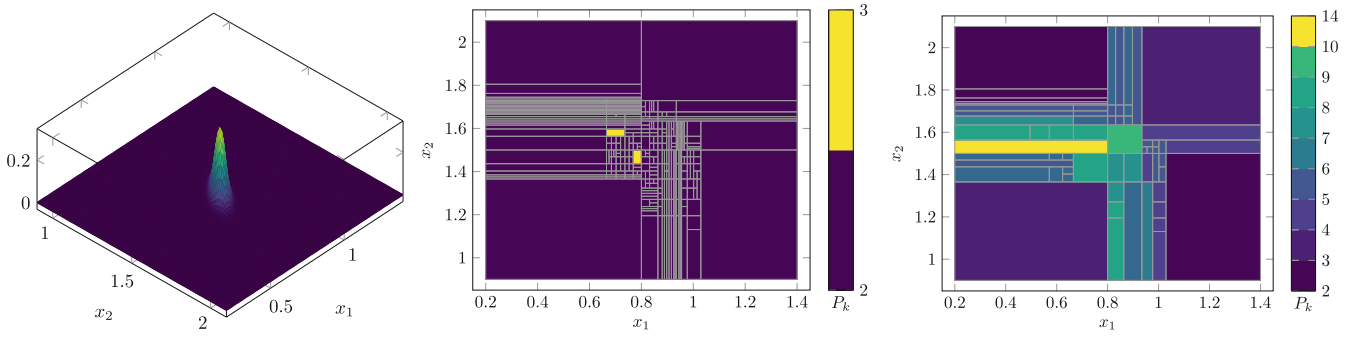

FIGURE 10 (A) Posterior over the parameters x_1 and x_2. (B, C) Adaptively generated interpolation grids and corresponding polynomial degrees for $\theta_1 = 10^{-3}$ and different values of θ_2. (B) $\theta_2 = 0.2$. (C) $\theta_2 = 0.8$.

The unknown parameter vector $\mathbf{x}$ can be computed by means of maximum likelihood estimation (MLE), such that

$$\mathbf{x} = \arg\max_{\mathbf{x}} L(\mathbf{x}|\mathbf{b}) = \arg\max_{\mathbf{x}} \frac{1}{(2\pi)^{D/2}\sigma^D} \exp\left(-\frac{||\mathcal{G}(\mathbf{x}) - \mathbf{b}||^2}{2\sigma^2}\right), \tag{61}$$

where $L(\mathbf{x}|\mathbf{b})$ denotes the likelihood function. However, inverse problems are often ill-posed, for example, due to the amplification of the measurement noise, thus resulting in very challenging parameter estimation tasks. A possible remedy is to regularize problem (60) by employing, for example, Tikhonov regularization or iterative regularization,[3,83] to name but a few options. Another measure is to utilize Bayesian inference.[83] Therein, the parameters $\mathbf{x}$ are modeled as RVs and a prior distribution is assumed for them. Given the observations $\mathbf{b}$, information about the noise, and our prior belief regarding the parameter vector $\mathbf{x}$, Bayes' theorem yields

$$\pi(\mathbf{x}|\mathbf{b}) \propto L(\mathbf{x}|\mathbf{b})p(\mathbf{x}), \tag{62}$$

where $p(\mathbf{x})$ denotes the prior distribution and $\pi(\mathbf{x}|\mathbf{b})$ the posterior distribution. The posterior distribution (62) is now regularized by the prior distribution $p(\mathbf{x})$. The parameter vector can be estimated by maximizing (62), which leads to the so-called maximum a posteriori (MAP) estimation.

Using the Bayesian approach, additional statistical quantities that characterize the posterior distribution, for example, its mean, variance, or higher-order moments, can be computed. In that case, the evidence, also referred to as the marginal likelihood, that is, the normalization of (62) over the parameter vector $\mathbf{x}$, must be computed. The computation of the evidence can be challenging, as the likelihood function and thus also the posterior can be highly concentrated, see Figure 10A. The proposed multi-element collocation method can then be employed to obtain a surrogate model of (62), such that the evidence and the statistical moments can be computed accurately with a comparatively lower computational cost.

Two-dimensional regression problem

We start with a two-dimensional regression test case where we search for the model parameters x_1 and x_2 such that

$$\xi \mapsto G(\xi, x_1, x_2) = x_2 \exp(x_1 \xi) - 2, \tag{63}$$

best fits experimental data. A measurement set with 5 noisy observations is assumed to be available, that is, $\{\xi_i, b_i\}_{i=1}^5$, where ξ_i are equidistantly sampled in $[-1, 1]$ and hence $\mathcal{G}(\mathbf{x}) = (G(\xi_1, x_1, x_2), \dots, G(\xi_5, x_1, x_2))^\top$ and $\mathbf{b} = (b_1, \dots, b_5)^\top$, respectively. Artificial observations are then generated as $b_i = G\left(\xi_i, x_1^\star, x_2^\star\right) + \varepsilon$, where $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ with $\sigma^2 = 0.01$. Note that $x_1^\star$ and $x_2^\star$ denote the true but currently unknown model parameters. We assume the parameters to follow a truncated normal distribution, denoted as $\mathcal{TN}(\mu, \sigma^2, \ell, u)$, where μ and σ^2 are the mean and variance of a Gaussian distribution $\mathcal{N}\left(\mu, \sigma^2\right)$ which is truncated within the interval $[\ell, u]$. The parameter priors are given as $x_1 \sim \mathcal{TN}(0.8, 0.04, 0.2, 1.4)$, $x_2 \sim \mathcal{TN}(1.5, 0.04, 0.9, 2.1)$.

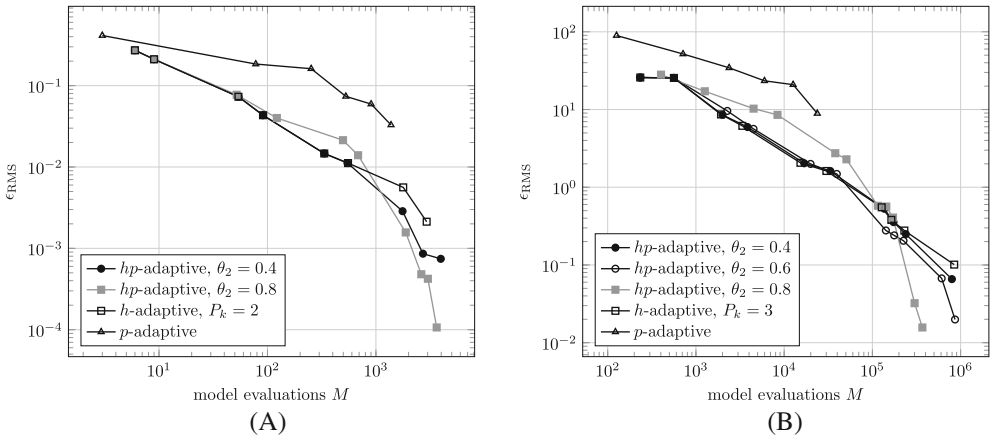

FIGURE 11 Convergence of the RMS error in dependence to model evaluations. The results have been obtained using TD polynomial bases. (A) Two-dimensional regression problem. (B) Four-dimensional deconvolution problem.

TABLE 2 Distribution of the random variables for the 4D Bayesian example.

Parameter	Distribution
x_1	$\mathcal{TN}(0.88, 0.16, -0.32, 2.08)$
x_2	$\mathcal{TN}(0.1, 0.04, -0.5, 0.7)$
x_3	$\mathcal{TN}(0.3, 0.01, 0, 0.6)$
x_4	$\mathcal{TN}(0.29, 0.01, -0.01, 0.59)$

The posterior distribution is depicted in Figure 10A, which shows the strongly concentrated posterior density. Figure 11A shows the RMS error versus the number of model evaluations using stochastic collocation on Leja interpolation grids based on h-refinement, global interpolation with p-refinement, and hp-refinement with a varying adaptivity parameter θ_2. All local polynomial approximations are based on TD bases. Except for p-refinement, all marks show the results after convergence for a specific marking parameter θ_1. The results clearly show that surrogates based on h- and hp-refinement are superior to the global collocation approach. For a moderate number of model evaluations, a similar performance is observed using either h- or hp-refinement. However, once the parameter domain has been sufficiently decomposed, in which case the hp-adaptive algorithm prefers p- over h-refinement, hp-adaptivity yields significant computational gains. Additionally, at this point, a switch from an algebraic to a geometric convergence rate is observed.

Figure 10B,C show the domain decomposition as well as the employed polynomial degrees P_k of the local TD bases, for $\theta_2 = 0.2$ and $\theta_2 = 0.8$, respectively. The results correspond to the converged algorithm for a marking parameter $\theta_1 = 10^{-3}$. Both figures clearly illustrate that the hp-adaptive approach identifies the highly concentrated posterior density. As expected, in the case of $\theta_2 = 0.2$, more elements are spent to isolate the posterior density, while the local polynomial degrees P_k have low values. Contrarily, choosing $\theta_2 = 0.8$ results in fewer elements with significantly higher local polynomial degrees.

Four-dimensional deconvolution problem

We now consider a posterior estimation problem featuring four parameters. We again solve the inverse problem (60) for the parameter vector $\mathbf{x}$ and given observations $\mathbf{b}$, however, the underlying problem is now a convolution problem of the form

$$b(t) = \int_{-\infty}^{\infty} K\left(t - t'\right) x\left(t'\right) \, \mathrm{d}t', \quad -\infty \leq t \leq \infty, \tag{64}$$

where $b(t)$ is the output function, $K(t)$ is the weighting function or kernel, and $x(t)$ the input function. For example, in the field of acoustics, $b(t)$ is the system response, $K(t)$ the fundamental solution of the wave equation, and $x(t)$ the sources. The example can be found in the book of Bardsley,[3] where a detailed description of the functions and parameters is given.

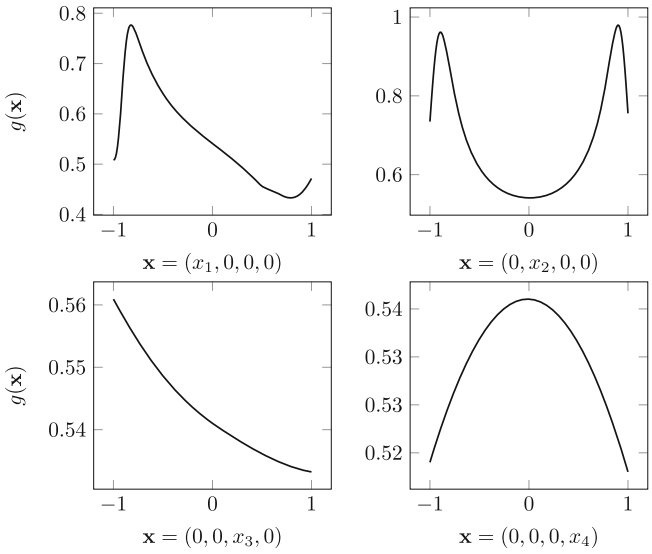

FIGURE 12 Snapshots of the quantity of interest for the Helmholtz problem for $N = 4$.

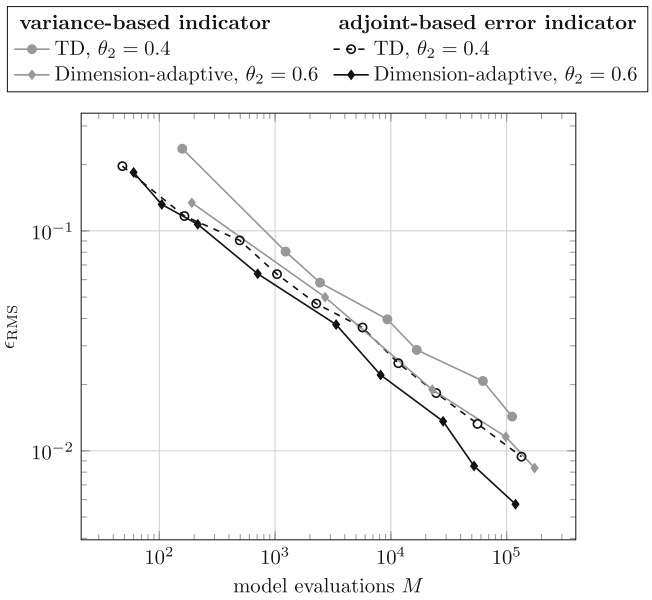

FIGURE 13 RMS error over model evaluations M for the Helmholtz problem with $N = 4$.

A MATLAB implementation is also available.[84] The convolution problem is discretized with 4 grid points, which leads to the discrete system (60). At each of the grid points, the unknown input values $x_1, \ldots, x_4$, are to be estimated based on the available data $\mathbf{b}$. Truncated normal distributions are considered as priors for the unknown input values, see Table 2. Furthermore, the measurements $\mathbf{b}$ are polluted by additive Gaussian noise with $\sigma = 0.1$. The other parameters remain as in the book of Bardsley.[3]

Figure 11B shows the convergence of the RMS error over the number of model evaluations M. Similar to the two-dimensional case, all local polynomial approximations employ TD bases. As previously observed, once the concentrated posterior density has been sufficiently resolved by the algorithm, the collocation based on *hp*-refinement outperforms *h*-refinement alone. Moreover, it must be noted that, in this test case, the computational cost for constructing the Leja grid in the global *p*-refinement case, cannot be regarded to be negligible anymore due to the increasing costs of resolving the optimization problem (11). Therefore, the *hp*-adaptive stochastic collocation is to be preferred overall.

Remark: In both test cases presented in this section, TD bases have been employed for the polynomial approximations. The reason for this choice is that the error indicator (43) employed for the dimension-adaptive basis expansion fails to

identify the elements that need further h- or p-refinement. This issue can be attributed to the high concentration of the posterior density, in combination with the greedy algorithm of the dimension-adaptive basis approach. Nevertheless, other sparse bases such as (non-adaptive) Smolyak sparse grids[16,85] can be used instead. In that case, the benefits of using Leja nodes, in particular the ability to re-use existing nodes and corresponding model evaluations, remain valid.

5.2.4 | Helmholtz problem

To illustrate the method with a PDE based problem, we consider the Helmholtz equation introduced in Section 2.3. For our numerical experiments on the Helmholtz equation we choose $N = 4$ terms for the log-Karhunen–Loève expansion (6). One-dimensional snapshots of the QoI are depicted in Figure 12. We assume that input parameters are distributed as $X_n \sim \mathcal{B}(2, 2, -1, 1),\ n = 1, \dots, N$, where $\mathcal{B}(\alpha, \beta, a, b)$ denotes the Beta distribution with shape parameters $\alpha,\ \beta$ and image space $[a, b]$. The RMS error is computed with a validation set of $Q = 10^5$. Figure 13 compares the results that are obtained for the variance-based error indicator and the adjoint-based error indicator. The adjoint-based error indicator behaves as expected and provides a better approximation of the actual error. Consequently a higher accuracy for a fixed computational budget is achieved if the adjoint-based error indicator is utilized.

6 | CONCLUSION

In this work, an hp-adaptive multi-element collocation method based on Leja nodes has been introduced. The presented method allows to re-use already computed model evaluations after h- or p-refinement. The local interpolation grids are stabilized after h-refinement by adding additional nodes and appropriately reordering the Leja sequence. To tackle the computationally demanding case of higher dimensional problems, a dimension-adaptive basis expansion algorithm has been employed. The work considered uniform, truncated normal, and beta distributed random inputs.

The numerical results showed that in the case of h-refinement, the proposed method is superior against the original multi-element approaches that do not utilize information re-use. For low dimensions and functions with discontinuities, the h-adaptive multi-element approach is the method of choice when compared against the spatially adaptive sparse-grids method. In higher dimensional settings with continuous functions, approaches based on spatially adaptive sparse grids are more efficient, because of the limitations of the underlying hypercube structure in higher dimensions. Furthermore, the results also indicated that hp-adaptive refinement is to be preferred over pure h-refinement, especially when higher-dimensional problems are considered. In case of hp-refinement, the proposed method was compared to the generalized spatially adaptive sparse-grids combination technique, which is inferior if the objective function features discontinuities or features low parameter dimensions. Again, the opposite is true for higher dimensional problems with continuous functions. Moreover, the proposed method was capable of approximating a highly concentrated posterior density in a two- and a four-dimensional inverse problem. Finally, the proposed method was applied to a PDE example (Helmholtz equation), where we showed that a deterministic, adjoint-based error indicator is to be preferred, if available.

In summary, we may conclude that the proposed hp-adaptive collocation method based on Leja nodes is a reliable approach for surrogate modeling and UQ for functions with reduced regularity, as well as for functions featuring large gradients or sharp transitions. We expect, that the computational demand of the proposed method can be further reduced when combined with methods that detect the critical areas in the parameter domain beforehand.[14,17,86] Then, a reduced number of h-refinement steps is expected, which in turn should reduce the overall complexity.

FUNDING INFORMATION

This research was supported by the German Research Foundation (DFG) via Grants GRK 2128 (project number: 264883531) and TRR 361 (project number: 492661287).

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

ACKNOWLEDGMENT

Open Access funding enabled and organized by Projekt DEAL.

ORCID

Armin Galetzka https://orcid.org/0000-0003-3444-8471
Ulrich Römer https://orcid.org/0000-0002-1277-7509

REFERENCES

1. Matthies HG. Quantifying uncertainty: modern computational representation of probability and applications. In: Ibrahimbegovic A, Kozar I, eds. *Extreme Man-Made and Natural Hazards in Dynamics of Structures*. Springer; 2007:105-135.
2. Lee SH, Chen W. A comparative study of uncertainty propagation methods for black-box-type problems. *Struct Multidiscipl Optim*. 2009;37(3):239-253.
3. Bardsley JM. *Computational Uncertainty Quantification for Inverse Problems*. Vol 19. SIAM; 2018.
4. Jiang P, Zhou Q, Shao X. *Surrogate Model-Based Engineering Design and Optimization*. Springer; 2020.
5. Koziel S, Leifsson L. *Surrogate-Based Modeling and Optimization*. Springer; 2013.
6. Tripathy RK, Bilionis I. Deep UQ: learning deep neural network surrogate models for high dimensional uncertainty quantification. *J Comput Phys*. 2018;375:565-588.
7. Sacks J, Welch WJ, Mitchell TJ, Wynn HP. Design and analysis of computer experiments. *Stat Sci*. 1989;4(4):409-423.
8. Ghanem R, Spanos P. Polynomial chaos in stochastic finite elements. *J Appl Mech*. 1990;57(1):197.
9. Le Maıtre O, Knio O, Najm H, Ghanem R. Uncertainty propagation using Wiener–Haar expansions. *J Comput Phys*. 2004;197(1):28-57.
10. Pflüger D. *Spatially Adaptive Sparse Grids for High-Dimensional Problems*. PhD Thesis. Technische Universität München; 2010.
11. Obersteiner M, Bungartz HJ. A generalized spatially adaptive sparse grid combination technique with dimension-wise refinement. *SIAM J Sci Comput*. 2021;43(4):A2381-A2403. doi:10.1137/20M1325885
12. Chouvion B, Sarrouy E. Development of error criteria for adaptive multi-element polynomial chaos approaches. *Mech Syst Signal Process*. 2016;66:201-222.
13. Halder YV, Sanderse B, Koren B. An adaptive minimum spanning tree multielement method for uncertainty quantification of smooth and discontinuous responses. *SIAM J Sci Comput*. 2019;41(6):A3624-A3648.
14. Kawai S, Oyama A. Multi-element stochastic Galerkin method based on edge detection for uncertainty quantification of discontinuous responses. *J Verif Valid Uncertain Quantif*. 2020;5(4):041004.
15. Wan X, Karniadakis GE. An adaptive multi-element generalized polynomial chaos method for stochastic differential equations. *J Comput Phys*. 2005;209(2):617-642.
16. Foo J, Wan X, Karniadakis GE. The multi-element probabilistic collocation method (ME-PCM): error analysis and applications. *J Comput Phys*. 2008;227(22):9572-9595.
17. Jakeman JD, Narayan A, Xiu D. Minimal multi-element stochastic collocation for uncertainty quantification of discontinuous functions. *J Comput Phys*. 2013;242:790-808.
18. Pettersson P, Krumscheid S. Adaptive stratified sampling for nonsmooth problems. *Int J Uncertain Quantif*. 2022;12(6):71-99.
19. Agarwal N, Aluru NR. A domain adaptive stochastic collocation approach for analysis of MEMS under uncertainties. *J Comput Phys*. 2009;228(20):7662-7688.
20. Foo J, Karniadakis GE. Multi-element probabilistic collocation method in high dimensions. *J Comput Phys*. 2010;229(5):1536-1557.
21. Fuchs B, Garcke J. Simplex stochastic collocation for piecewise smooth functions with kinks. *Int J Uncertain Quantif*. 2020;10(1):1-24.
22. Ma X, Zabaras N. An adaptive hierarchical sparse grid collocation algorithm for the solution of stochastic differential equations. *J Comput Phys*. 2009;228(8):3084-3113.
23. Witteveen JA, Iaccarino G. Simplex stochastic collocation with ENO-type stencil selection for robust uncertainty quantification. *J Comput Phys*. 2013;239:1-21.
24. Witteveen JA, Iaccarino G. Simplex stochastic collocation with random sampling and extrapolation for nonhypercube probability spaces. *SIAM J Sci Comput*. 2012;34(2):A814-A838.
25. Giovanis D, Shields M. Variance-based simplex stochastic collocation with model order reduction for high-dimensional systems. *Int J Numer Methods Eng*. 2019;117(11):1079-1116.
26. Ainsworth M, Oden JT. A posteriori error estimation in finite element analysis. *Comput Methods Appl Mech Eng*. 1997;142(1-2):1-88.
27. Ge L, Sun T. An adaptive hp-version stochastic Galerkin method for constrained optimal control problem governed by random reaction diffusion equations. *Comput Appl Math*. 2022;41(3):1-30.
28. Narayan A, Jakeman JD. Adaptive Leja sparse grid constructions for stochastic collocation and high-dimensional approximation. *SIAM J Sci Comput*. 2014;36(6):A2952-A2983.
29. Bos L, De Marchi S, Sommariva A, Vianello M. Computing multivariate Fekete and Leja points by numerical linear algebra. *SIAM J Numer Anal*. 2010;48(5):1984-1999.
30. Chkifa A, Cohen A, Schwab C. High-dimensional adaptive sparse polynomial interpolation and applications to parametric PDEs. *Found Comput Math*. 2014;14(4):601-633.
31. Loukrezis D, Römer U, De Gersem H. Assessing the performance of Leja and Clenshaw-Curtis collocation for computational electromagnetics with random input data. *Int J Uncertain Quantif*. 2019;9(1):33-57.
32. Babuška I, Nobile F, Tempone R. A stochastic collocation method for elliptic partial differential equations with random input data. *SIAM Rev*. 2010;52(2):317-355.

33. Xiu D, Hesthaven JS. High-order collocation methods for differential equations with random inputs. *SIAM J Sci Comput*. 2005;27(3):1118-1139.
34. Barthelmann V, Novak E, Ritter K. High dimensional polynomial interpolation on sparse grids. *Adv Comput Math*. 2000;12(4):273-288.
35. Sudret B. Global sensitivity analysis using polynomial chaos expansions. *Reliab Eng Syst Saf*. 2008;93(7):964-979.
36. Marzouk Y, Xiu D. A stochastic collocation approach to Bayesian inference in inverse problems. *Commun Comput Phys*. 2009;6(4):826-847.
37. Wagner PR, Marelli S, Sudret B. Bayesian model inversion using stochastic spectral embedding. *J Comput Phys*. 2021;436:110141.
38. Schillings C, Schwab C. Sparse, adaptive Smolyak quadratures for Bayesian inverse problems. *Inverse Probl*. 2013;29(6):065011.
39. Farcas IG, Latz J, Ullmann E, Neckel T, Bungartz HJ. Multilevel adaptive sparse Leja approximations for Bayesian inverse problems. *SIAM J Sci Comput*. 2020;42(1):A424-A451.
40. Eigel M, Gruhlke R, Marschall M. Low-rank tensor reconstruction of concentrated densities with application to Bayesian inversion. *Stat Comput*. 2022;32(2):1-27.
41. Parno MD, Marzouk YM. Transport map accelerated Markov chain Monte Carlo. *SIAM/ASA J Uncertain Quantif*. 2018;6(2):645-682.
42. Leja F. Sur certaines suites liées aux ensembles plans et leur application à la représentation conforme. *Ann Pol Math*. 1957;4(1):8-13.
43. Bos LP, Caliari M. Application of modified Leja sequences to polynomial interpolation. *Dolomites Res Notes Approx*. 2015;8:66-74.
44. Carnicer JM, Khiar Y, Peña JM. Central orderings for the Newton interpolation formula. *BIT Numer Math*. 2019;59(2):371-386. doi:10.1007/s10543‐018‐00743‐2
45. Carnicer J, Khiar Y, Peña J. Optimal stability of the Lagrange formula and conditioning of the Newton formula. *J Approx Theory*. 2017;238:52-66. doi:10.1016/j.jat.2017.07.005
46. Reichel L. Newton interpolation at Leja points. *BIT*. 1990;30(2):332-346. doi:10.1007/BF02017352
47. Calvi JP, Van MP. On the Lebesgue constant of Leja sequences for the unit disk and its applications to multivariate interpolation. *J Approx Theory*. 2011;163(5):608-622.
48. Jantsch P, Webster CG, Zhang G. On the Lebesgue constant of weighted Leja points for Lagrange interpolation on unbounded domains. *IMA J Numer Anal*. 2019;39(2):1039-1057.
49. Taylor R, Totik V. Lebesgue constants for Leja points. *IMA J Numer Anal*. 2008;30(2):462-486. doi:10.1093/imanum/drn082
50. Jakeman JD, Franzelin F, Narayan A, Eldred M, Pflüger D. Polynomial chaos expansions for dependent random variables. *Comput Methods Appl Mech Eng*. 2019;351:643-666.
51. Loukrezis D, De Gersem H. Approximation and uncertainty quantification of systems with arbitrary parameter distributions using weighted Leja interpolation. *Algorithms*. 2020;13(3):51.
52. van den Bos L, Sanderse B, Bierbooms W, Bussel G. Bayesian model calibration with interpolating polynomials based on adaptively weighted Leja nodes. *Commun Comput Phys*. 2019;27(1):33-69.
53. Farcaş IG, Görler T, Bungartz HJ, Jenko F, Neckel T. Sensitivity-driven adaptive sparse stochastic approximations in plasma microinstability analysis. *J Comput Phys*. 2020;410:109394.
54. Georg N, Loukrezis D, Römer U, Schöps S. Enhanced adaptive surrogate models with applications in uncertainty quantification for nanoplasmonics. *Int J Uncertain Quantif*. 2020;10(2):165-193.
55. Nobile F, Tamellini L, Tempone R. Comparison of Clenshaw–Curtis and Leja quasi-optimal sparse grids for the approximation of random PDEs. In: Kirby R, Berzins M, Hesthaven J, eds. *Spectral and High Order Methods for Partial Differential Equations ICOSAHOM 2014*. Springer; 2015:475-482.
56. Clenshaw CW, Curtis AR. A method for numerical integration on an automatic computer. *Numer Math*. 1960;2(1):197-205.
57. Gerstner T, Griebel M. Dimension–adaptive tensor–product quadrature. *Computing*. 2003;71(1):65-87.
58. Gander W. Change of basis in polynomial interpolation. *Numer Linear Algebra Appl*. 2005;12(8):769-778.
59. Ghanem RG, Spanos PD. *Stochastic Finite Elements: A Spectral Approach*. Courier Corporation; 2003.
60. Xiu D, Karniadakis GE. The Wiener–Askey polynomial chaos for stochastic differential equations. *SIAM J Sci Comput*. 2002;24(2):619-644.
61. Buzzard GT. Global sensitivity analysis using sparse grid interpolation and polynomial chaos. *Reliab Eng Syst Saf*. 2012;107:82-89.
62. Buzzard GT. Efficient basis change for sparse-grid interpolating polynomials with application to T-cell sensitivity analysis. *Comput Biol J*. 2013;2013:562767.
63. Porta G, Tamellini L, Lever V, Riva M. Inverse modeling of geochemical and mechanical compaction in sedimentary basins through polynomial chaos expansion. *Water Resour Res*. 2014;50(12):9414-9431.
64. Loukrezis D, De Gersem H. Adaptive sparse polynomial chaos expansions via Leja interpolation. arXiv preprint arXiv:1911.08312, 2019.
65. Crestaux T, Le Maıtre O, Martinez JM. Polynomial chaos expansion for sensitivity analysis. *Reliab Eng Syst Saf*. 2009;94(7):1161-1172.
66. Loukrezis D, Galetzka A, De Gersem H. Robust adaptive least squares polynomial chaos expansions in high-frequency applications. *Int J Numer Model Electron Netw Devices Fields*. 2020;33(6):e2725.
67. Wan X, Karniadakis GE. Multi-element generalized polynomial chaos for arbitrary probability measures. *SIAM J Sci Comput*. 2006;28(3):901-928.
68. Bürg M, Dörfler W. Convergence of an adaptive *hp* finite element strategy in higher space-dimensions. *Appl Numer Math*. 2011;61(11):1132-1146.
69. Demkowicz L. *Computing with hp-Adaptive Finite Elements: Volume 1 One and Two Dimensional Elliptic and Maxwell Problems*. Chapman and Hall/CRC; 2006.
70. Dörfler W, Heuveline V. Convergence of an adaptive *hp* finite element strategy in one space dimension. *Appl Numer Math*. 2007;57(10):1108-1124.

71. Houston P, Schwab C, Süli E. Discontinuous *hp*-finite element methods for advection-diffusion-reaction problems. *SIAM J Numer Anal*. 2002;39(6):2133-2163.
72. Mitchell WF, McClain MA. A comparison of *hp*-adaptive strategies for elliptic partial differential equations. *ACM Trans Math Softw*. 2014;41(1):1-39.
73. Jakeman JD, Eldred MS, Sargsyan K. Enhancing l_1-minimization estimates of polynomial chaos expansions using basis selection. *J Comput Phys*. 2015;289:18-34.
74. Houston P, Süli E. A note on the design of hp-adaptive finite element methods for elliptic partial differential equations. *Comput Methods Appl Mech Eng*. 2005;194(2-5):229-243.
75. Nobile F, Tempone R. Analysis and implementation issues for the numerical approximation of parabolic equations with random coefficients. *Int J Numer Methods Eng*. 2009;80(6-7):979-1006.
76. Babuska I, Tempone R, Zouraris GE. Galerkin finite element approximations of stochastic elliptic partial differential equations. *SIAM J Numer Anal*. 2004;42(2):800-825.
77. Saltelli A, Ratto M, Andres T, et al. *Global Sensitivity Analysis: The Primer*. John Wiley & Sons; 2008.
78. Saltelli A, Annoni P, Azzini I, Campolongo F, Ratto M, Tarantola S. Variance based sensitivity analysis of model output. Design and estimator for the total sensitivity index. *Comput Phys Commun*. 2010;181(2):259-270.
79. Sobol IM. Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. *Math Comput Simul*. 2001;55(1-3):271-280.
80. Genz A. *A Package for Testing Multiple Integration Subroutines*. Springer; 1987:337-340.
81. Marelli S, Wagner PR, Lataniotis C, Sudret B. Stochastic spectral embedding. *Int J Uncertain Quantif*. 2020;11:25-47. doi:10.1615/Int.J.UncertaintyQuantification.2020034395
82. Boyd JP. *Chebyshev and Fourier Spectral Methods*. Courier Corporation; 2001.
83. Calvetti D, Somersalo E. Inverse problems: from regularization to Bayesian inference. *WIREs Comput Stat*. 2018;10(3):e1427. doi:10.1002/wics.1427
84. Bardsley JM. Computational uncertainty quantification for inverse problems; 2018. https://github.com/bardsleyj/SIAMBookCodes
85. Smolyak S. Quadrature and interpolation formulas for tensor products of certain classes of functions. *Soviet Math Dokl*. 1963;4:240-243.
86. Kawai S, Yamazaki W, Oyama A. Gegenbauer reconstruction method with edge detection for multi-dimensional uncertainty propagation. *J Comput Phys*. 2022;468:111505. doi:10.1016/j.jcp.2022.111505

How to cite this article: Galetzka A, Loukrezis D, Georg N, De Gersem H, Römer U. An *hp*-adaptive multi-element stochastic collocation method for surrogate modeling with information re-use. *Int J Numer Methods Eng*. 2023;124(12):2902-2930. doi: 10.1002/nme.7234