Sparse polynomial chaos expansions: literature survey and benchmark

Nora Lüthen, Stefano Marelli, Bruno Sudret

▶ **To cite this version:**

Nora Lüthen, Stefano Marelli, Bruno Sudret. Sparse polynomial chaos expansions: literature survey and benchmark. SIAM/ASA Journal on Uncertainty Quantification, 2021. hal-02539750v3

HAL Id: hal-02539750
https://hal.science/hal-02539750v3

Submitted on 20 May 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

ETH zürich

Sparse Polynomial Chaos Expansions: Literature Survey and Benchmark

N. Lüthen, S. Marelli and B. Sudret

Chair of Risk, Safety and Uncertainty Quantification
Stefano-Franscini-Platz 5
CH-8093 Zürich

Risk, Safety & Uncertainty Quantification

Data Sheet

Journal:	SIAM/ASA Journal of Uncertainty Quantification
Report Ref.:	RSUQ-2020-002C
Arxiv Ref.:	https://arxiv.org/abs/2002.01290 [cs.NA, stat.CO, stat.ME]
DOI:	10.1137/20M1315774
Date submitted:	January 29, 2020
Date accepted:	January 24, 2021

Sparse Polynomial Chaos Expansions: Literature Survey and Benchmark

Nora Lüthen[1], Stefano Marelli[1], and Bruno Sudret[1]

[1] *Chair of Risk, Safety and Uncertainty Quantification, ETH Zürich, Stefano-Franscini-Platz 5, 8093 Zürich, Switzerland*

May 19, 2021

Abstract

Sparse polynomial chaos expansions (PCE) are a popular surrogate modelling method that takes advantage of the properties of PCE, the sparsity-of-effects principle, and powerful sparse regression solvers to approximate computer models with many input parameters, relying on only few model evaluations. Within the last decade, a large number of algorithms for the computation of sparse PCE have been published in the applied math and engineering literature. We present an extensive review of the existing methods and develop a framework for classifying the algorithms. Furthermore, we conduct a unique benchmark on a selection of methods to identify which approaches work best in practical applications. Comparing their accuracy on several benchmark models of varying dimensionality and complexity, we find that the choice of sparse regression solver and sampling scheme for the computation of a sparse PCE surrogate can make a significant difference, of up to several orders of magnitude in the resulting mean-squared error. Different methods seem to be superior in different regimes of model dimensionality and experimental design size.

1 Introduction

Computer modelling is used in nearly every field of science and engineering. Often, these computer codes model complex phenomena, have many input parameters, and are expensive to evaluate. In order to explore the behavior of the model under uncertainty (e.g., uncertainty propagation, parameter calibration from data or sensitivity analysis), many model runs are required. However, if the model is costly, only a few model evaluations can be afforded, which often do not suffice for thorough uncertainty quantification. In engineering and applied sciences, a popular work-around in this situation is to construct a surrogate model. A surrogate model is a cheap-to-evaluate proxy to the original model, which typically can be constructed from a relatively small number of model evaluations and approximates the input-output relation of the original model well. Since the surrogate model is cheap to evaluate, uncertainty quantification can be performed at a low cost by using the surrogate model instead of the original model.

Therefore, surrogate modelling aims at constructing a metamodel that provides an accurate approximation to the original model while requiring as few model evaluations as possible for its construction.

In this article, we focus on *nonintrusive regression-based sparse polynomial chaos expansions* (PCE), which is a popular surrogate modelling technique, and within the last decade is has received attention from the communities of applied mathematics and engineering. PCE express the computational model in terms of a basis of polynomials orthonormal with respect to the input random variables (Xiu and Karniadakis, 2002) and work well for globally smooth problems, which are common in many engineering applications. In addition to being a surrogate model, PCE are also often used for uncertainty propagation and sensitivity analysis, since moments and Sobol' sensitivity indices can be computed analytically (Sudret, 2008). *Nonintrusive* PCE treat the model as a black box (unlike intrusive PCE commonly used for solving stochastic PDEs). It is often advantageous to compute a *sparse PCE*, which is an expansion for which most coefficients are zero. This can be justified by the *sparsity-of-effects principle* and by *compressibility*: The sparsity-of-effects principle is a heuristic stating that most models describing physical phenomena are dominated by main effects and interactions of low order (Montgomery, 2004). Furthermore, PCE of real-world models are usually either sparse or at least compressible, meaning that the PCE coefficients, sorted by magnitude, decay quickly. Additional advantages of sparse expansions are given in Section 2.2.

Within the last decade, a large number of articles has been published on the topic of regression-based sparse PCE, each containing promising improvements on how to perform sparse PCE but often lacking a thorough comparison to previously published methods. In this work, we survey the state-of-the-art literature, develop a general framework into which the various approaches can be fit, and carry out a numerical benchmark of a selection of methods to assess which of the many sparse PCE methods perform best on a representative set of realistic benchmark models.

The paper is structured as follows. Section 2 contains the description of our framework for classifying the sparse PCE literature as well as the extensive literature review. Section 3 contains the benchmark description and the numerical results. Finally, conclusions are drawn in section 4. More detailed descriptions of selected sparse solvers and experimental design techniques are given in the Appendices using unified notation.

2 Framework and literature survey for sparse polynomial chaos expansions

2.1 Regression-based polynomial chaos expansions

Let $\boldsymbol{X}$ be a d-dimensional random vector on a domain $\mathcal{D} \subset \mathbb{R}^d$ with independent components and probability density function $f_{\boldsymbol{X}}(\boldsymbol{x}) = \prod_{i=1}^{d} f_{X_i}(x_i)$. Let $L^2_{f_{\boldsymbol{X}}}(\mathcal{D})$ be the space of all scalar-valued models with finite variance under $f_{\boldsymbol{X}}$, i.e., $L^2_{f_{\boldsymbol{X}}}(\mathcal{D}) = \{h : \mathcal{D} \to \mathbb{R} \mid \mathrm{Var}_{\boldsymbol{X}}[h(\boldsymbol{X})] < +\infty\}$. Under certain assumptions on the input distribution $f_{\boldsymbol{X}}$ (Xiu and Karniadakis, 2002; Ernst

et al., 2012), there exists a polynomial orthonormal basis $\{\psi_{\boldsymbol{\alpha}} : \boldsymbol{\alpha} \in \mathbb{N}^d\}$ for $L^2_{f_{\boldsymbol{X}}}(\mathcal{D})$. Since the components of $\boldsymbol{X}$ are assumed to be independent, the basis elements are products of univariate orthonormal polynomials and are characterized by the multi-index $\boldsymbol{\alpha} \in \mathbb{N}^d$ of polynomial degrees in each dimension.

We consider a particular model $\mathcal{M} \in L^2_{f_{\boldsymbol{X}}}(\mathcal{D})$ and denote by $Y = \mathcal{M}(\boldsymbol{X})$ the corresponding output random variable. Y can be represented exactly through an infinite expansion in $\{\psi_{\boldsymbol{\alpha}} : \boldsymbol{\alpha} \in \mathbb{N}^d\}$. In practice, however, not all infinitely many coefficients can be computed, and we are interested in a truncated expansion

$$Y = \mathcal{M}(\boldsymbol{X}) \approx \mathcal{M}^{\mathrm{PCE}}(\boldsymbol{X}) = \sum_{\boldsymbol{\alpha} \in \mathcal{A}} c_{\boldsymbol{\alpha}} \psi_{\boldsymbol{\alpha}}(\boldsymbol{X}) \tag{1}$$

whose accuracy depends on the choice of the finite set $\mathcal{A} \subset \mathbb{N}^d$ (i.e., on the basis elements used for the expansion) as well as on the coefficients $c_{\boldsymbol{\alpha}}$. Several truncation techniques are described in Section 2.4.

To compute the coefficients, one well-known and practical approach is regression (Isukapalli, 1999; Berveiller et al., 2006)[1]. The basic regression approach is *ordinary least squares* (OLS). Let $\{\boldsymbol{x}^{(k)}\}_{k=1}^N \subset \mathcal{D}$ be a sample of the input space called *experimental design* (ED). Let $\boldsymbol{y} = (y^{(1)}, \dots, y^{(N)})^T$ be the vector of model responses with $y^{(k)} = \mathcal{M}(\boldsymbol{x}^{(k)})$. Define the matrix of basis function evaluations $\boldsymbol{\Psi}$ with entries $\Psi_{ij} = \psi_j(\boldsymbol{x}^{(i)})$, where the basis functions are enumerated in an arbitrary way. Denoting the number of basis functions with P, we see that the *regression matrix* $\boldsymbol{\Psi}$ is an $N \times P$-matrix. Then, the OLS regression problem can be written as

$$\hat{\boldsymbol{c}} = \arg\min_{\boldsymbol{c} \in \mathbb{R}^P} \|\boldsymbol{\Psi}\boldsymbol{c} - \boldsymbol{y}\|_2 \,. \tag{2}$$

For a unique and robust solution, a heuristic number of model evaluations is $N \approx 2P, 3P$ (Hosder et al., 2007; Fajraoui et al., 2017), which can be infeasible for high-dimensional or high-degree PCE approximations.

2.2 Sparse PCE

Sparse coefficient vectors are determined through *sparse regression*, which, in addition to a good regression fit, requires that the solution be sparse. This constraint on sparsity is realized e.g. by adding as a regularization term the ℓ^0-"norm" or the ℓ^1-norm of the coefficient vector to the OLS formulation of (2) (see Appendix B for more details). Many sparse regression methods used in PCE were originally developed in the context of compressive sensing (Donoho, 2006; Candès et al., 2006). For an introduction to the concepts and ideas of compressive sensing, see, e.g., Candès and Wakin (2008); Bruckstein et al. (2009); Kougioumtzoglou et al. (2020).

Unlike OLS, compressive sensing methods allow one to use fewer design points N than basis functions P and still recover the true sparse solution, or find a good sparse approximation to it.

[1] Other, earlier approaches for computing the coefficients are stochastic Galerkin and stochastic collocation methods (Ghanem and Spanos, 1991; Xiu and Hesthaven, 2005; Shen et al., 2020). For a comparison of their performance to regression-based PCE, see, e.g., Berveiller (2005); Hosder et al. (2007); Doostan and Owhadi (2011); Mathelin and Gallivan (2012).

This and its robustness to noise, both of which are induced by the sparsity constraint, are the main reasons why sparse PCE are preferred to full PCE in practical settings when the number of model evaluations necessary for OLS-based PCE would be infeasible to compute. Note that while the use of sparse PCE for engineering models can be justified by compressibility and the sparsity-of-effects heuristic (section 1), the main goal in sparse PCE is to compute a good surrogate model from a few model evaluations and not to find the sparsest possible expansion. The assumption of sparsity is used as a tool for finding robust solutions to underdetermined systems of linear equations.

The first publications on sparse regression-based PCE proposed greedy forward-backward selection algorithms (Blatman and Sudret, 2008, 2010a) and introduced the LARS algorithm for sparse PCE (Blatman and Sudret, 2011). On the mathematical side, Doostan and Owhadi (2011) analyzes convergence properties for sparse Legendre PCE when the design points are sampled from the uniform distribution. Another early work is Mathelin and Gallivan (2012) demonstrating that sparse PCE are less costly and more accurate than PCE based on Smolyak sparse grids. Since then, a large number of articles has been published on the topic of sparse PCE suggesting new methods for specific aspects of the sparse regression procedure. In the following, we present a framework into which the existing literature can be fit. The framework provides an overview of the available choices and enables a structured comparison of their impact on the performance of the resulting sparse PCE. Naturally, some new combinations of methods arise that have not yet been considered in the literature.

2.3 Framework: Classifying the literature on sparse PCE

Here, we present the framework we developed in order to gain an overview of the extensive literature proposing new methods for computing sparse PCE. Figure 1 shows a sketch of this framework. To compute a sparse PCE, the first step is to choose a set $\mathcal{A}$ of candidate polynomials for the expansion (Section 2.4) as well as an experimental design (Section 2.5). The experimental design defines the locations of the model evaluations. Once the model evaluations are obtained, the sparse solution can be computed by applying a sparse regression solver (see Section 2.6). This solver often depends on a number of hyperparameters that have to be selected carefully in order to get good results. Then, a suitable model selection criterion is evaluated (Section 2.7). If the obtained solution is satisfactory, the process can be stopped. Otherwise, the basis can be adapted (usually augmented; see Section 2.4), and/or the experimental design can be enriched (see Section 2.5.4). This process is repeated until the value of the model selection criterion either is satisfactory or cannot be reduced.

In addition to the components shown in Figure 1, there are methods (we call them *enhancements*) that aim at generally improving the solution to the sparse regression problem by, e.g., adapting the input space or preconditioning the regression matrix. They are discussed in Section 2.8.

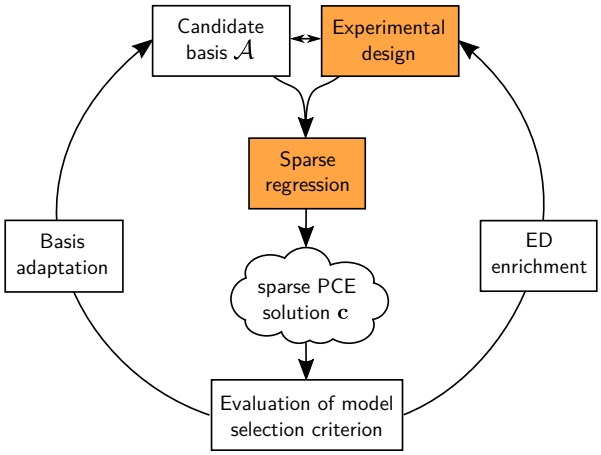


Figure 1: Framework for computing sparse PCE. For each component of the framework, a number of methods has been proposed in the literature. In the first part (section 2) of this paper, we review the literature for each of the components. Details on selected methods are given in Appendices A and B. In the second part (section 3), we conduct a benchmark of selected methods for the components marked in orange, performing a single iteration of the framework. Iterative basis adaptation and experimental design enrichment are not considered in this work and are left for future benchmarks.

2.4 Choice of basis and basis adaptation

The approximation quality of a truncated PCE hinges on the polynomial functions available for building the surrogate model, which are characterized by the associated set of multi-indices $\mathcal{A} \subset \mathbb{N}^d$. We call the finite set of polynomials $\{\psi_{\boldsymbol{\alpha}} : \boldsymbol{\alpha} \in \mathcal{A}\}$ included in the current truncated PCE model *basis* and call its members *candidate polynomials* or *candidate basis functions*. A sparse PCE algorithm will find nonzero coefficients only for a subset $\mathcal{A}^{\text{active}} \subset \mathcal{A}$ of the basis functions, which we call *active basis functions*. On the one hand, $\mathcal{A}$ should include enough candidate polynomials to facilitate a good approximation. On the other hand, unnecessary basis functions decrease the ratio N/P of model evaluations to unknown coefficients and deteriorate the regression matrix. Therefore, it is beneficial to carefully select the polynomials to be included in the expansion.

The choice of basis is often motivated by the *sparsity-of-effects principle*, a heuristic guideline stating that most real-world models are well approximated by terms of low degree and low interaction order. The following are popular ways to construct a basis:

- *Total-degree* A total-degree basis of degree p is defined by $\mathcal{A}^p = \{\boldsymbol{\alpha} : \|\boldsymbol{\alpha}\|_1 \leq p\}$.
- *Hyperbolic truncation* Let p be fixed. Define the q-norm-truncated basis

$$\mathcal{A}^{p,q} = \{\boldsymbol{\alpha} : \|\boldsymbol{\alpha}\|_q \leq p\} \tag{3}$$

  with $q \in (0, 1]$ (Blatman and Sudret, 2011) and the quasi-norm $\|\boldsymbol{x}\|_q = \left(\sum_{i=1}^d |x_i|^q\right)^{\frac{1}{q}}$. For $q = 1$, $\mathcal{A}^{p,1}$ is the total-degree basis of order p. For smaller q, this truncation scheme

excludes terms with high interaction order while keeping univariate polynomials up to degree p.

- *Interaction order* The interaction order of the basis can be restricted by defining

$$\mathcal{A}^{p,r} = \{\boldsymbol{\alpha} \in \mathcal{A}^p : \#\{i : \alpha_i \neq 0, i = 1, \ldots, d\} \leq r\} \tag{4}$$

(Blatman and Sudret, 2008; Marelli and Sudret, 2019). This is useful for reducing the number of basis functions especially in high dimensions and when it is known (e.g., for physical reasons) that only a certain number of variables might interact.

Instead of using a fixed basis $\mathcal{A}$, it can be beneficial to employ an iterative scheme which starts from a small set of basis functions (low-dimensional, low-order) and, after computing a sparse solution, repeatedly adapts the basis by including a set of the most promising candidate polynomials and possibly removing others. This is called *basis adaptivity* (not to be confused with Gaussian adaptation (Tipireddy and Ghanem, 2014); see also Section 2.8).

A simple instance of basis adaptivity is *degree adaptivity* (Blatman and Sudret, 2010a), which is based on total-degree bases. The procedure starts with a basis of low total degree and iteratively increases the total degree of the basis. Finally, a model selection criterion is used to select the best basis and associated sparse solution. Similarly, q-norm and interaction order, or a combination of all three, can also be used to design a basis adaptation scheme (Blatman and Sudret, 2010a, 2011). This basis adaptivity is solution-agnostic in the sense that it does not use any information from the solutions computed in previous runs for the augmentation of the basis. Another solution-agnostic method is the dimension- and order-incrementing algorithm of Alemazkoor and Meidani (2017). Two methods that adapt the basis based on the active terms of the previous sparse solution are forward neighbor basis adaptivity (Sargsyan et al., 2014; Jakeman et al., 2015) and anisotropic degree basis adaptivity (Hampton and Doostan, 2018). These approaches keep the size of the basis small by strictly controlling which functions are added to the basis, often starting with a constant surrogate model and adding dimensions only when necessary. A discussion and benchmark of basis-adaptive methods for sparse PCE is available in Lüthen et al. (2021).

2.5 Experimental design

Generally, experimental design techniques aim to select points in order to achieve certain goals related to exploration of the space (space-filling design), or to achieve certain properties of the regression matrix such as (in expectation) orthonormal columns, small determinant, small condition number, etc. For regression-based PCE, there are several main classes of experimental design techniques:

- *Sampling based on the input distribution.* The samples are drawn from the input distribution. Techniques like Latin hypercube sampling (LHS) can be used to improve the space-filling properties.

- *Sampling from a different distribution* (also called *induced sampling* (Guo et al., 2020)). A different distribution and associated basis are constructed that have better properties than the input distribution and its basis.

- *Choosing points according to an optimality criterion from a candidate set.* Certain properties of the regression matrix are optimized by choosing the design points from a suitable candidate set.

Note that none of these sampling methods considers the evaluations $\boldsymbol{y}$ of the computational model. For model-aware sampling techniques (active and supervised learning), see Section 2.5.4.

The following sections contain an overview of available sampling methods for sparse PCE. Selected experimental design techniques are described in more detail in Appendix A. Note that due to the large amount of literature on experimental design for sparse PCE, our review cannot be exhaustive. There are many more approaches available, including the deterministic Weil points (Zhou et al., 2014), sparse grids (Perkó et al., 2014), randomized or subsampled quadrature points (Berveiller et al., 2006; Tang and Iaccarino, 2014; Guo et al., 2017), etc.

2.5.1 Sampling based on the input distribution

The most basic sampling method is Monte Carlo (MC) sampling, where the points are sampled independently from the input distribution (Doostan and Owhadi, 2011; Hampton and Doostan, 2015b). LHS (McKay et al., 1979) aims at distributing the design points in a more space-filling way than MC sampling, using a stratification of the input quantile space in each dimension. LHS is known to filter main effects; i.e., it reduces the variance of linear regression estimators when the quantity of interest is dominated by terms of interaction order one (Shields and Zhang, 2016). LHS with sample decorrelation can further reduce the variance (Owen, 1994). LHS can also be used together with a criterion such as maximin distance (maximize the minimal distance between the design points in quantile space) (Pronzato and Müller, 2012), where several LHS designs are generated and the one that optimizes the criterion is returned.

A generalization of LHS that combines it with stratified sampling is *Latinized partially stratified sampling* (Shields and Zhang, 2016), which filters both main effects and low-order interaction terms and has been shown to consistently outperform LHS in high-dimensional cases.

Other space-filling/low-discrepancy methods are Sobol' sequences (Sobol', 1967) and Halton sequences (Halton, 1960), which are deterministic but appear to be quasi-random and space-filling in low dimensions.

2.5.2 Sampling from a different distribution

Several methods consider the *coherence* parameter of a basis, defined by

$$\mu(\mathcal{A}, \{\psi_{\boldsymbol{\alpha}}\}) = \sup_{\boldsymbol{x}\in\mathcal{D}} \max_{\boldsymbol{\alpha}\in\mathcal{A}} |\psi_{\boldsymbol{\alpha}}(\boldsymbol{x})|^2, \tag{5}$$

which can be used to bound the number of samples needed for accurate recovery by ℓ^1-minimization (Candès and Plan, 2011; Hampton and Doostan, 2015b). For Hermite and Legendre polynomial bases, the coherence parameter of a total-degree basis grows exponentially with the total degree p (Rauhut and Ward, 2012; Yan et al., 2012; Hampton and Doostan, 2015b).

To construct a *coherence-optimal design*, a new probability distribution and its associated orthonormal basis are constructed that achieve minimal coherence (Hampton and Doostan, 2015b,a). The new basis can be derived from the original PCE basis by multiplying each member by a weight function. Coherence-optimal samples can be drawn by Markov Chain Monte Carlo (MCMC) (Hampton and Doostan, 2015b) or by rejection sampling (see section A.2.3). A related sampling scheme is obtained by constructing a new probability distribution and associated orthonormal basis which have improved but not optimal coherence; however, the distribution is constructed to belong to some classical family and is therefore straightforward to sample. This is called *asymptotic sampling* (Hampton and Doostan, 2015b,a) and results in a Chebyshev distribution for uniform input, and in a uniform distribution (within a ball of degree-dependent radius) for Gaussian input. Numerical experiments confirm the expected performance gain of coherence-optimal sampling over both MC and asymptotic sampling, and of asymptotic sampling over MC in the case of low dimension d and high total degree p. For high-dimensional problems with low degree, MC often performs better than asymptotic sampling (Hampton and Doostan, 2015b).

The so-called *Christoffel sparse approximation (CSA)* (Jakeman et al., 2017; Narayan et al., 2017; Cohen and Migliorati, 2017) is a related approach which constructs a new orthonormal basis that minimizes the quantity

$$\tilde{\mu}(\mathcal{A}, \{\psi_{\boldsymbol{\alpha}}\}) = \sup_{\boldsymbol{x}\in\mathcal{D}} \left(\frac{1}{|\mathcal{A}|} \sum_{\boldsymbol{\alpha}\in\mathcal{A}} |\psi_{\boldsymbol{\alpha}}(\boldsymbol{x})|^2 \right)^{\frac{1}{2}}. \tag{6}$$

As for coherence-optimal sampling, the new basis can be derived from the original basis by multiplying each member by a weight function, which results in a weighted regression problem. The corresponding probability distribution is chosen to be the so-called *weighted pluripotential equilibrium measure*, which for bounded distributions is the Chebyshev distribution. For one-dimensional Gaussian input, this measure is a symmetric Beta distribution with degree-dependent bounds.

Note that all three sampling methods described in this section introduce weights and therefore modify the objective function into a weighted regression problem $\boldsymbol{W}\boldsymbol{\Psi}\boldsymbol{c} \approx \boldsymbol{W}\boldsymbol{y}$. Since the objective function belongs to the scope of the solver, these methods cannot be considered as pure sampling methods in the sense of being completely independent of the solver.

2.5.3 Choosing points according to an optimality criterion from a candidate set

The following methods choose points from a candidate set in order to optimize properties of the regression matrix. Candidate points can be sampled, e.g., using MC, LHS (Fajraoui et al., 2017), coherence-optimal sampling (Diaz et al., 2018; Alemazkoor and Meidani, 2018a), or Christoffel

sampling (Shin and Xiu, 2016b). Note that some of these methods introduce weights, resulting in a weighted regression problem. The candidate set can have a large influence on the resulting design.

- *D-optimal sampling* aims at maximizing the determinant $D(\mathbf{\Psi}) = \det(\frac{1}{N}\mathbf{\Psi}^T\mathbf{\Psi})^{\frac{1}{P}}$ of the information matrix (Kiefer and Wolfowitz, 1959). Note that $D(\mathbf{\Psi}) = 0$ if $N < P$. Maximizing the D-value is connected to minimizing the variance of the coefficients of the PCE estimate (Zein et al., 2013). Algorithms for D-optimal designs include greedy augmentation (Dykstra, 1971), exchange techniques (Fedorov, 2013; Cook and Nachtsheim, 1980; Nguyen and Miller, 1992; Zein et al., 2013), maxvol (Mikhalev and Oseledets, 2018), gradient descent (Zankin et al., 2018), and rank-revealing QR decomposition (RRQR)/subset selection (Diaz et al., 2018; Gu and Eisenstat, 1996). The advantage of the last method is that it can also be applied for wide matrices $\mathbf{\Psi} \in \mathbb{R}^{N\times P}$ where $N < P$.

- *S-optimal sampling* (also called "quasi-optimal" in Shin and Xiu (2016a)) selects samples from a large pool of candidate points so that the PCE coefficients computed using the selected set are as close as possible to the coefficients computed from the whole set of candidate points (Shin and Xiu, 2016a). The S-value is defined by[2]

$$S(\mathbf{\Psi}) = \left(\frac{\sqrt{\det \mathbf{\Psi}^T\mathbf{\Psi}}}{\prod_{i=1}^{P} \|\Psi_i\|_2}\right)^{\frac{1}{P}} \tag{7}$$

  where Ψ_i denotes the ith column of the regression matrix. Its maximization has the effect of maximizing the column orthogonality of the regression matrix while at the same time maximizing the determinant of the information matrix (Shin and Xiu, 2016a). Note that $S(\mathbf{\Psi}) = 0$ if $N < P$. An S-optimal experimental design can be computed using a greedy exchange algorithm (Shin and Xiu, 2016a,b; Fajraoui et al., 2017).

- *Near-optimal sampling* simultaneously minimizes the two matrix properties *mutual coherence* and *average cross-correlation* (Alemazkoor and Meidani, 2018a), both of which quantify the correlation between normalized columns of the regression matrix (see section A.3.3 for the definitions of these properties). A near-optimal design can be built by a greedy algorithm (Alemazkoor and Meidani, 2018a). Note that for near-optimal sampling, it is not necessary that $N \geq P$, since this method does not rely on the determinant of the information matrix.

2.5.4 Sequential enrichment of the experimental design

Instead of sampling the whole experimental design at once, it has been proposed to use *sequential enrichment*. Starting with a small experimental design, additional points are chosen based on the last computed sparse PCE solution or on an augmented basis. In the context of machine learning, sequential sampling is also known as *active learning* (Settles, 2012). Sequential enrichment

[2]This definition assumes that the columns of the matrix $\mathbf{\Psi}_{\text{cand}}$, containing the evaluations of all candidate points, are mutually orthogonal.

has been proposed in the context of S-optimal sampling (Fajraoui et al., 2017), D-optimal sampling (Diaz et al., 2018), and coherence-optimal sampling (Hampton and Doostan, 2018). Zhou et al. (2019) suggest an enrichment strategy based on approximations to the expected quadratic loss function, i.e., the mean-squared error. Ji et al. (2008) and Seeger and Nickisch (2008) propose choosing points that minimize the differential entropy of the posterior distribution of the coefficients (using a Bayesian regression setting). In all cases, numerical examples show that the sequential strategy generally leads to solutions with a smaller validation error compared to nonsequential strategies. Due to the complexity of the topic and the already large extent of our benchmark, this strategy, albeit promising, is not explored further in this paper.

2.6 Solution of the minimization problem

There are many formulations of the regression problem that lead to a sparse solution, such as ℓ^0-minimization, ℓ^1-minimization (basis pursuit denoising (BPDN), LASSO), $\ell^1 - \ell^2$ minimization, Bayesian methods, etc. (see also Appendix B). Based on these formulations, a vast number of sparse solvers has been proposed in the compressed sensing literature; see, e.g., Carron (2013) and the surveys of Qaisar et al. (2013); Zhang et al. (2015); Arjoune et al. (2017). We focus here on solvers that have been proposed in the context of sparse PCE. Of course, it is straightforward to use any other sparse solver to compute a sparse PCE.

The following solvers have been proposed in the sparse PCE literature:

- *Convex optimization solvers.* ℓ^1-minimization in its various formulations is a (constrained) convex optimization problem. Least angle regression (LARS) (Efron et al., 2004; Blatman and Sudret, 2011; Marelli and Sudret, 2019) is an iterative method that adds regressors one by one according to their correlation with the current residual, and updates the coefficients following a least angle strategy. With the LARS-LASSO modification, which allows for backwards elimination of regressors, LARS is able to generate the whole LASSO path (Efron et al., 2004). Unmodified LARS can also be classified as a greedy method. SPGL1 (van den Berg and Friedlander, 2008; Van den Berg and Friedlander, 2015) solves the BPDN formulation by solving a succession of LASSO instances using the spectral projected gradient (SPG) method. Other solvers belonging to this class are e.g. the solvers implemented in $\ell_1$magic (Candès and Romberg, 2005) and SparseLab (Donoho et al., 2007).

- *Greedy methods* are variants of stepwise regression where the regressors are added to the model one by one according to some selection criterion, aiming at finding a heuristic solution to the intractable ℓ^0-minimization formulation. Orthogonal matching pursuit (OMP) (Tropp and Gilbert, 2007; Doostan and Owhadi, 2011; Marelli and Sudret, 2019) is a classical forward selection algorithm in which orthonormalized regressors are added to the model one by one according to their correlation with the residual, and the coefficients are computed by least-squares. Baptista et al. (2019) suggests extensions to OMP such as parallelization, randomization and a modified regressor selection procedure. Subspace

pursuit (SP) (Dai and Milenkovic, 2009; Diaz et al., 2018; Diaz, 2018) is an iterative algorithm that repeatedly uses least squares on a subset of regressors. LARS (Efron et al., 2004; Blatman and Sudret, 2011) without the LASSO modification (allowing for removal of regressors) can also be classified as a greedy method. Another greedy method is ranking-based sparse PCE (Tarakanov and Elsheikh, 2019) which employs batch updating, coordinatewise gradient descent of the elastic net formulation, and a correlation- and stability-based ranking procedure for the regressors. Many more greedy stepwise regression techniques have been proposed, utilizing various selection criteria, solvers, and stopping criteria. An overview of methods following this scheme is given in section B.6.

- *Bayesian compressive sensing (BCS)* (a.k.a. sparse Bayesian learning) is a class of methods that use a Bayesian setting to find a sparse solution. They impose a sparsity-inducing prior on the coefficients, whose parameters are again considered to be random variables with a hyperprior (Tipping, 2001; Ji et al., 2008; Sargsyan et al., 2014; Tsilifis et al., 2020). The solution is typically the maximum a posteriori estimate of the coefficients and can be computed e.g. by differentiation (Tipping, 2001), expectation-maximization (Figueiredo, 2003; Wipf and Rao, 2004), expectation-propagation (Seeger and Nickisch, 2008), variational inference (Tsilifis et al., 2020; Bhattacharyya, 2020), or a fast approximate algorithm (Faul and Tipping, 2002; Tipping and Faul, 2003). An extension called FastLaplace with an additional layer of hyperparameters has been proven to attain even sparser solutions (Babacan et al., 2010; Babacan, 2011). A greedy algorithm using the Bayesian setting to select the regressors is the greedy Bayesian Kashyap information criterion (KIC)-based algorithm (Shao et al., 2017).

- *Iteratively reweighted methods.* Iteratively reweighted ℓ^1-minimization uses the coefficients computed in a previous iteration to construct a weighted ℓ^1-minimization problem (Candès et al., 2008; Yang and Karniadakis, 2013). Cheng and Lu (2018b) suggest an iterative reweighted method with D-MORPH regression (Li and Rabitz, 2010) as its computational core, which is a technique that follows a certain path, defined by a quadratic objective function, on the manifold of solutions to the underdetermined system.

Each of the solvers mentioned above features one or more hyperparameters whose values must be calibrated. This is usually done by cross-validation. Popular choices are leave-one-out (LOO) cross-validation (accelerated for least-squares solutions) (Blatman and Sudret, 2010a, 2011), LOO cross-validation with a modification factor for small sample sizes (Chapelle et al., 2002; Blatman and Sudret, 2011), and k-fold cross-validation (Doostan and Owhadi, 2011; Jakeman et al., 2015; Huan et al., 2018).

Selected sparse regression solvers are described in more detail in Appendix B.

2.7 Model selection criterion

To decide whether to continue iterating in the framework or stop the process, we need to assess how well the current sparse solution performs. Our main quantity of interest is the *generalization*

error, which quantifies the mean-square accuracy of the surrogate. It is given by

$$E_{\text{gen}} = \mathbb{E}_{\boldsymbol{X}}\left[\left(\mathcal{M}(\boldsymbol{X}) - \mathcal{M}^{\text{PCE}}(\boldsymbol{X})\right)^2\right] \tag{8}$$

where $\mathcal{M}$ is the computational model, $\boldsymbol{X}$ is the random input vector, and $\mathcal{M}^{\text{PCE}}$ is the sparse PC surrogate.

The generalization error can be approximated by the *validation error*, which is the MC estimate of (8) on a validation set $\{(\boldsymbol{x}_{\text{val}}^{(i)}, y_{\text{val}}^{(i)}) : \boldsymbol{x}_{\text{val}}^{(i)} \sim_{\text{i.i.d.}} f_{\boldsymbol{X}}, y_{\text{val}}^{(i)} = \mathcal{M}(\boldsymbol{x}_{\text{val}}^{(i)}), i = 1, \ldots, N_{\text{val}}\}$. To make the validation error independent of the scaling of the model, it is convenient to use the *relative mean-squared error* defined by

$$\text{RelMSE} = \frac{\sum_{i=1}^{N_{\text{val}}}(y_{\text{val}}^{(i)} - \mathcal{M}^{\text{PCE}}(\boldsymbol{x}_{\text{val}}^{(i)}))^2}{\sum_{i=1}^{N_{\text{val}}}(y_{\text{val}}^{(i)} - \bar{y})^2} \tag{9}$$

where $\bar{y} = \frac{1}{N_{\text{val}}}\sum_{i=1}^{N_{\text{val}}} y_{\text{val}}^{(i)}$.

The best surrogate model $\mathcal{M}^{\text{PCE}}$ (defined by $\mathcal{A}$ and $\boldsymbol{c}$) is the one that has the smallest generalization error. In practical applications, the generalization error typically cannot be computed, and a large validation set is not available due to computational constraints. Instead, we define a *model selection criterion* that acts as a proxy for the generalization error. A typical stopping criterion in the PCE framework of Figure 1 is the observation that the model selection criterion no longer improves. The following model selection criteria have been proposed in the sparse PCE literature:

- *k-fold cross-validation (CV)* (Hastie et al., 2001; Jakeman et al., 2015; Hampton and Doostan, 2018), which approximates the validation error by building a surrogate several times on different subsets of the data, and evaluating the error on the remaining data points.

- *Leave-one-out (LOO) cross-validation* (Hastie et al., 2001; Blatman and Sudret, 2010a, 2011), which is N-fold cross-validation (where N is the size of the experimental design). For PCE approximations computed by OLS, there exists an efficient formula to evaluate the LOO error (Blatman and Sudret, 2011, Appendix D).

- *Modified LOO* (Blatman and Sudret, 2011), which uses a correction factor for the LOO which was derived for the empirical error for OLS with small sample size (Chapelle et al., 2002). The correction factor depends on the experimental design and the active basis functions.

- *Kashyap information criterion (KIC)* (Shao et al., 2017; Zhou et al., 2019), an approximation to the Bayesian model evidence, which is the likelihood of observations given the model.

- *Sparsity* (Alemazkoor and Meidani, 2017), which uses the idea that a larger basis should lead to a sparser solution when the necessary basis functions enter the model, unless the ratio of basis functions to model evaluations becomes too large.

A model selection criterion is also often used to determine the hyperparameter of the sparse solver (see Section 2.6).

If cross-validation is used to select the solver hyperparameter, this estimate of the validation error is often too optimistic due to model selection bias. Instead of reusing this estimate for model selection, it is better to perform an outer loop of k-fold or LOO cross-validation, a procedure called *double cross-validation* or cross-model validation (Baumann and Baumann, 2014; Liu et al., 2020b).

2.8 Further enhancements of sparse PCE

There are many enhancements to the simple scheme for sparse PCE presented in Figure 1. The following methods have been suggested to improve the accuracy of the solution and reduce the number of model evaluations needed:

- Alemazkoor and Meidani (2018b) construct a preconditioning matrix for a given regression matrix which reduces the mutual coherence while avoiding deterioration of the signal-to-noise ratio.
- Huan et al. (2018) suggest a technique called stop-sampling, which guides the decision of whether to obtain more samples (sequential ED enrichment) by observing the decrease of the CV error.
- In the case when prior information about the magnitude of the coefficients is available, Peng et al. (2014) use this information to construct a weighted regression problem which allows a more accurate solution with fewer points (similar to iteratively reweighted ℓ^1-minimization).
- Liu et al. (2020a) use resampled PCE, which is a technique for improving the PCE solution by aggregating the results of several solver runs on different subsets of the data. Only the terms that are chosen most often by the solvers are retained in the final solution.
- Several methods exist to reduce the dimension of the input space before computing the sparse PCE. Unsupervised methods are principal component analysis (PCA) and kernel PCA (Lataniotis et al., 2019). "Basis adaptation" methods (referring to a basis of the input random space) determine a suitable rotation of the input space, often assumed to be independent standard Gaussian, into new coordinates which permit a sparser representation in fewer coordinates (see Tipireddy and Ghanem (2014); Yang et al. (2018); Tsilifis et al. (2019) and others). A related technique is nonlinear PCE-driven partial least squares (PLS) (Papaioannou et al., 2019; Zhou et al., 2020), which reduces the input dimension by identifying directions in the input space that are able to explain the output well in terms of a sum of one-dimensional PCEs.

3 Numerical results

3.1 Benchmark design

While the number of methods for computing sparse PCE is large, to the best of the authors' knowledge there is no comprehensive benchmark study on this topic. Most publications only compare the newly developed method to one or two baseline methods. An overview of publications containing comparisons of sparse PCE methods is presented in Appendix D.

Since the number of possible combinations of sampling schemes, sparse regression solvers, basis adaptation schemes, model evaluation criteria etc. is huge (see Section 2.3 and thereafter), we restrict our benchmark as follows to some of the most promising and best-known methods:

- We consider the sampling schemes MC, LHS, coherence-optimal, and D-optimal. LHS is used together with a maximin criterion to improve the space-filling property (using the MATLAB function `lhsdesign`). D-optimal designs are constructed from a coherence-optimal candidate set[3] using the subset selection/RRQR algorithm, which allows for the construction of D-optimal experimental designs with size N smaller than the number of regressors P (Diaz et al., 2018). There is no such algorithm for S-optimal sampling, which is why we do not consider the latter in this benchmark. We do not consider Sobol' sequences, since they have been shown to be outperformed by LHS in sparse PCE applications (Fajraoui et al., 2017). Near-optimal sampling can realistically be used only for rather small bases ($P \in \mathcal{O}(100)$), since its algorithm scales as $\mathcal{O}(MP^2)$, with $M = 10P$ as suggested by (Diaz et al., 2018) (see below). We use it with a coherence-optimal candidate set for two models with small basis.

- We consider the sparse regression solvers LARS, OMP, subspace pursuit (SP), FastLaplace (which we call here BCS), and SPGL1. Each of these solvers involves at least one hyperparameter, whose range is chosen according to reasonable guesses. For LARS and OMP, the hyperparameter is the number K of selected regressors and its range is $[1, \min\{P, N-1\}]$. For SP, K must fulfill $2K \leq \min\{P, N\}$. For BCS and SPGL1, the hyperparameter σ is chosen from the range $\sigma^2 \in N \cdot \hat{\mathrm{Var}}[\boldsymbol{y}] \cdot [10^{-16}, 10^{-1}]$ which resembles a suitable range of possible relative MSE values. The hyperparameter values of LARS and OMP are determined by modified LOO cross-validation, while the hyperparameters of SP, BCS, and SPGL1 are determined by k-fold cross-validation (Section 2.6). In addition, we consider a variant of SP which uses LOO cross-validation instead of k-fold cross-validation, which we name SP_{LOO}.

- We only consider the nonadaptive setting, in which both the basis and the size of the experimental design are fixed before the sparse PCE is computed.

[3] We have also conducted all benchmark experiments with D-optimal designs constructed from LHS candidate sets, but we do not display these results, because we found that in most cases, D-opt(LHS) sampling performs (significantly) worse than most other sampling schemes, and often worse than its candidate set LHS. This matches with the results of Fajraoui et al. (2017) who observed this in a sequential enrichment setting and with the LARS solver.

- For each model, we define a reasonable range of 5–7 experimental design sizes. Each experiment is repeated 30–50 times to account for statistical uncertainty. The experimental designs are generated anew for each repetition and each ED size. All solvers are tested on the same ED realizations.

- The coherence-optimal candidate sets from which the D-optimal designs are selected have size $M = 10P$ as in (Diaz et al., 2018). For computational reasons, they are not sampled completely anew for each replication, but are rather drawn uniformly at random without replacement from a larger set of size $2M = 20P$ as in (Diaz et al., 2018).

- Since we are interested in sparse PCE for the purpose of surrogate modelling, our main quantity of interest is the relative mean-squared error[4][5] (RelMSE) as defined in (9). We investigate the RelMSE for several models, sparse solvers, and experimental design techniques. Typically, the practical interest lies in small experimental designs.

- Since the experimental design is random, the resulting validation error is a random variable. We visualize the data with boxplots. When comparing the performance of different methods, we consider the median performance and the spread of the resulting validation error. However, often there can be considerable overlap of validation errors between methods.

3.2 Software

For the implementation of the benchmark, we use the general-purpose uncertainty quantification software UQLab (Marelli and Sudret, 2014). UQLab supports the integration of other software packages.[6] We utilize the following code:

- UQLab for MC sampling and LHS (Marelli and Sudret, 2014).

- `DOPT_PCE` for D-optimal sampling (subset selection/RRQR) and subspace pursuit (Diaz et al., 2018; Diaz, 2018).

- An in-house developed rejection-based implementation of coherence-optimal sampling.

[4]Note that some authors such as Doostan and Owhadi (2011); Hampton and Doostan (2015b); Diaz et al. (2018); Alemazkoor and Meidani (2018a) choose to normalize instead by $\sum_{\boldsymbol{x}\in\mathcal{X}_{\text{val}}} \mathcal{M}(\boldsymbol{x})^2$ or use the unnormalized mean-squared error (Shin and Xiu, 2016b). To assess the recovery of sparse vectors just as in compressed sensing, some consider the error in the coefficient vector instead of the error in the model approximation (Alemazkoor and Meidani, 2018a).

[5]Since a typical application of PCE is the computation of moments and Sobol' indices, the error in these quantities is another possible performance measure. However, globally accurate prediction as considered in this paper is more challenging that the prediction of moments and Sobol' indices, which are accurate if the largest-in-magnitude coefficients are estimated accurately. If a globally accurate surrogate model can be constructed, typically also the moments and Sobol' indices are accurate.

[6]A description of how to use custom sparse solvers and sampling schemes in the UQLab framework can be found in the supplementary material.

- An in-house implementation of near-optimal sampling based on the description by Alemazkoor and Meidani (2018a).
- UQLab for the solvers LARS and OMP (Marelli and Sudret, 2014).
- `spgl1-1.9` for SPGL1 (van den Berg and Friedlander, 2008; Van den Berg and Friedlander, 2015).
- `FastLaplace` for the hierarchical implementation "FastLaplace" of BCS (Babacan et al., 2010; Babacan, 2011).

3.3 Benchmark: Considered models

Our benchmark is performed on a selection of 11 computational models of varying complexity and input dimensionality, which are typical benchmark models in the context of sensitivity and reliability analysis. An overview of these models is given in Table 1. For details on the models, we refer the reader to the respective publications supplied in the last column of the table. While of course not representative of all possible classes of engineering models, we believe that this sample provides a good testing ground for the comparative performance among different approaches for computing sparse PCE.

In addition to analyzing aggregated performance on all 11 models, we investigate the behavior of the methods in detail on a subset of four *spotlight models*, each of which possesses characteristic properties that might influence the approximation quality of sparse PCE methods: the *Ishigami function* is low-dimensional and highly compressible in the PCE basis but requires a high-degree basis to be approximated accurately. The *borehole function* is smooth and nonlinear and therefore is an example for a well-behaved engineering model. A *two-dimensional diffusion model*, a stochastic heat diffusion PDE in two physical dimensions, is high-dimensional, not analytical, and the magnitude of its expansion coefficients decays only slowly. Finally, the $100D$ *function* is high-dimensional, analytical, and compressible.

The Ishigami model is the well-known three-dimensional, highly nonlinear, smooth analytical function

$$f(X_1, X_2, X_3) = \sin(X_1) + a \sin^2(X_2) + b X_3^4 \sin(X_1) \tag{10}$$

taking uniform input $\boldsymbol{X} \sim \mathcal{U}([-\pi, \pi]^3)$. A typical choice is $a = 7, b = 0.1$. For this function, any sparse solver should be able to find a sparse solution.

The borehole function simulates the water flow through a borehole between two aquifers (Harper and Gupta, 1983). It is an eight-dimensional nonlinear function which, despite having an analytical form, is not trivial to approximate. It is defined by

$$B(r_w, L, K_w, T_u, T_l, H_u, H_l, r) = \frac{2\pi T_u (H_u - H_l)}{\ln(r/r_w)\left(1 + \frac{2LT_u}{\ln(r/r_w) r_w^2 K_w} + \frac{T_u}{T_l}\right)}. \tag{11}$$

Its input random variables and their distributions are provided in Table 2.

Table 1: Overview of the 11 computational models used in our benchmark. Italic font denotes finite element (FE) models , all other models are analytical. For each model, a static total-degree basis with hyperbolic truncation defined by p and q is used. The values are chosen to fulfill $P \approx \frac{10}{3} N_{\max}$, where $N_{\max}$ is the largest tested experimental design size. The values for p in parentheses for the Ishigami and borehole models refer to the smaller basis used in Section 3.6. The column "Reference" provides the relevant literature in which the models and their probabilistic inputs are described in detail.

Model	Dimension	Input distributions	Basis	$N_{\max}$	Reference
Ishigami function	3	uniform	$p = 14$ (12), $q = 1$	200	(Blatman and Sudret, 2011)
Undamped oscillator	6	Gaussian	$p = 5$, $q = 1$	150	(Echard et al., 2013)
Borehole function	8	Gaussian, lognormal, uniform	$p = 5$ (4), $q = 1$	300	(Harper and Gupta, 1983)
Damped oscillator	8	lognormal	$p = 5$, $q = 1$	400	(Dubourg, 2011)
Wingweight function	10	uniform	$p = 4$, $q = 1$	300	(Forrester et al., 2008)
Truss model	10	lognormal, Gumbel	$p = 4$, $q = 1$	300	(Blatman and Sudret, 2011)
Morris function	20	uniform	$p = 8$, $q = 0.5$	400	(Blatman and Sudret, 2010b)
Structural frame model	21	lognormal, Gaussian; dependent input variables	$p = 8$, $q = 0.5$	400	(Blatman and Sudret, 2010a)
2-*dim diffusion model*	53	Gaussian	$p = 4$, $q = 0.5$	500	(Konakli and Sudret, 2016)
1-*dim diffusion model*	62	Gaussian	$p = 4$, $q = 0.5$	500	(Fajraoui et al., 2017)
100D function	100	uniform	$p = 4$, $q = 0.5$	1400	UQLab example[7]

Table 2: Borehole function: Input random variables and their distributions

Variable	Distribution	Description
r_w	$\mathcal{N}(0.10, 0.0161812)$	borehole radius
L	$\mathcal{U}([1120, 1680])$	borehole length
K_w	$\mathcal{U}([9855, 12045])$	borehole hydraulic conductivity
T_u	$\mathcal{U}([63070, 115600])$	transmissivity of upper aquifer
T_l	$\mathcal{U}([63.1, 116])$	transmissivity of lower aquifer
H_u	$\mathcal{U}([990, 1110])$	potentiometric head of upper aquifer
H_l	$\mathcal{U}([700, 820])$	potentiometric head of lower aquifer
r	Lognormal$([7.71, 1.0056])$	radius of influence

The two-dimensional heat diffusion model (Konakli and Sudret, 2016) is defined by the partial differential equation (PDE)

$$-\nabla \cdot (\kappa(\boldsymbol{x})\nabla T(\boldsymbol{x})) = Q\mathbb{1}_A(\boldsymbol{x}) \qquad \text{in } \Omega = [-0.5, 0.5]^2 \tag{12}$$

with boundary conditions $T = 0$ on the top boundary and $\nabla T \cdot \boldsymbol{n} = 0$ on the left, lower, and right boundaries of the square domain Ω, where $\boldsymbol{n}$ denotes the outer unit normal (see (Konakli and Sudret, 2016) for an illustration of the setup). Here, the source is in $A = [0.2, 0.3]^2$ with strength $Q = 500$. The output quantity of interest is the average temperature in $B = [-0.3, -0.2]^2$. The diffusion coefficient $\kappa(\boldsymbol{x})$ is modelled by a lognormal random field with mean $\mu_\kappa = 1$ and standard deviation $\sigma_\kappa = 0.3$. The autocorrelation function of the underlying Gaussian random field is an isotropic squared-exponential with length scale $l = 0.2$. The random field $\kappa(\boldsymbol{x})$ is discretized using the EOLE method (Li and Der Kiureghian, 1993) with $d = 53$ terms, which comprises 99% of its variance. The solution to an individual heat diffusion problem is computed using an in-house finite element code (Konakli and Sudret, 2016). The input comprises $d = 53$ independent standard normal random variables.

Finally, the so-called 100D function is an analytical model of the form

$$\begin{aligned} f(\boldsymbol{X}) = 3 - \frac{5}{d}\sum_{i=1}^{d} iX_i + \frac{1}{d}\sum_{i=1}^{d} iX_i^3 + \frac{1}{3d}\sum_{i=1}^{d} i\ln\left(X_i^2 + X_i^4\right) \\ + X_1X_2^2 + X_2X_4 - X_3X_5 + X_{51} + X_{50}X_{54}^2 \end{aligned} \tag{13}$$

taking uniform inputs $X_i \sim \mathcal{U}([1, 2]), i \neq 20$, and $X_{20} \sim \mathcal{U}([1, 3])$. We use $d = 100$. This function was designed for sensitivity analysis: the first-order sensitivity indices of the input variables are generally nonlinearly increasing with their index, with certain variables having especially high sensitivity. The model also contains four interaction terms. It is an example from UQLab.[7]

For each of the models, we use a fixed basis for the benchmark. In general, the best total degree p and the hyperbolic truncation q are a priori unknown. We heuristically choose $q = 1$ for low-dimensional models ($d \leq 10$) and $q = 0.5$ for high-dimensional models ($d \geq 20$). The degree p is chosen so that the number of basis functions P is approximately $\frac{10}{3}N_{\max}$, where $N_{\max}$ is the largest number of experimental design points for the specific benchmark. This choice is based

[7] https://www.uqlab.com/sensitivity-high-dimension

on the reasoning that for an experimental design of size N, sparse solvers like LARS often select an active basis of size $\approx \frac{N}{3}$, and that the candidate basis might be 10 times larger than the final active basis to be sufficiently rich. We focus on rather small experimental designs, since our goal is not to investigate the convergence of the methods as $N \to \infty$ (which has been demonstrated elsewhere), but to decide which methods are most efficient for small N. This results in the choice of values for p, q, and $N_{\max}$ displayed in Table 1.

3.4 Results: Comparison of solvers

First, we use a fixed sampling scheme (LHS) to compare the performance of the six solvers LARS, OMP, subspace pursuit (SP) using k-fold cross-validation, Subspace Pursuit using LOO cross-validation (SP_{LOO}), FastLaplace (BCS), and SPGL1 on all 11 benchmark models described above.

In Figure 2 we display boxplots (50 replications) of relative MSE against experimental design size for all six solvers for the four spotlight models. For the remaining seven models, the corresponding boxplots (30 replications) of relative MSE against experimental design size are provided in Figure 10 in Appendix C. In the plots, the lines as well as the dot inside the white circle denote the median of the relative MSE. We make the following observations:

- For the smallest experimental designs, all solvers perform similarly poorly: there is not enough information in the ED to construct an accurate surrogate model. For larger experimental designs, there can be considerable differences between the solvers' generalization errors of up to several orders of magnitude, which demonstrates that the solvers do not use the available information in identical ways.

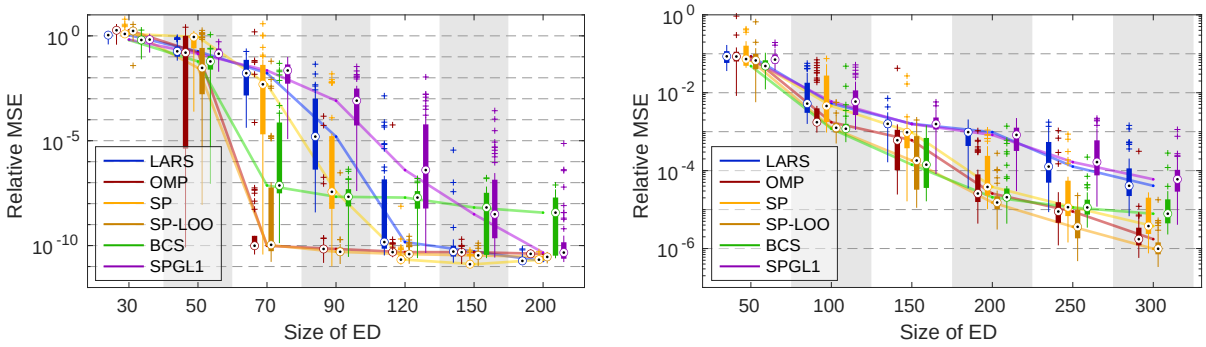
- BCS, SP_{LOO}, and OMP are often among the best solvers. BCS performs especially well for smaller experimental design sizes. In the case of the Ishigami model, it seems to plateau earlier than the other solvers. It also tends to find sparser solutions than the other solvers (not shown in plots), which might explain these observations: sparse solutions are advantageous when only limited data is available, but at the same time they carry the risk of ignoring important terms. For large ED sizes and the two highly compressible models (Ishigami and 100D function), BCS has a larger spread than the other solvers, possibly because the sparsity-enforcing procedure does not always include all of the important terms. In contrast, the greedy solver OMP returns rather dense solutions, which seem to generalize well. SP_{LOO} performs well in general for low-dimensional models.

- SP does not perform well for small ED sizes, but for large ED sizes it sometimes outperforms the other solvers. Together with BCS, it tends to find sparser solutions than the other solvers (not shown in plots).

- LARS and SPGL1 often achieve a similar generalization error, which is often larger, sometimes significantly, than that of the other solvers. SPGL1 tends to return rather dense solutions (not shown in plots), which might not generalize as well as other solutions.

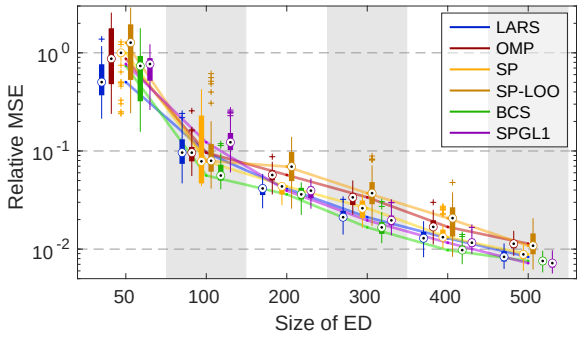
- Some models characterized by relatively poor compressibility (e.g. diffusion and frame) show comparable performance among all solvers. This is expected, as in such cases the sparsity assumption is a rather weak proxy for solution quality.

- The exceptions to the general observations outlined above are the damped oscillator and the Morris function, for which the solver performance is reversed, with LARS and SPGL1 among the best solvers, and OMP and SP_{LOO} among the worst. In these cases, however, none of the methods achieves satisfactory accuracy within the available computational budget.

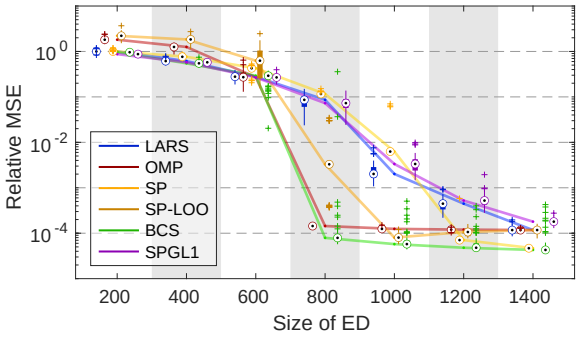


(a) Ishigami function

(b) Borehole model



(c) Two-dimensional diffusion



(d) 100D function

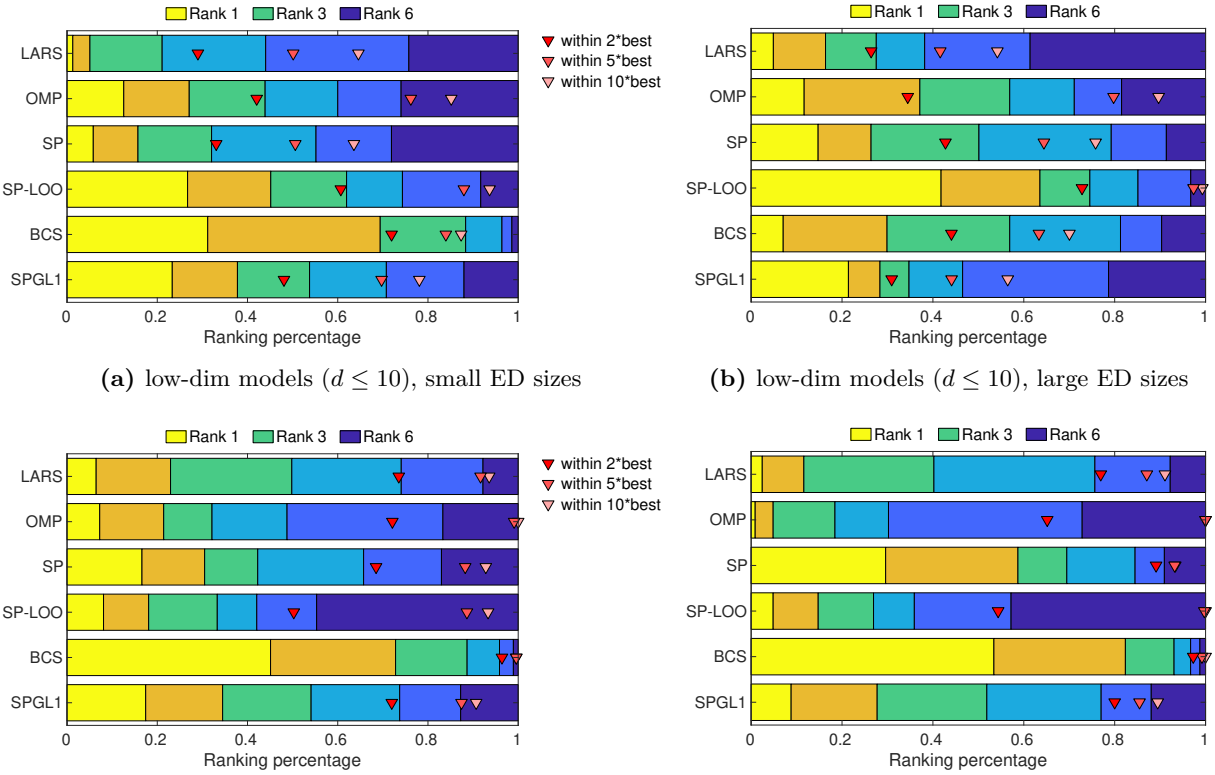
Figure 2: Boxplots of relative MSE against ED size from the solver benchmark for the four spotlight models. Results for six sparse solvers and LHS design. For the corresponding plots for the seven remaining models, see Figure 10 in Appendix C.

To objectively assess the performance of the methods, we now aggregate the results across models. Since all solvers are tested on the same set of experimental designs, we can determine the ranking of solvers for each experimental design (50 replications $\times$ $6-7$ ED sizes for the four spotlight models, and 30 replications $\times$ $5-7$ ED sizes for each of the seven additional models, resulting in 2620 PCEs) and count how often each solver achieved each rank. This is displayed in Figure 3 in the form of stacked bar plots, where the counts are given as percentages. The counts have been normalized by the number of replications and ED sizes used for each model, so that each of the models contributes equally to the final percentages.

This ranking alone, however, does not provide a complete picture; e.g., a solver ranked last can

be off by orders of magnitude or barely worse than the best-performing one. Therefore, we added an additional set of triangle markers detailing for how many of the EDs the respective solver returned a result that was within two, five, or 10 times the smallest relative MSE attained by any of the six solvers on the same ED. For example, the red triangle in the top row of Figure 3a indicates that in ca. 30% of the runs, the solution returned by LARS had a validation error that was at most twice as large as that of the best solution on the same ED.

We have grouped the analysis results separately for the 6 low-dimensional ($d \leq 10$) and the 5 high-dimensional ($d \geq 20$) models, because we observed that dimension had a significant impact on the rankings, and this information is readily available even for black-box models. We also analyze small and large ED sizes separately, where the first half of considered ED sizes (3–4) are regarded small, and the second half large.



(a) low-dim models ($d \leq 10$), small ED sizes

(b) low-dim models ($d \leq 10$), large ED sizes

(c) high-dim models ($d \geq 20$), small ED sizes

(d) high-dim models ($d \geq 20$), large ED sizes

Figure 3: Aggregated results for the solver benchmark (Section 3.4), separately for low-dimensional (a),(b) and high-dimensional models (c),(d). Left: small ED sizes. Right: large ED sizes. For each model and experimental design, the ranking of the six solvers is determined. The stacked bar plots visualize how often each solver achieved the respective rank. The triangle markers in hues of red additionally demonstrate in how many runs the obtained relative MSE was within a factor of $\{2, 5, 10\}$ of the smallest relative MSE achieved on this experimental design.

We make the following observations:

- *Low-dimensional models, small ED sizes*: BCS is the best solver most often (31% of runs) and also most often within two times the smallest error (72% of runs). However, it comes

within one order of magnitude of the smallest relative MSE (a property which we here call *robustness*) in only 87% of runs, while SP_{LOO} achieves this in 94% of runs. In the two former metrics, SP_{LOO} comes second. LARS and SP perform worst, followed by OMP and SPGL1. Here and in the following, we observe that OMP and SP_{LOO} often result in quite robust solutions.

- *Low-dimensional models, large ED sizes*: SP_{LOO} outperforms the other solvers in all respects. It provides the smallest relative MSE of all solvers in 42% of runs, is in 73% of runs within two times of the smallest relative MSE, and is even in 99% of runs within one order of magnitude of the smallest relative MSE. For the other solvers, the statistics confirm the observations highlighted by the spotlight models: LARS and SPGL1 overall do not perform well. SP and BCS come close to the best solution quite often, but are less robust, whereas OMP is robust but often not as close to the best solution.

- *High-dimensional models*: For the small as well as the large ED sizes, we see that BCS performs exceptionally well. It is the best solver in 45% (53%) of runs and comes within two times the smallest relative MSE in even 96% (97%) of runs. Both BCS and OMP attain in *all* cases a relative MSE within 10 times the smallest relative MSE. SP performs better for large rather than small ED sizes. While SP_{LOO} performed best for the low-dimensional models, here it shows poor performance. Note that all solvers come within 10 times the smallest relative MSE in more than 89% of all runs, showing that the choice of solver has a smaller impact for high-dimensional than for low-dimensional models.

3.5 Results: Comparison of sampling schemes together with solvers

We pair the five solvers LARS, OMP, SP, SP_{LOO}, and BCS[8] with the sampling schemes MC, LHS, coherence-optimal, and D-optimal based on a coherence-optimal candidate set. We use the abbreviations coh-opt and D-opt(coh-opt) for the latter two. Since SP_{LOO} performed poorly for high-dimensional models in the solver benchmark of the previous section, we do not consider it here for the high-dimensional models.

We run the benchmark for the low-dimensional models Ishigami, undamped oscillator, borehole, damped oscillator, and wingweight function, and for the high-dimensional models Morris function, two-dimensional diffusion, one-dimensional diffusion, and 100D function. The truss model has Gumbel input, for which we (as of now) cannot construct a coherence-optimal sample. The same holds for the structural frame model with its dependent input.

Detailed boxplots of the relative MSE against ED size for the spotlight models, showing how each solver performed when paired with the sampling schemes, can be found in Appendix C, Figures 11–14. For the sake of readability, in this section we only show results that are aggregated over models, separately for the low- and high-dimensional cases. For every model and repetition index, we determine the relative ranking of the 20 (16) combinations (5 (4) solvers $\times$ 4 sampling

[8]Since SPGL1 did not perform well in the previous section, and is quite slow, we do not include it further in this benchmark.

strategies). We also determine which of the combinations came within a factor of $\{2, 5, 10\}$ of the smallest relative MSE among this set (robustness). Then, we count how often each combination achieved each rank, and how often each combination achieved a relative MSE within a factor of the smallest relative MSE. The results are displayed in Figure 4 in the form of stacked bar plots for the ranks, with triangle markers denoting the percentage of robust runs. The combinations are sorted by the percentage of runs in which they achieved a relative MSE within two times the smallest relative MSE, because we find that this metric is a good compromise between performance and robustness. We analyze small and large ED sizes separately, where the first half of considered ED sizes (3–4) are regarded as small and the second half as large.

Note that, as opposed to the aggregated results in Figure 3, where the solvers are compared on the same experimental designs, here the comparison is done on *different* experimental designs, which are matched randomly.[9] Our results in Figure 4 are bootstrapped four times using random permutations of the replication index, corresponding in total to 250 replications, to minimize the influence of this randomness (which is in any case not large, as can be seen from permutation tests).

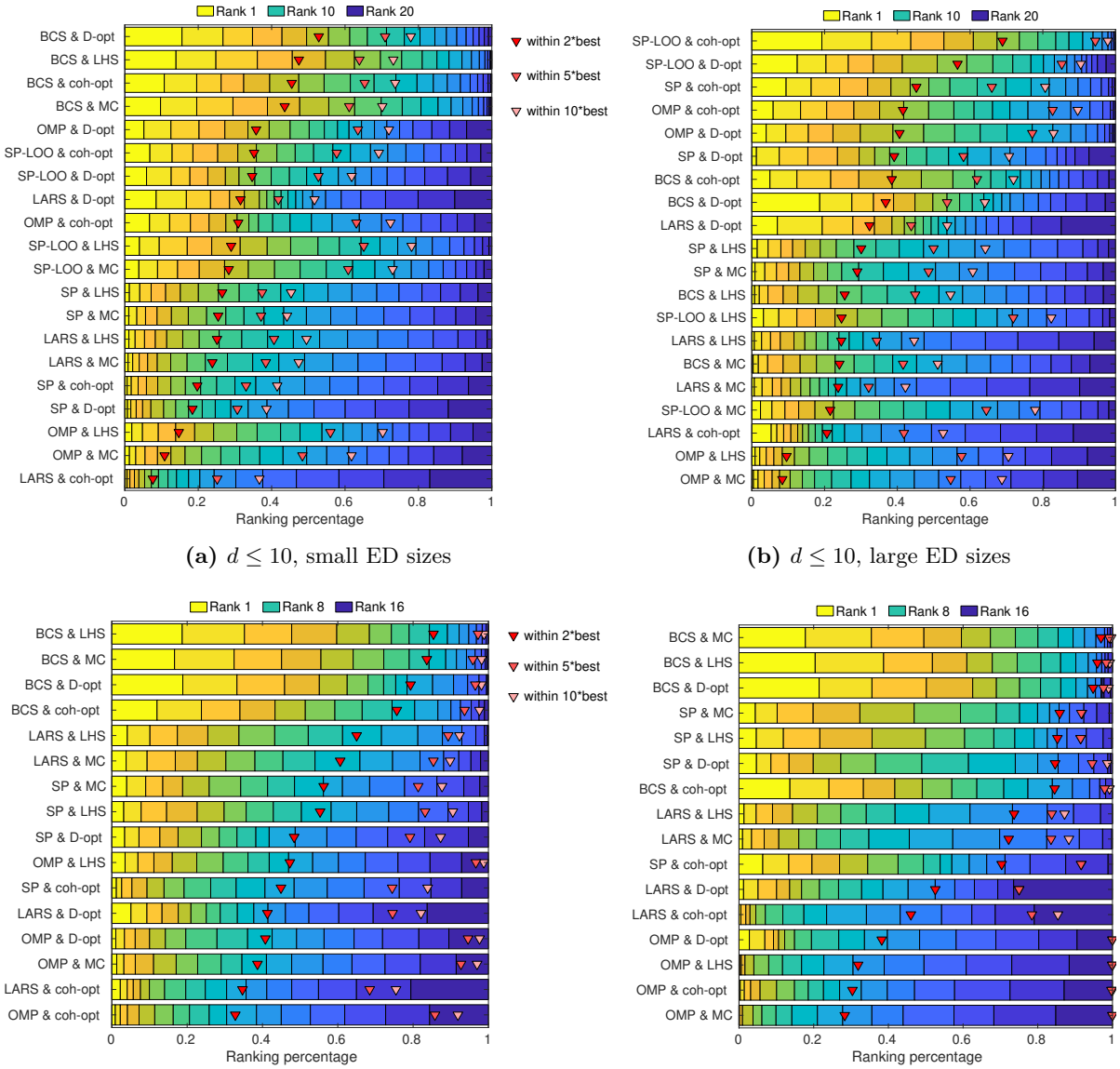
From Figure 4 and Figures 11–14, we make the following observations:

- There can be considerable differences in the performance of different combinations of solvers and sampling schemes. The differences are larger for low-dimensional models, visible in the spread of triangle markers in Figures 4a and 4b. For high-dimensional models, many combinations find an error that is close to the smallest error, which can be seen from the red triangle markers at high percentages in Figures 4c and 4d, and from the clustered boxplots in Figures 13 and 14, (e) and (f).

- MC and LHS perform comparably, and for the high-dimensional models almost identically. For the low-dimensional models, LHS sampling has, in most cases, median error and variability that are the same as or smaller than MC. This is consistent with the literature (Shields and Zhang, 2016). These observations are confirmed by the plots in Figure 4: for almost every solver, the combination with LHS is slightly better than the corresponding one with MC in each of the metrics.

- The advanced sampling schemes coh-opt and D-opt(coh-opt) show a clear advantage over MC and LHS sampling for low-dimensional models and large experimental designs (consistent with theoretical considerations and numerical experiments (Hampton and Doostan, 2015b)). For low-dimensional models and small experimental designs, they show mixed performance; for high-dimensional models, they perform the same as or worse than LHS and MC.

[9] The relative MSE of each combination is interpreted as a random variable $E_{\text{solver}}^{\text{sampling}}$, where the randomness is induced by the randomness in the experimental design. E.g., $E_{\text{BCS}}^{\text{LHS}}$ is the random variable of relative MSE attained by BCS applied to an LHS design of specified size. The reference error ("smallest relative MSE") E^* is a random variable as well, defined as the minimum over *one realization* of each combination of methods: $E^* = \min_{s \in \text{sampling}, t \in \text{solvers}} E_s^t$. The plots in Figure 4 are therefore read as follows: e.g., in the low-dimensional, small ED size case (Figure 4a): $\mathbb{P}\left(E_{\text{BCS}}^{\text{D-opt}} = E^*\right) = 0.16$, $\mathbb{P}\left(E_{\text{BCS}}^{\text{D-opt}} \leq 10E^*\right) = 0.78$.



(a) $d \leq 10$, small ED sizes

(b) $d \leq 10$, large ED sizes

(c) $d \geq 20$, small ED sizes

(d) $d \geq 20$, large ED sizes

Figure 4: Aggregated results for the five low-dimensional models Ishigami, undamped oscillator, borehole, damped oscillator, and wingweight (top), and for the four high-dimensional models Morris function, structural frame, two-dimensional diffusion, and 100D function (bottom). Separately for small (a),(c) and large (b),(d) experimental designs. For the low-dimensional (high-dimensional) case, we investigate five (four) solvers and four sampling schemes, resulting in 20 (16) combinations. For each model and repetition, the ranking of all the combinations is determined (note that as opposed to Figure 3, here the comparison is done on different EDs, which are matched randomly. Results are bootstrapped four times by random permutations to increase robustness). The stacked bar plots visualize how often each combination achieved the respective rank. The triangle markers in hues of red additionally demonstrate in how many runs the obtained relative MSE was within a factor of $\{2, 5, 10\}$ of the smallest relative MSE achieved in this comparison. The combinations are sorted by the percentage of runs in which they achieved a relative MSE within two times the smallest relative MSE of the respective random pairing (red triangle marker). Plots of relative MSE against ED size for the spotlight models can be found in Appendix C, Figures 11–14.

It is known that coh-opt sampling leads to a greater improvement over MC sampling for low-dimensional, high-degree expansions than for high-dimensional, low-degree expansions (Hampton and Doostan, 2015b; Alemazkoor and Meidani, 2018a). Note also that all numerical experiments in the literature testing coh-opt sampling were performed in $d \leq 30$ dimensions, using only models with uniform input, or manufactured sparse PCE, i.e., polynomial models with an exactly sparse representation (Hampton and Doostan, 2015b,a; Alemazkoor and Meidani, 2018a; Diaz et al., 2018).

Both coh-opt and D-opt are sampling methods that aim to improve properties of the regression matrix. They are adapted to the candidate basis. If the candidate basis is large and contains many regressors that are not needed in the final sparse expansion, this adaptation might even deteriorate the solution.

- BCS is one of the best-performing solvers, almost regardless of sampling scheme. The exceptions are low-dimensional models with large experimental designs, where SP_{LOO} with coh-opt sampling outperforms all other solvers. This might be related to BCS plateauing earlier than other solvers (see Figure 2a and 2b). It seems BCS is preferable whenever the information content is low (small ED sizes or high-dimensional models).

- OMP and SP_{LOO} are generally quite robust (within one order of magnitude of the best solution). However, OMP often does not come close to the best solution, especially when paired with LHS or MC. BCS is more robust for high-dimensional models than for low-dimensional models. LARS and SP show mixed performance, with LARS being one of the least robust solvers.

- Aggregating the results for each sampling scheme separately (not shown), we observe that the behavior of the solvers is very similar in terms of ranking and robustness to the behavior observed on LHS (Figure 3), suggesting that the ranking of solvers is mostly independent of the sampling scheme.

Note that the results in Figures 11, 12, and 14 exhibit plateauing for larger sample sizes. This indicates that the maximal accuracy achievable with this set of basis functions has been reached. Using a larger basis might lead to more accurate solutions, if it contains an important regressor that was previously missing. However, note that a larger basis can also lead to less accurate solutions: when the experimental design size is held fixed while a larger basis is used, the ratio of experimental design points to basis functions is smaller, and the properties of the regression matrix might deteriorate.

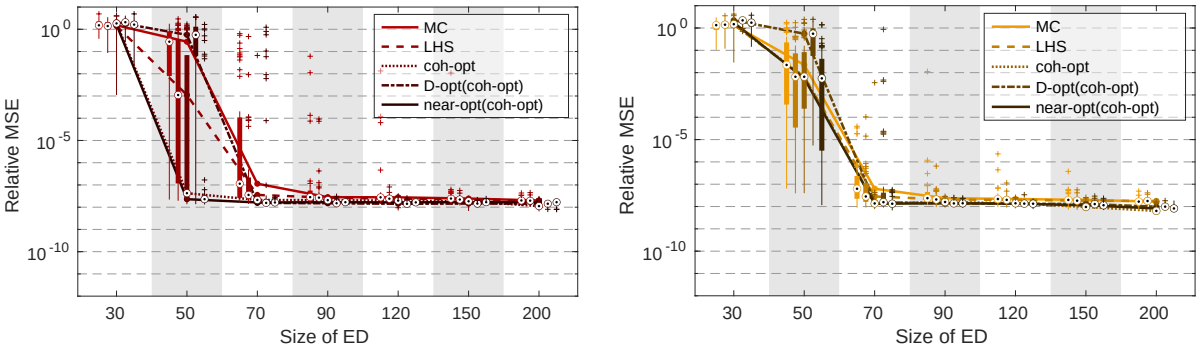
3.6 Results: Comparison of sampling schemes together with solvers, using a smaller candidate basis

We repeat the experiments from the previous section for the Ishigami and borehole models, using a smaller candidate basis for which near-optimal sampling is feasible. The tested solvers

are LARS, OMP, SP, SP_{LOO}, and BCS. We use the sampling schemes MC, LHS, coh-opt, D-opt(coh-opt) and near-opt(coh-opt). Boxplots of relative MSE against ED size are shown in Figures 5 and 6. For the sake of conciseness, we only show the combinations involving OMP and SP_{LOO}. The remaining plots are provided in Appendix C, Figure 15.

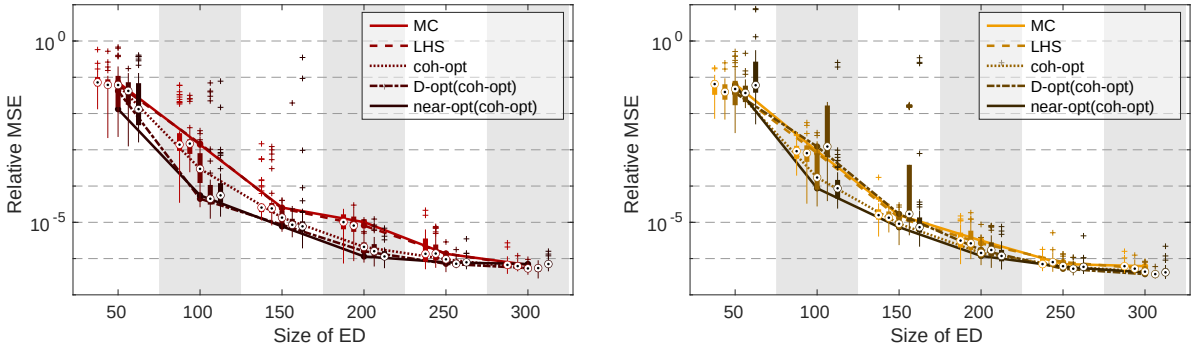
We observe the following:

- Since the basis is smaller, the relative MSE reaches a plateau already for smaller experimental design sizes.
- Most qualitative observations regarding solver and sampling performance are the same as in the previous section, where a larger basis was used.
- Near-optimal sampling often achieves the same or a slightly smaller error than coh-opt sampling, which is consistent with (Alemazkoor and Meidani, 2018a). In many cases, near-optimal sampling achieves the smallest median error. For the Ishigami model, near-optimal sampling additionally exhibits small variability, while for the borehole model, it has a rather large spread, i.e., several outliers.



(a) OMP **(b)** SP_{LOO}

Figure 5: Results for the Ishigami model with a smaller basis ($d = 3, p = 12, q = 1$). Results for two sparse solvers and five experimental design schemes. 50 replications. For the remaining plots, see Figure 15 in Appendix C.



(a) OMP **(b)** $\mathrm{SP}_{\mathrm{LOO}}$

Figure 6: Results for the borehole model with a smaller basis ($d = 8, p = 4, q = 1$). Results for two sparse solvers and five experimental design schemes. 50 replications. For the remaining plots, see Figure 15 in Appendix C.

4 Discussion and conclusions

In this paper, we investigated sparse PCE methods with the goal of computing accurate surrogate models based on a few model evaluations.

We presented a literature survey and a framework describing the general computation procedure for sparse PCE. We have seen that the existing literature on sparse PCE can be fit into this framework and that methods developed for different components of the framework can naturally be combined.

In order to give recommendations to practitioners who want to use sparse PCE surrogates for their applications, we performed a numerical benchmark based on 11 example functions which are intended to resemble real-world engineering problems presenting different challenges. We tested several popular sparse solvers and sampling schemes on a fixed set of basis functions, using a range of experimental design sizes and 30–50 replications, and made the following observations:

- The choice of sampling scheme and sparse regression solver can make a difference of up to several orders of magnitude for the relative MSE. Mostly, the rankings of solvers and sampling schemes seem to be independent of one another: an experimental design that works best for one solver will also perform well with other solvers, and the ranking of solvers looks similar independent of which sampling scheme is used. Both solvers and sampling schemes make a greater difference for low-dimensional models.

- For low-dimensional models and small ED sizes, the solver BCS performs best, regardless of sampling scheme (with D-opt(coh-opt) being slightly preferable), while the solver $\mathrm{SP}_{\mathrm{LOO}}$ (a variant of SP) appears to be especially robust.

- For low-dimensional models and large ED sizes, $\mathrm{SP}_{\mathrm{LOO}}$ together with coherence-optimal sampling outperforms all other combinations.

- For low-dimensional models, and when the basis is small enough to make it feasible, near-optimal sampling outperforms all other sampling schemes, regardless of the solver.
- For high-dimensional models, BCS is by far the best solver. All solvers perform better when paired with LHS; in other words, no advanced sampling scheme appears competitive compared to LHS for such problems, whatever the solver used.

The benchmark results demonstrate that in costly, real-world applications it is worth choosing the sparse PCE training strategy carefully, since the methods can make a substantial difference in the quality of the resulting surrogate. While a more accurate surrogate model is generally desirable, in industrial applications it might have a higher impact for purposes such as optimization, rather than, e.g., sensitivity analysis.

Our conclusions are based on a number of benchmark models, which we consider representative of engineering models in terms of dimensionality and complexity. Naturally, however, no selection of models can cover the whole space of engineering models. Further work would be required to understand the connection among model properties, basis choice, experimental design size, and sparse PCE techniques like solvers and sampling schemes.

All results were obtained using a fixed basis based on a heuristic choice (see Section 3.3). Generally, when the optimal degree of the basis is unknown, degree adaptivity (based on a cross-validation error) can be a useful strategy. Due to time and space constraints, this was not investigated in the present work. Adaptivity critically depends on the availability of an accurate error estimator. Some of the best solvers in this study (i.e., OMP and BCS) tend to underestimate the generalization error (not shown in the plots), which might be a drawback in the setting of adaptive degree selection and might change the effect and ranking of solvers and sampling methods. For a detailed discussion and benchmark of basis-adaptive schemes, we refer the reader to (Lüthen et al., 2021).

As evident from the extensive literature on the topic, sparse PCE is an already well-established technique, as well as an active field of methodological research. Recent innovations include Bayesian techniques for sparse PCE and the identification of suitable rotated coordinates for the expansion. Such innovative ideas are expected to lead to further improvements in the computation of sparse PCE, which will in turn benefit applications as well as all advanced methods that use sparse PCE as one of their building blocks (see, e.g., (Schöbi et al., 2015; Chatterjee et al., 2019; Zhu and Sudret, 2020; Marelli et al., 2021)).

PCE is a popular metamodelling tool in the engineering community, and many different methods are available. Up to now, the choice of which of the many PCE methods to apply was mostly left to chance or the personal experience of the practitioner. In our benchmark, we explored a significant set of methods that have received attention in the past few years. We hope that this work can serve as a basis for further benchmarking efforts, in order to identify which of the many available methods are most suitable for real-world problems. These might include sequential enrichment of experimental design, Gaussian adaptation of the input space, stepwise

regression algorithms and many other ideas for sparse solvers, as well as methods for extremely high-dimensional problems.

Our benchmark code is available on request. The solvers BCS and SP_{LOO} will be made available in the 1.4.0 release of UQLab. For a description of how to add custom sampling schemes and sparse solvers for PCE to UQLab, we refer the reader to the supplementary material accompanying this paper. To facilitate easier benchmarking of PCE techniques on a large number of examples in a standardized setup, we are actively engaging in designing and developing a benchmarking platform for surrogate modelling methods similar to the UCI machine learning repository[10] or the structural reliability platform RPrepo[11] where data sets, models, and methods can be made available for testing and benchmarking.

Acknowledgments

We thank Dr. Emiliano Torre for helpful discussions.

References

Abraham, S., M. Raisee, G. Ghorbaniasl, F. Contino, and F. Lacor (2017). A robust and efficient stepwise regression method for building sparse polynomial chaos expansions. *J. Comput. Phys. 332*, 461 – 474.

Alemazkoor, N. and H. Meidani (2017). Divide and conquer: An incremental sparsity promoting compressive sampling approach for polynomial chaos expansions. *Comput. Methods Appl. Mech. Engrg. 318*, 937–956.

Alemazkoor, N. and H. Meidani (2018a). A near-optimal sampling strategy for sparse recovery of polynomial chaos expansions. *J. Comput. Phys. 371*, 137–151.

Alemazkoor, N. and H. Meidani (2018b). A preconditioning approach for improved estimation of sparse polynomial chaos expansions. *Comput. Methods Appl. Mech. Engrg. 342*, 474–489.

Arjoune, Y., N. Kaabouch, H. El Ghazi, and A. Tamtaoui (2017). Compressive sensing: Performance comparison of sparse recovery algorithms. In *2017 IEEE CCWC*, pp. 1–7. IEEE.

Babacan, D. (2011). MATLAB code for the TIP 2010 paper "Bayesian Compressive Sensing using Laplace Priors". http://www.dbabacan.info/software.html. [Online; accessed 28-August-2019].

Babacan, S., R. Molina, and A. Katsaggelos (2010). Bayesian compressive sensing using Laplace priors. *IEEE Trans. Image Process. 19*(1), 53–63.

[10] https://archive.ics.uci.edu

[11] https://rprepo.readthedocs.io/en/latest/reliability_problems.html

Baptista, R., V. Stolbunov, and P. B. Nair (2019). Some greedy algorithms for sparse polynomial chaos expansions. *J. Comput. Phys. 387*, 303–325.

Baumann, D. and K. Baumann (2014). Reliable estimation of prediction errors for QSAR models under model uncertainty using double cross-validation. *J. Cheminf. 6*(1), 47.

Berchier, M. (2015). Orthogonal matching pursuit for sparse polynomial chaos expansions. ETH Zürich. Semester project.

Berveiller, M. (2005). *Stochastic finite elements : intrusive and non intrusive methods for reliability analysis*. Ph. D. thesis, Université Blaise Pascal, Clermont-Ferrand.

Berveiller, M., B. Sudret, and M. Lemaire (2006). Stochastic finite elements: a non intrusive approach by regression. *Eur. J. Comput. Mech. 15*(1–3), 81–92.

Bhattacharyya, B. (2020). Global sensitivity analysis: A Bayesian learning based polynomial chaos approach. *J. Comput. Phys. 415*, 109539.

Birgin, E. G., J. M. Martínez, and M. Raydan (2000). Nonmonotone spectral projected gradient methods on convex sets. *SIAM J. Optim. 10*(4), 1196–1211.

Blatman, G. and B. Sudret (2008). Sparse polynomial chaos expansions and adaptive stochastic finite elements using a regression approach. *Comptes Rendus Mécanique 336*(6), 518–523.

Blatman, G. and B. Sudret (2010a). An adaptive algorithm to build up sparse polynomial chaos expansions for stochastic finite element analysis. *Prob. Eng. Mech. 25*, 183–197.

Blatman, G. and B. Sudret (2010b). Efficient computation of global sensitivity indices using sparse polynomial chaos expansions. *Reliab. Eng. Sys. Safety 95*, 1216–1229.

Blatman, G. and B. Sudret (2011). Adaptive sparse polynomial chaos expansion based on Least Angle Regression. *J. Comput. Phys 230*, 2345–2367.

Bruckstein, A. M., D. L. Donoho, and M. Elad (2009). From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM review 51*(1), 34–81.

Candès, E. and J. Romberg (2005). ℓ1-MAGIC. https://statweb.stanford.edu/~candes/software/l1magic/. [Online; accessed 22-January-2020].

Candès, E., J. Romberg, and T. Tao (2006). Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Inform. Theory 52*(2), 489–509.

Candès, E. J. and Y. Plan (2011). A probabilistic and RIPless theory of compressed sensing. *IEEE Trans. Inform. Theory 57*(11), 7235–7254.

Candès, E. J. and M. B. Wakin (2008). An introduction to compressive sampling: A sensing/sampling paradigm that goes against the common knowledge in data acquisition. *IEEE Signal Process. Mag. 25*(2), 21–30.

Candès, E. J., M. B. Wakin, and S. P. Boyd (2008). Enhancing sparsity by reweighted ℓ_1 minimization. *J. Fourier Anal. Appl. 14*(5-6), 877–905.

Carron, I. (2013). Compressive Sensing: The Big Picture. https://sites.google.com/site/igorcarron2/cs. [Online; accessed 22-January-2020].

Chapelle, O., V. Vapnik, and Y. Bengio (2002). Model selection for small sample regression. *Machine Learning 48*(1), 9–23.

Chatterjee, T., S. Chakraborty, and R. Chowdhury (2019). A critical review of surrogate assisted robust design optimization. *Arch. Comput. Method. E. 26*(1), 245–274.

Cheng, K. and Z. Lu (2018a). Adaptive sparse polynomial chaos expansions for global sensitivity analysis based on support vector regression. *Comput. Struct. 194*, 86–96.

Cheng, K. and Z. Lu (2018b). Sparse polynomial chaos expansion based on D-MORPH regression. *Appl. Math. Comput. 323*, 17–30.

Cohen, A. and G. Migliorati (2017). Optimal weighted least-squares methods. *SMAI J. Comp. Math. 3*, 181–203.

Cook, R. D. and C. J. Nachtsheim (1980). A comparison of algorithms for constructing exact D-optimal designs. *Technometrics 22*(3), 315–324.

Dai, W. and O. Milenkovic (2009). Subspace pursuit for compressive sensing signal reconstruction. *IEEE Trans. Inform. Theory 55*(5), 2230–2249.

Diaz, P. (2018). DOPT_PCE. https://github.com/CU-UQ/DOPT_PCE. [Online; accessed 22-January-2020].

Diaz, P., A. Doostan, and J. Hampton (2018). Sparse polynomial chaos expansions via compressed sensing and D-optimal design. *Comput. Methods Appl. Mech. Engrg. 336*, 640–666.

Donoho, D., I. Drori, V. Stodden, Y. Tsaig, and M. Shahram (2007). SparseLab - Seeking Sparse Solutions to Linear Systems of Equations. http://sparselab.stanford.edu/. [Online; accessed 22-January-2020].

Donoho, D. L. (2006). Compressed sensing. *IEEE Trans. Inform. Theory 52*(4), 1289–1306.

Doostan, A. and H. Owhadi (2011). A non-adapted sparse approximation of PDEs with stochastic inputs. *J. Comput. Phys. 230*(8), 3015–3034.

Dubourg, V. (2011). *Adaptive surrogate models for reliability analysis and reliability-based design optimization*. Ph. D. thesis, Université Blaise Pascal, Clermont-Ferrand, France.

Dutta, S. and A. H. Gandomi (2020). Design of experiments for uncertainty quantification based on polynomial chaos expansion metamodels. In *Handbook of Probabilistic Models*, pp. 369–381. Elsevier.

Dykstra, O. (1971). The augmentation of experimental data to maximize $[X'X]$. *Technometrics 13*(3), 682–688.

Echard, B., N. Gayton, M. Lemaire, and N. Relun (2013). A combined importance sampling and Kriging reliability method for small failure probabilities with time-demanding numerical models. *Reliab. Eng. Syst. Safety 111*, 232–240.

Efron, B., T. Hastie, I. Johnstone, and R. Tibshirani (2004). Least angle regression. *Ann. Stat. 32*, 407–499.

Ernst, O., A. Mugler, H.-J. Starkloff, and E. Ullmann (2012). On the convergence of generalized polynomial chaos expansions. *ESAIM: Math. Model. Numer. Anal. 46*(02), 317–339.

Fajraoui, N., S. Marelli, and B. Sudret (2017). Sequential design of experiment for sparse polynomial chaos expansions. *SIAM/ASA J. Unc. Quant. 5*(1), 1061–1085.

Faul, A. C. and M. E. Tipping (2002). Analysis of sparse Bayesian learning. In *Advances in neural information processing systems*, pp. 383–389.

Fedorov, V. V. (2013). *Theory of optimal experiments*. Elsevier.

Figueiredo, M. A. (2003). Adaptive sparseness for supervised learning. *IEEE Trans. Pattern Anal. Mach. Intell. 25*(9), 1150–1159.

Figueiredo, M. A. and R. D. Nowak (2001). Wavelet-based image estimation: an empirical Bayes approach using Jeffrey's noninformative prior. *IEEE Trans. Image Process. 10*(9), 1322–1331.

Forrester, A., A. Sobester, and A. Keane (2008). *Engineering design via surrogate modelling: a practical guide*. Wiley.

Ghanem, R. G. and P. Spanos (1991). *Stochastic finite elements – A spectral approach*. Springer Verlag, New York. (Reedited by Dover Publications, Mineola, 2003).

Gu, M. and S. C. Eisenstat (1996). Efficient algorithms for computing a strong rank-revealing QR factorization. *SIAM J. Sci. Comput. 17*(4), 848–869.

Guo, L., A. Narayan, Y. Liu, and T. Zhou (2020). Sparse approximation of data-driven polynomial chaos expansions: An induced sampling approach. *Commun. Math. Res. 36*(2), 128–153.

Guo, L., A. Narayan, T. Zhou, and Y. Chen (2017). Stochastic collocation methods via ℓ_1 minimization using randomized quadratures. *SIAM J. Sci. Comput. 39*(1), A333–A359.

Hadigol, M. and A. Doostan (2018). Least squares polynomial chaos expansion: A review of sampling strategies. *Comput. Methods Appl. Mech. Engrg. 332*, 382–407.

Halton, J. H. (1960). On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numer. Math. 2*(1), 84–90.

Hampton, J. and A. Doostan (2015a). Coherence motivated sampling and convergence analysis of least squares polynomial chaos regression. *Comput. Methods Appl. Mech. Engrg. 290*, 73–97.

Hampton, J. and A. Doostan (2015b). Compressive sampling of polynomial chaos expansions: Convergence analysis and sampling strategies. *J. Comput. Phys. 280*, 363–386.

Hampton, J. and A. Doostan (2017). COH-OPT. https://github.com/CU-UQ/COH-OPT. [Online; accessed 22-January-2020].

Hampton, J. and A. Doostan (2018). Basis adaptive sample efficient polynomial chaos (BASE-PC). *J. Comput. Phys. 371*, 20–49.

Harper, W. V. and S. K. Gupta (1983). Sensitivity/uncertainty analysis of a borehole scenario comparing Latin hypercube sampling and deterministic sensitivity approaches. Technical Report No. BMI/ONWI-516, Battelle Memorial Inst. - Office of Nuclear Waste Isolation, Columbus, OH (USA).

Hastie, T., R. Tibshirani, and J. Friedman (2001). *The elements of statistical learning: Data mining, inference and prediction*. Springer, New York.

Hong, Y. P. and C.-T. Pan (1992). Rank-revealing QR factorizations and the singular value decomposition. *Math. Comput. 58*(197), 213–232.

Hosder, S., R. Walters, and M. Balch (2007). Efficient sampling for non-intrusive polynomial chaos applications with multiple uncertain input variables. In *48th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, pp. 1939.

Hu, C. and B. D. Youn (2011). Adaptive-sparse polynomial chaos expansion for reliability analysis and design of complex engineering systems. *Struct. Multidisc. Optim. 43*, 419–442.

Hu, R. and M. Ludkovski (2017). Sequential design for ranking response surfaces. *SIAM/ASA J. Unc. Quant. 5*(1), 212–239.

Huan, X., C. Safta, K. Sargsyan, Z. P. Vane, G. Lacaze, J. C. Oefelein, and H. N. Najm (2018). Compressive sensing with cross-validation and stop-sampling for sparse polynomial chaos expansions. *SIAM/ASA J. Unc. Quant. 6*(2), 907–936.

Isukapalli, S. S. (1999). *Uncertainty Analysis of Transport-Transformation Models*. Ph. D. thesis, The State University of New Jersey.

Jakeman, J. D., M. S. Eldred, and K. Sargsyan (2015). Enhancing ℓ_1-minimization estimates of polynomial chaos expansions using basis selection. *J. Comput. Phys. 289*, 18–34.

Jakeman, J. D., A. Narayan, and T. Zhou (2017). A generalized sampling and preconditioning scheme for sparse approximation of polynomial chaos expansions. *SIAM J. Sci. Comput. 39*(3), A1114–A1144.

Ji, S., Y. Xue, and L. Carin (2008). Bayesian compressive sensing. *IEEE Trans. Signal Process. 56*(6), 2346–2356.

Kiefer, J. and J. Wolfowitz (1959). Optimum designs in regression problems. *Ann. Math. Stat. 30*(2), 271–294.

Konakli, K. and B. Sudret (2016). Global sensitivity analysis using low-rank tensor approximations. *Reliab. Eng. Sys. Safety 156*, 64–83.

Kougioumtzoglou, I. A., I. Petromichelakis, and A. F. Psaros (2020). Sparse representations and compressive sampling approaches in engineering mechanics: A review of theoretical concepts and diverse applications. *Prob. Eng. Mech. 61*, 103082.

Lataniotis, C., S. Marelli, and B. Sudret (2019). Extending classical surrogate modelling to high dimensions through supervised dimensionality reduction: A data-driven approach. *Int. J. Uncertain. Quantif.*. accepted.

Li, C. C. and A. Der Kiureghian (1993). Optimal discretization of random fields. *J. Eng. Mech. 119*(6), 1136–1154.

Li, G. and H. Rabitz (2010). D-MORPH regression: application to modeling with unknown parameters more than observation data. *J. Math. Chem. 48*(4), 1010–1035.

Liu, Z., D. Lesselier, B. Sudret, and J. Wiart (2020a). Surrogate modeling based on resampled polynomial chaos expansions. *Reliab. Eng. Sys. Safety 202*, 107008.

Liu, Z., D. Lesselier, B. Sudret, and J. Wiart (2020b). Surrogate modeling of indoor down-link human exposure based on sparse polynomial chaos expansion. *Int. J. Uncertainty Quantification 10*(2), 145–163.

Lüthen, N., S. Marelli, and B. Sudret (2021). A benchmark of basis-adaptive sparse polynomial chaos expansions for engineering regression problems. *Int. J. Uncertainty Quantification*. (submitted).

Marelli, S. and B. Sudret (2014). UQLab: A framework for uncertainty quantification in Matlab. In *Vulnerability, Uncertainty, and Risk (Proc. 2nd Int. Conf. on Vulnerability, Risk Analysis and Management (ICVRAM2014), Liverpool, United Kingdom)*, pp. 2554–2563.

Marelli, S. and B. Sudret (2019). UQLab user manual – Polynomial chaos expansions. Technical report, Chair of Risk, Safety & Uncertainty Quantification, ETH Zurich. Report# UQLab-V1.3-104.

Marelli, S., P.-R. Wagner, C. Lataniotis, and B. Sudret (2021). Stochastic spectral embedding. *International Journal for Uncertainty Quantification 11*(2), 25–47.

Mathelin, L. and K. Gallivan (2012). A compressed sensing approach for partial differential equations with random input data. *Commun. Comput. Phys. 12*(4), 919–954.

McKay, M. D., R. J. Beckman, and W. J. Conover (1979). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics 2*, 239–245.

Mikhalev, A. and I. V. Oseledets (2018). Rectangular maximum-volume submatrices and their applications. *Linear Algebra Appl. 538*, 187–211.

Montgomery, D. C. (2004). *Design and analysis of experiments*. John Wiley and Sons, New York.

Narayan, A., J. Jakeman, and T. Zhou (2017). A Christoffel function weighted least squares algorithm for collocation approximations. *Math. Comput. 86*(306), 1913–1947.

Needell, D. and J. A. Tropp (2009). CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. *Appl. Comput. Harmon. A. 26*(3), 301–321.

Nguyen, N.-K. and A. J. Miller (1992). A review of some exchange algorithms for constructing discrete D-optimal designs. *Comput. Stat. Data An. 14*(4), 489–498.

Owen, A. B. (1994). Controlling correlations in Latin hypercube samples. *J. Am. Stat. Assoc. 89*(428), 1517–1522.

Papaioannou, I., M. Ehre, and D. Straub (2019). PLS-based adaptation for efficient PCE representation in high dimensions. *J. Comput. Phys. 387*, 186–204.

Pati, Y. C., R. Rezaiifar, and P. S. Krishnaprasad (1993). Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Proc. of 27th Asilomar Conf. on signals, systems and computers*, pp. 40–44. IEEE.

Peng, J., J. Hampton, and A. Doostan (2014). A weighted ℓ_1-minimization approach for sparse polynomial chaos expansions. *J. Comput. Phys. 267*, 92–111.

Perkó, Z., L. Gilli, D. Lathouwers, and J. L. Kloosterman (2014). Grid and basis adaptive polynomial chaos techniques for sensitivity and uncertainty analysis. *J. Comput. Phys. 260*, 54–84.

Pronzato, L. and W. G. Müller (2012). Design of computer experiments: space filling and beyond. *Stat. Comput. 22*(3), 681–701.

Qaisar, S., R. M. Bilal, W. Iqbal, M. Naureen, and S. Lee (2013). Compressive sensing: From theory to applications, a survey. *J. Commun. Netw. 15*(5), 443–456.

Rauhut, H. and R. Ward (2012). Sparse Legendre expansions via ℓ_1-minimization. *J. Approx. Theory 164*(5), 517–533.

Sargsyan, K., C. Safta, H. Najm, B. Debusschere, D. Ricciuto, and P. Thornton (2014). Dimensionality reduction for complex models via Bayesian compressive sensing. *Int. J. Uncertain. Quantificat. 4*(1), 63–93.

Schöbi, R., B. Sudret, and J. Wiart (2015). Polynomial-chaos-based Kriging. *Int. J. Uncertainty Quantification 5*(2), 171–193.

Seeger, M. W. and H. Nickisch (2008). Compressed sensing and Bayesian experimental design. In *Proc. of the 25th Int. Conf. on Machine Learning*, pp. 912–919. ACM.

Seshadri, P., A. Narayan, and S. Mahadevan (2017). Effectively subsampled quadratures for least squares polynomial approximations. *SIAM/ASA J. Unc. Quant. 5*(1), 1003–1023.

Settles, B. (2012). Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning 6*(1), 1–114.

Shao, Q., A. Younes, M. Fahs, and T. Mara (2017). Bayesian sparse polynomial chaos expansion for global sensitivity analysis. *Comput. Meth. Appl. Mech. Eng. 318*, 474–496.

Shen, D., H. Wu, B. Xia, and D. Gan (2020). Polynomial chaos expansion for parametric problems in engineering systems: A review. *IEEE Syst. J.*.

Shields, M. D. and J. Zhang (2016). The generalization of Latin hypercube sampling. *Reliab. Eng. Syst. Saf. 148*, 96–108.

Shin, Y. and D. Xiu (2016a). Nonadaptive quasi-optimal points selection for least squares linear regression. *SIAM J. Sci. Comput. 38*(1), A385–A411.

Shin, Y. and D. Xiu (2016b). On a near optimal sampling strategy for least squares polynomial regression. *J. Comput. Phys. 326*, 931–946.

Sobol', I. M. (1967). Distribution of points in a cube and approximate evaluation of integrals. *U.S.S.R Comput. Maths. Math. Phys. 7*, 86–112.

Sudret, B. (2008). Global sensitivity analysis using polynomial chaos expansions. *Reliab. Eng. Sys. Safety 93*, 964–979.

Tang, G. and G. Iaccarino (2014). Subsampled Gauss quadrature nodes for estimating polynomial chaos expansions. *SIAM/ASA J. Unc. Quant. 2*(1), 423–443.

Tarakanov, A. and A. H. Elsheikh (2019). Regression-based sparse polynomial chaos for uncertainty quantification of subsurface flow models. *J. Comput. Phys. 399*, 108909.

Tipireddy, R. and R. Ghanem (2014). Basis adaptation in homogeneous chaos spaces. *J. Comput. Phys. 259*, 304–317.

Tipping, M. E. (2001). Sparse Bayesian learning and the relevance vector machine. *J. Mach. Learn. Res. 1*(Jun), 211–244.

Tipping, M. E. and A. C. Faul (2003). Fast marginal likelihood maximisation for sparse Bayesian models. In *Proc. 9th Int. Workshop on Artificial Intelligence and Statistics*.

Tropp, J. A. and A. C. Gilbert (2007). Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Trans. Inform. Theory 53*(12), 4655–4666.

Tsilifis, P., X. Huan, C. Safta, K. Sargsyan, G. Lacaze, J. C. Oefelein, H. N. Najm, and R. G. Ghanem (2019). Compressive sensing adaptation for polynomial chaos expansions. *J. Comput. Phys. 380*, 29–47.

Tsilifis, P., I. Papaioannou, D. Straub, and F. Nobile (2020). Sparse Polynomial Chaos expansions using variational relevance vector machines. *J. Comput. Phys. 416*, 109498.

van den Berg, E. and M. P. Friedlander (2008). Probing the Pareto frontier for basis pursuit solutions. *SIAM J. Sci. Comput. 31*(2), 890–912.

Van den Berg, E. and M. P. Friedlander (2015). SPGL1 - A Matlab solver for sparse optimization. https://friedlander.io/spgl1/. [Online; accessed 22-January-2020].

Wipf, D. P., J. Palmer, and B. D. Rao (2004). Perspectives on sparse Bayesian learning. In *Advances in neural information processing systems*, pp. 249–256.

Wipf, D. P. and B. D. Rao (2004). Sparse Bayesian learning for basis selection. *IEEE Trans. Signal Process. 52*(8), 2153–2164.

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

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

Yan, L., L. Guo, and D. Xiu (2012). Stochastic collocation algorithms using ℓ_1-minimization. *Int. J. Uncertain. Quantif. 2*(3).

Yang, X. and G. E. Karniadakis (2013). Reweighted ℓ_1 minimization method for stochastic elliptic differential equations. *J. Comput. Phys. 248*, 87–108.

Yang, X., W. Li, and A. Tartakovsky (2018). Sliced-inverse-regression–aided rotated compressive sensing method for uncertainty quantification. *SIAM/ASA J. Unc. Quant. 6*(4), 1532–1554.

Yin, P., Y. Lou, Q. He, and J. Xin (2015). Minimization of ℓ_{1-2} for compressed sensing. *SIAM J. Sci. Comput. 37*(1), A536–A563.

Zankin, V. P., G. V. Ryzhakov, and I. V. Oseledets (2018). Gradient descent-based D-optimal design for the least-squares polynomial approximation. *arXiv preprint arXiv:1806.06631*.

Zein, S., B. Colson, and F. Glineur (2013). An efficient sampling method for regression-based polynomial chaos expansion. *Commun. Comput. Phys. 13*(4), 1173–1188.

Zhang, Z., Y. Xu, J. Yang, X. Li, and D. Zhang (2015). A survey of sparse representation: algorithms and applications. *IEEE access 3*, 490–530.

Zhao, H., Z. Gao, F. Xu, Y. Zhang, and J. Huang (2019). An efficient adaptive forward–backward selection method for sparse polynomial chaos expansion. *Comput. Methods Appl. Mech. Engrg. 355*, 456–491.

Zhao, W. and L. Bu (2019). Global sensitivity analysis with a hierarchical sparse metamodeling method. *Mech. Syst. Signal Pr. 115*, 769–781.

Zhou, T., A. Narayan, and Z. Xu (2014). Multivariate discrete least-squares approximations with a new type of collocation grid. *SIAM J. Sci. Comput. 36*(5), A2401–A2422.

Zhou, Y., Z. Lu, and K. Cheng (2019). Sparse polynomial chaos expansions for global sensitivity analysis with partial least squares and distance correlation. *Struct. Multidiscip. O. 59*(1), 229–247.

Zhou, Y., Z. Lu, K. Cheng, and C. Ling (2019). An efficient and robust adaptive sampling method for polynomial chaos expansion in sparse Bayesian learning framework. *Comput. Methods Appl. Mech. Engrg. 352*, 654–674.

Zhou, Y., Z. Lu, K. Cheng, and Y. Shi (2019). An expanded sparse Bayesian learning method for polynomial chaos expansion. *Mech. Syst. Signal Pr. 128*, 153–171.

Zhou, Y., Z. Lu, J. Hu, and Y. Hu (2020). Surrogate modeling of high-dimensional problems via data-driven polynomial chaos expansions and sparse partial least square. *Comput. Methods Appl. Mech. Engrg. 364*, 112906.

Zhu, X. and B. Sudret (2020). Replication-based emulation of the response distribution of stochastic simulators using generalized lambda distributions. *Int. J. Uncertainty Quantification 10*(3), 249–275.

A Details on experimental design sampling techniques

It depends on certain properties of the regression matrix $\mathbf{\Psi}$ whether or not sparse regression techniques are able to find the true sparse solution of a linear system of equations (assuming that it exists). In the context of polynomial chaos, the entries of the regression matrix are the basis polynomials evaluated at the design points. The basis polynomials are determined by the distribution of the input random variables and the choice of the index set $\mathcal{A}$, while the design points $\{\boldsymbol{x}^{(j)}\}_{j=1}^{N}$ can be chosen freely from the input space to optimize properties of the resulting regression matrix.

In the following, we present sampling schemes that have been proposed in the literature for the computation of sparse PCE. Some of the schemes come with theoretical results about their performance for sparse PCE, others have heuristic justification or have guarantees for least-squares regression. They can be broadly grouped into three categories:

- Sampling according to the input distribution
  - MC (Doostan and Owhadi, 2011; Hampton and Doostan, 2015b)
  - LHS (McKay et al., 1979)
- Sampling from a modified distribution (induced sampling)
  - asymptotic (Hampton and Doostan, 2015b)
  - coherence-optimal (Hampton and Doostan, 2015b)
  - Christoffel sparse approximation (Jakeman et al., 2017)
- Optimizing matrix properties
  - D-optimal (Diaz et al., 2018)
  - S-optimal (Shin and Xiu, 2016a; Fajraoui et al., 2017)
  - near-optimal (Alemazkoor and Meidani, 2018a)

Some of the sampling schemes are nontrivial or costly to evaluate, or even not available for all input distributions. However, the bottleneck in surrogate modelling for practical applications is typically the repeated evaluation of the model, which justifies the use of a complex sampling scheme if it allows better approximation with fewer samples.

A.1 Sampling according to the input distribution

This class of sampling methods consists of all methods that are oblivious to the choice of truncation set $\mathcal{A}$ and whose main objective is to distribute design points evenly in the quantile space. Heuristically, the more uniformly the points are distributed in the quantile space, the more information about the model is captured in the model evaluations, since no region of the input domain is forgotten.

LHS is one technique for achieving a space-filling design. For each component of the input random vector $\boldsymbol{X}$, the corresponding quantile space is divided into N intervals. In each interval, one point is sampled uniformly at random. Then, the points for each dimension are combined randomly into vectors and finally transformed into the input space using an isoprobabilistic transform. LHS can be shown to reduce the variance of linear regression estimates when the main effects are dominant, i.e., when the most important terms have interaction order one (Shields and Zhang, 2016). LHS can be combined with heuristic criteria such as the maximin distance strategy, where among several random LHS designs the one with the largest minimal pairwise distance between points is chosen, to further improve on the space-filling property.

Stratified sampling is a related sampling technique in which the input space is divided into disjoint regions, called strata, from which points are sampled and weighted according to the probability mass of their stratum. Stratified sampling reduces the variance of statistical estimators (McKay et al., 1979). There exists a range of methods between stratified sampling and LHS, called partially stratified sampling, which are able to reduce the variance of statistical estimators when interaction terms are dominant (Shields and Zhang, 2016). The authors propose an additional method called Latinized partially stratified sampling (LPSS) which combines LHS and stratified sampling with the aim tif minimizing the variance of the resulting estimator. It is especially beneficial when there is prior knowledge about which variable groups interact, and it has been used for several problems with input dimension $d = 100$.

MC sampling, i.e., sampling from the input distribution, is a special case of the coherence-based theory detailed in Section A.2.1 below, and bounds on the coherence and the associated number of points needed for recovery can be derived (Doostan and Owhadi, 2011; Rauhut and Ward, 2012; Yan et al., 2012; Hampton and Doostan, 2015b).

A.2 Sampling from a different distribution

The ability of sparse regression to recover the true sparse solution (if it exists; otherwise it recovers the best sparse approximation) largely depends on the regression matrix. In the case of PCE, the entries of this matrix are the evaluations of the basis polynomials at the experimental design points. The points can be chosen in a way that improves the recovery properties of the matrix.

Several approaches exist in which the ℓ^1-minimization problem is modified into a weighted problem and samples are drawn not from the input distribution, but from a suitable modified distribution. The idea of these approaches is as follows. Define a weight function $w(x) : \Omega \to \mathbb{R}$ in a suitable way, which will be explained later. For an ED $\{\boldsymbol{x}^{(k)}\}_{k=1}^N$, define the diagonal matrix $\boldsymbol{W} = \mathrm{diag}(w(\boldsymbol{x}^{(1)}), \ldots, w(\boldsymbol{x}^{(N)}))$. Then the following modified system is solved:

$$\min_{\boldsymbol{c}} \|\boldsymbol{c}\|_1 \ \text{ s.t. } \ \|\boldsymbol{W}\boldsymbol{\Psi}\boldsymbol{c} - \boldsymbol{W}\boldsymbol{y}\|_2 \leq \epsilon. \tag{14}$$

Depending on $w(\boldsymbol{x})$, this modification can improve or deteriorate the solution $\boldsymbol{c}$. Of course, the weight function is chosen to improve the solution. The matrix $\boldsymbol{W}\boldsymbol{\Psi}$ can also be interpreted as

the evaluation of a modified basis $\{\tilde{\psi}_{\boldsymbol{\alpha}}(\boldsymbol{x}) = w(\boldsymbol{x})\psi_{\boldsymbol{\alpha}}(\boldsymbol{x})\}_{\boldsymbol{\alpha}\in\mathcal{A}}$. To ensure orthonormality of the columns of $\boldsymbol{W\Psi}$, the design points are drawn from a suitably modified input distribution $f_{\tilde{\boldsymbol{X}}}$.

A.2.1 Coherence, isotropy, and weighted orthonormal systems

In this section, we define concepts that are the basis for guarantees on accuracy and stability for different sampling distributions. We mainly follow the exposition in (Hampton and Doostan, 2015b).

In the setting of PCE, the *coherence* of an orthonormal system $\{\psi_{\boldsymbol{\alpha}}\}_{\boldsymbol{\alpha}\in\mathcal{A}}$ is defined by

$$\mu(\mathcal{A}, \{\psi_{\boldsymbol{\alpha}}\}) = \sup_{\boldsymbol{x}\in\mathcal{D}} \max_{\boldsymbol{\alpha}\in\mathcal{A}} |\psi_{\boldsymbol{\alpha}}(\boldsymbol{x})|^2. \tag{15}$$

For distributions for which this quantity would be ∞, such as a Gaussian distribution, see the remark below.

A second important concept is *isotropy* (Candès and Plan, 2011): a random matrix, whose rows are chosen randomly following some distribution $\boldsymbol{a} \sim F_{\boldsymbol{a}}$, is isotropic if it holds that $\mathbb{E}\left[\boldsymbol{a}^T\boldsymbol{a}\right] = \mathbb{1}$. In the case of PCE, $F_{\boldsymbol{a}}$ is induced by propagating the input distribution $F_{\boldsymbol{X}}$ through the basis functions. By construction, the regression matrix of standard PCE is isotropic if the ED is sampled from the input distribution. Under the assumption that the regression matrix $\boldsymbol{\Psi}$ is isotropic, the number of samples needed for perfect recovery of sparse solutions in the noiseless case is proportional to $\mu(\mathcal{A}, \{\psi_{\boldsymbol{\alpha}}\})s\log(P)$ with high probability (Candès and Plan, 2011), where s is the sparsity of the solution vector and $P = |\mathcal{A}|$ is the number of basis functions. A similar result holds in the noisy case.

Thus, an orthonormal system $\{\psi_{\boldsymbol{\alpha}}\}_{\boldsymbol{\alpha}\in\mathcal{A}}$ with low coherence $\mu(\mathcal{A}, \{\psi_{\boldsymbol{\alpha}}\})$ requires fewer samples for perfect recovery. The goal of coherence-optimal sampling is to find a weighted system $\{\tilde{\psi}_{\boldsymbol{\alpha}}(\boldsymbol{x}) = w(\boldsymbol{x})\psi_{\boldsymbol{\alpha}}(\boldsymbol{x})\}_{\boldsymbol{\alpha}\in\mathcal{A}}$ that achieves $\mu(\mathcal{A}, \{\tilde{\psi}_{\boldsymbol{\alpha}}\}) < \mu(\mathcal{A}, \{\psi_{\boldsymbol{\alpha}}\})$ and is orthonormal with respect to some distribution $\tilde{f}_{\boldsymbol{X}}$.

The ideas of isotropy and coherence were applied to PCE by Hampton and Doostan (2015b), who construct an isotropic regression matrix with improved coherence as follows. Let $B : \mathcal{D} \to \mathbb{R}$ be the tight upper bound for the polynomial basis,

$$B(\boldsymbol{x}) = \max_{\boldsymbol{\alpha}\in\mathcal{A}} |\psi_{\boldsymbol{\alpha}}(\boldsymbol{x})|. \tag{16}$$

Let $G : \mathcal{D} \to \mathbb{R}$ be a loose upper bound with $G(\boldsymbol{x}) \geq B(\boldsymbol{x})\ \forall \boldsymbol{x} \in \mathcal{D}$. ($G$ is useful because using a simple expression for the upper bound can in some cases result in $\tilde{f}_{\boldsymbol{X}}$ being a well-known distribution that can be sampled from easily.) Define a new probability distribution $\tilde{f}_{\boldsymbol{X}}(\boldsymbol{x})$ by

$$\tilde{f}_{\boldsymbol{X}}(\boldsymbol{x}) = c^2 G(\boldsymbol{x})^2 f_{\boldsymbol{X}}(\boldsymbol{x}), \tag{17}$$

where $c = \left(\int_\Omega f_{\boldsymbol{X}}(\boldsymbol{x})G(\boldsymbol{x})^2 \mathrm{d}\boldsymbol{x}\right)^{-\frac{1}{2}}$ is the normalizing constant. Then, with the weight function

$$w(\boldsymbol{x}) = \frac{1}{cG(\boldsymbol{x})}, \tag{18}$$

the set of functions $\{\tilde{\psi}_{\boldsymbol{\alpha}}(\boldsymbol{x}) = w(\boldsymbol{x})\psi_{\boldsymbol{\alpha}}(\boldsymbol{x})\}_{\boldsymbol{\alpha}\in\mathcal{A}}$ is an orthonormal system with respect to the distribution $\tilde{f}_{\boldsymbol{X}}$. This follows directly from the orthonormality of $\{\psi_{\boldsymbol{\alpha}}\}_{\boldsymbol{\alpha}\in\mathcal{A}}$ with respect to $f_{\boldsymbol{X}}$. Furthermore, if $G = B$, the coherence $\mu(\mathcal{A}, \{\tilde{\psi}_{\boldsymbol{\alpha}}\})$ is minimal.

Remark Some polynomial bases (e.g. Hermite polynomials) do not have a finite upper bound. It is still possible to obtain similar results by considering a smaller domain $\mathcal{S} \subset \mathcal{D}$ on which the upper bound is finite and the isotropy is still approximately fulfilled. The modified probability distribution is then $\tilde{f}_{\boldsymbol{X}}(\boldsymbol{x}) = c^2 G(\boldsymbol{x})^2 f_{\boldsymbol{X}}(\boldsymbol{x}) \mathbb{1}_{\mathcal{S}}(\boldsymbol{x})$.

A.2.2 Sampling using a loose upper bound ("asymptotic sampling")

In the case of Legendre and Hermite polynomials, and using a certain loose upper bound $G(\boldsymbol{x}) \geq \max_{\boldsymbol{\alpha} \in \mathcal{A}} |\psi_{\boldsymbol{\alpha}}(\boldsymbol{x})|$, analytical expressions for distributions with improved coherence can be obtained (Hampton and Doostan, 2015b).

In the case of Legendre polynomials on $[-1, 1]^d$, a loose upper bound on the polynomials is given by $G(\boldsymbol{x}) \propto \prod_{i=1}^d (1 - x_i^2)^{-\frac{1}{4}}$, which leads to the Chebyshev distribution $\tilde{f}_{\boldsymbol{X}}(\boldsymbol{x}) = \prod_{i=1}^d \frac{1}{\pi \sqrt{1-x_i^2}}$ and to the weight function $w(\boldsymbol{x}) = \prod_{i=1}^d (1 - x_i^2)^{\frac{1}{4}}$.

In the case of Hermite polynomials for standard Gaussian variables, a loose upper bound on the polynomials is given by $G(\boldsymbol{x}) \propto \exp(\frac{1}{4} \|\boldsymbol{x}\|_2^2)$, and the subset $\mathcal{S}$ is chosen to be the d-dimensional ball with radius $\sqrt{2}\sqrt{2p+1}$. This leads to a uniform distribution $\tilde{f}_{\boldsymbol{X}}$ on $\mathcal{S}$ and to the weight function $w(\boldsymbol{x}) = \exp(-\frac{1}{4} \|\boldsymbol{x}\|_2^2)$.

Additionally, asymptotic distributions for Laguerre polynomials (corresponding to the Gamma distribution) and for Jacobi polynomials (Beta distribution) have been implemented in the software package COH-OPT (Hampton and Doostan, 2017).

For Legendre polynomials, asymptotic sampling has a smaller coherence than standard sampling in the case $d < p$ (asymptotically). In the case $d > p$, which is more common in applications, standard sampling has (asymptotically) a smaller coherence. According to theory, the sampling scheme with smaller coherence should exhibit better recovery rates. This is confirmed numerically (Hampton and Doostan, 2015b, section 5.1). For Hermite polynomials, the same observation is made.

A.2.3 Coherence-optimal sampling

The choice $G = B$ leads to the minimum possible coherence $\mu(\mathcal{A}, \{\tilde{\psi}_{\boldsymbol{\alpha}}\})$ (Hampton and Doostan, 2015b, Theorem 4.5). B is simple to evaluate for a single point $\boldsymbol{x} \in \mathcal{D}$, but its functional form is in general not known. Therefore, Hampton and Doostan (2015b) suggest sampling $\tilde{f}_{\boldsymbol{X}} \propto B^2 f_{\boldsymbol{X}}$ using Markov chain Monte Carlo (MCMC) sampling with proposal distribution equal to the input distribution in the case $d \geq p$ and equal to the asymptotic distribution in the case $d < p$. The resulting (unnormalized) weights are $w(\boldsymbol{x}) = \frac{1}{B(\boldsymbol{x})}$. As expected from theory, numerical examples indicate that coherence-optimal sampling achieves better recovery and a smaller error in various norms than both standard and asymptotic sampling (Hampton and Doostan, 2015b). Coherence-optimal sampling can be shown to have good properties also when used as a sampling scheme for least-squares regression (Hampton and Doostan, 2015a).

MATLAB code for MCMC-based coherence-optimal sampling is available (Hampton and Doostan, 2015a,b). However, MCMC-based coherence-optimal sampling can be very slow for high-dimensional input. An alternative is rejection-based coherence-optimal sampling. Here, samples $\boldsymbol{x}_{\text{cand}}$ are generated from a proposal distribution f_{prop}, which has the property that there is a $\gamma \in \mathbb{R}$ such that $\gamma f_{\text{prop}}(\boldsymbol{x}) \geq \tilde{f}_{\boldsymbol{X}}(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{D}$. Uniform random numbers $u \sim_{\text{i.i.d.}} \mathcal{U}([0,1])$ are generated. A proposed point $\boldsymbol{x}_{\text{cand}}$ is accepted if $u \leq \frac{\tilde{f}(\boldsymbol{x}_{\text{cand}})}{\gamma f_{\text{prop}}(\boldsymbol{x}_{\text{cand}})}$. This is the implementation used in this benchmark. We use a product proposal density whose marginals are determined by the input marginals, the dimension of the problem, and the degree of the expansion: we choose uniform proposal marginals for uniform input marginals. For Gaussian input marginals, we use Gaussian proposal marginals if $d \geq p$; otherwise, we use the corresponding asymptotic distribution. As usual, lognormal input is mapped to Gaussian random variables before sampling (Blatman and Sudret, 2011).

Note that for Gaussian input, coherence-optimal and asymptotic sampling have a significantly larger spread than input sampling, as can be seen from Figure 8. Their support is the ball of radius $r = \sqrt{2}\sqrt{2p+2}$ (as implemented in (Hampton and Doostan, 2017)). This can potentially cause problems in engineering applications, for which simulations may be less accurate when the input parameters are far from typical operating conditions.

A.2.4 Christoffel sparse approximation

A similar weighted sampling scheme is Christoffel sparse approximation (Narayan et al., 2017; Jakeman et al., 2017; Cohen and Migliorati, 2017). Those authors propose to use the weight function

$$w(\boldsymbol{x}) = \left(\frac{1}{|\mathcal{A}|} \sum_{\boldsymbol{\alpha} \in \mathcal{A}} |\psi_{\boldsymbol{\alpha}}(\boldsymbol{x})|^2 \right)^{-\frac{1}{2}} \tag{19}$$

which leads to a modified basis that has pointwise minimal average squared basis magnitude (compare to (18) with $G = B$). This quantity (6) is a measure similar to coherence (5) and is used by Hampton and Doostan (2015a) and Cohen and Migliorati (2017) together with the induced probability measure to obtain convergence results for weighted least-squares regression. Narayan et al. (2017) and Jakeman et al. (2017) choose as probability distribution the so-called *weighted pluripotential equilibrium measure* (possibly degree-dependent), which asymptotically coincides with $\tilde{f}(\boldsymbol{x}) = c^2 w(\boldsymbol{x})^2 f(\boldsymbol{x})$ when the total degree of the truncated basis $p \to \infty$. However, the modified basis is not orthonormal with respect to this measure, which leads to weaker theoretical recovery results. Theoretical results are available only for the univariate case. In numerical examples, the method performs well for low-dimensional high-degree cases and often very similarly to asymptotic sampling. In high dimensions, it performs worse than input sampling (i.e., MC). It has not been compared to coherence-optimal sampling.

A.3 Choosing points according to an optimality criterion from a candidate set

The following methods aim to improve the properties of the regression matrix by choosing the "best" design points from a large candidate set. The methods differ in the criterion defining what are the "best" points. Most presented algorithms are greedy or heuristic and are actually only able to find a suboptimal design (local optimum). The choice of the candidate set obviously influences the quality of the resulting design. In the literature, candidate sets were sampled from MC (Diaz et al., 2018), LHS (Fajraoui et al., 2017), coherence-optimal sampling (Diaz et al., 2018; Alemazkoor and Meidani, 2018a), or Christoffel sparse approximation (Shin and Xiu, 2016b). In the case of coherence-optimal sampling or Christoffel sparse approximation, the resulting optimized sample inherits the weights. It also often preserves the spread of the candidate set, as can be seen in Figure 8.

A.3.1 D-optimal sampling

D-optimal design of experiments (Kiefer and Wolfowitz, 1959; Dykstra, 1971) aims at maximizing the determinant of the so-called *information matrix* $\frac{1}{N}\boldsymbol{\Psi}^T\boldsymbol{\Psi} \in \mathbb{R}^{P\times P}$. The D-value is defined as

$$D(\boldsymbol{\Psi}) = \det(\boldsymbol{\Psi}^T\boldsymbol{\Psi}). \tag{20}$$

Sometimes the determinant of the inverse information matrix is minimized (Nguyen and Miller, 1992), or the Pth root is taken for normalization purposes (Diaz et al., 2018). The maximization of this determinant is connected to the minimization of the variance of the PCE coefficient estimate (Nguyen and Miller, 1992; Zein et al., 2013). Note that $D(\boldsymbol{\Psi}) = 0$ if $N < P$.

There exists a large selection of methods for constructing D-optimal experimental designs. For an overview of methods for constructing designs following alphabetic optimality criteria (such as A-, D-, or E-optimality), see (Hadigol and Doostan, 2018, Section 4.5).

Here, we only discuss D-optimal sampling based on rank-revealing QR decomposition (RRQR) (Diaz et al., 2018), since this is the technique used in our benchmark. We decided to use RRQR-based D-optimal sampling because it can be used even in the case $N < P$ when other D-optimal methods fail due to singularity of the information matrix. Note that RRQR is not guaranteed to find a design with maximal D-value but only a local optimum (Diaz et al., 2018, Section 3.4).

Let $\boldsymbol{\Psi}_{\text{cand}} \in \mathbb{R}^{M\times P}$ be the regression matrix evaluated at a set of M candidate points. The goal is to select $N \leq M$ points from this candidate set with the property that the D-value of the resulting regression matrix $\boldsymbol{\Psi} \in \mathbb{R}^{N\times P}$ is as large as possible. Since in the case of sparse PCE often $N < P$, which leads to $D(\boldsymbol{\Psi}) = 0$, another strategy is necessary. The RRQR decomposition, also known as pivoted QR, aims at permuting the columns of the original matrix in a way that ensures the R-matrix of the associated QR decomposition is as well-behaved[12]

[12] $\boldsymbol{R} = \begin{pmatrix} A_k & B_k \\ 0 & C_k \end{pmatrix}$ where A_k is well-conditioned and $\|C_k\|_2$ is small

as possible. This is useful for inexpensively determining the numerical rank of a matrix (Hong and Pan, 1992; Gu and Eisenstat, 1996). RRQR has a strong connection to SVD and to the selection of submatrices of maximal determinant (Hong and Pan, 1992). Gu and Eisenstat (1996) propose a pivoted QR decomposition where pivots are chosen to maximize the determinant of the resulting quadratic submatrix of R. The exchange of rows is based on a formula relating the determinant of the quadratic submatrix of R before and after the row exchange by a simple factor (Gu and Eisenstat, 1996, Lemma 3.1). This algorithm can be used together with SVD to perform subset selection to construct an initial experimental design from a large set of candidate samples (Seshadri et al., 2017; Diaz et al., 2018). Here, first an SVD of the matrix $\boldsymbol{\Psi}_{\text{cand}}^T$ is computed. Then RRQR is applied to the transpose of the matrix consisting of the first N right singular vectors. The resulting permutation matrix is used to determine the points to be chosen from the candidate set.

A.3.2 Quasi-optimal sampling based on the S-value

Here, the idea is to select samples from a pool of candidate points so that the PCE coefficients obtained using the selected set are as close as possible to the coefficients that would be obtained if the whole set of candidate points was used (Shin and Xiu, 2016a). Under the assumption that the columns of the matrix $\boldsymbol{\Psi}_{\text{cand}}$ are mutually orthogonal, the S-value is defined by

$$S(\boldsymbol{\Psi}) = \left(\frac{\sqrt{\det \boldsymbol{\Psi}^T \boldsymbol{\Psi}}}{\prod_{i=1}^{P} \|\Psi_i\|_2} \right)^{\frac{1}{P}}, \tag{21}$$

where Ψ_i denotes the ith column of the regression matrix $\boldsymbol{\Psi}$. Its maximization has the (heuristic) effect of maximizing the column orthogonality of the regression matrix while at the same time maximizing the determinant of the information matrix (Shin and Xiu, 2016a). It holds that $S(\boldsymbol{\Psi}) \in [0, 1]$ due to Hadamard's inequality. If $N < P$, $S(\boldsymbol{\Psi}) = 0$. If $N \geq P$, $S(\boldsymbol{\Psi}) = 1$ if and only if the columns of $\boldsymbol{\Psi}$ are mutually orthogonal. There exists an update formula for the S-value when the regression matrix is augmented by one row, which thus avoids the repeated calculation of determinants.

Shin and Xiu (2016a) suggest a greedy algorithm that in every iteration augments the current matrix by an additional row which maximizes the S-value of the resulting matrix among all candidate rows. When the current number of rows in the matrix $\boldsymbol{\Psi}$ is smaller than the number of columns, the procedure can be adapted to avoid $S(\boldsymbol{\Psi}) = 0$. We do not include it in our benchmark because it is not well suited for situations where there are more basis polynomials than design points, which is the case in sparse PCE without experimental design enrichment. However, in a sequential enrichment context (Fajraoui et al., 2017) and for least-squares regression (Shin and Xiu, 2016b), this algorithm performs well.

A.3.3 Near-optimal sampling

The coherence parameter (15) gives a bound on the recovery rate, but it is not the only criterion that has been studied with respect to recovery accuracy. Two other matrix properties related to

recovery accuracy are mutual coherence and average cross-correlation. Both of them consider the correlation between normalized columns of the regression matrix, i.e., their scalar product. They are scalar measures of how "orthonormal" the columns of a rectangular matrix $\mathbf{\Psi} \in \mathbb{R}^{N \times P}$ with $N < P$ are. The heuristic idea is that columns should point in as different directions as possible, so that the multiplication with sparse coefficient vectors, which results in a linear combination of a subset of the columns, is "as unique as possible". This facilitates the recovery of the true sparse solution (assuming that it exists).

The *mutual coherence* is defined by

$$\mu(\mathbf{\Psi}) = \max_{i \neq j} \frac{|\Psi_i^T \Psi_j|}{\|\Psi_i\|_2 \|\Psi_j\|_2}, \tag{22}$$

where Ψ_i denotes the ith column of the regression matrix $\mathbf{\Psi} \in \mathbb{R}^{N \times P}$. The mutual coherence is the worst-case cross-correlation between any two columns of the matrix. It is zero for orthonormal matrices and positive for $N < P$.

The *average (squared) cross-correlation* is defined by

$$\gamma(\mathbf{\Psi}) = \frac{1}{P(P-1)} \left\| \mathbb{1}_P - \tilde{\mathbf{\Psi}}^T \tilde{\mathbf{\Psi}} \right\|_F^2 = \frac{1}{P(P-1)} \sum_{i \neq j} \frac{|\Psi_i^T \Psi_j|^2}{\|\Psi_i\|_2^2 \|\Psi_j\|_2^2} \tag{23}$$

where $\tilde{\mathbf{\Psi}}$ is the column-normalized version of $\mathbf{\Psi}$, and Ψ_i denotes the ith column of the regression matrix. The norm is the Frobenius-norm, taking the sum of squares of all matrix entries, and the factor $P(P-1)$ is the number of column pairs.

Alemazkoor and Meidani (2018a) suggest simultaneously optimizing mutual coherence and average cross-correlation by using the greedy procedure described in Algorithm 1 below: In each iteration, the current regression matrix is augmented by one row. This row corresponds to that point $\boldsymbol{x}_j$ from the large pool of candidate points which minimizes the (normalized) distance of $(\mu'_j, \gamma'_j) \in \mathbb{R}^2$ to the "utopia point" $(\min(\boldsymbol{\mu}'), \min(\boldsymbol{\gamma}'))$ among all candidate points.

Algorithm 1 Near-optimal sampling (Alemazkoor and Meidani, 2018a).

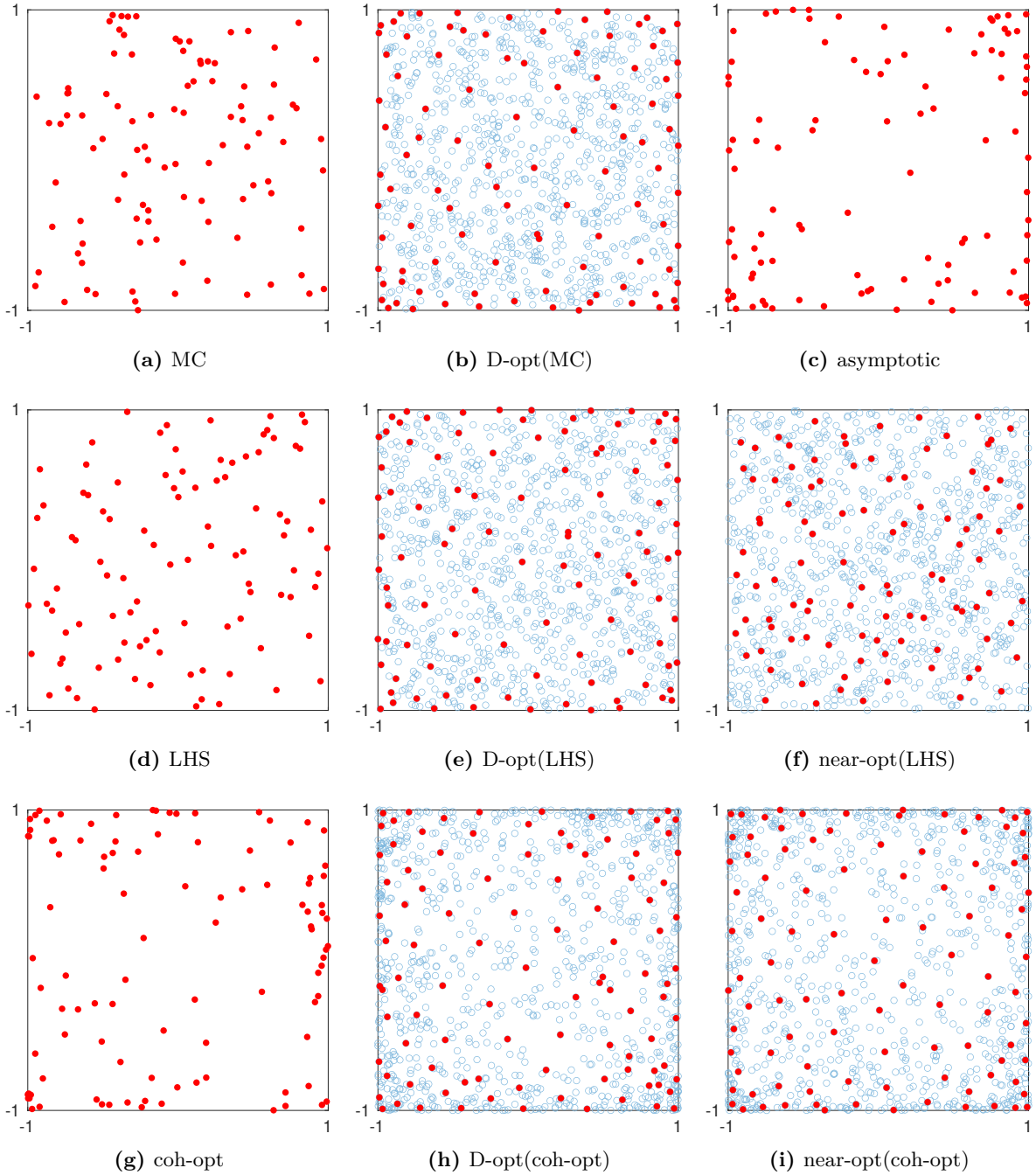
1: Sample a large number M of candidate points from the coherence-optimal distribution and compute candidate rows arranged in a matrix $\mathbf{\Psi}_{\text{cand}}$
2: Initialize $\mathbf{\Psi}_{\text{opt(1)}}$ to be a random row from $\mathbf{\Psi}_{\text{cand}}$
3: **for** $i = 2 \dots N$ **do**
4: **for** $j = 1 \dots M$ **do**
5: $\mathbf{\Psi}_{\text{temp}} = \text{row-concatenate}(\mathbf{\Psi}_{\text{opt(i-1)}}, \Psi_{\text{cand}}^{(j)})$
6: $\mu'_j = \mu(\mathbf{\Psi}_{\text{temp}})$ and $\gamma'_j = \gamma(\mathbf{\Psi}_{\text{temp}})$
7: **end for**
8: $\boldsymbol{\mu}' = (\mu'_1, \dots, \mu'_M)$ and $\boldsymbol{\gamma}' = (\gamma'_1, \dots, \gamma'_M)$
9: $j^* = \arg\min_j \left(\frac{\mu'_j - \min(\boldsymbol{\mu}')}{\max(\boldsymbol{\mu}') - \min(\boldsymbol{\mu}')} \right)^2 + \left(\frac{\gamma'_j - \min(\boldsymbol{\gamma}')}{\max(\boldsymbol{\gamma}') - \min(\boldsymbol{\gamma}')} \right)^2$
10: $\mathbf{\Psi}_{\text{opt(i)}} = \text{row-concatenate}(\mathbf{\Psi}_{\text{opt(i-1)}}, \Psi_{\text{cand}}^{(j^*)})$
11: **end for**

The algorithm is called near-optimal because it is a greedy algorithm, finding only a local optimum, and because optimized mutual coherence and average cross-correlation are only hinting at, but not guaranteeing, good recovery accuracy (Alemazkoor and Meidani, 2018a). Its computational complexity is $\mathcal{O}(NMP^2)$, where N is the size of the final experimental design, M is the number of candidate samples (chosen to be, e.g., proportional to P (Diaz et al., 2018)), and P is the number of regressors. This makes the algorithm prohibitively expensive in the case of large bases (P in the order of thousands), which is why we do not use it for some of the benchmark examples.

A.4 Illustration of sampling schemes

In Figures 7 and 8, we show illustrations of experimental designs in $d = 2$ dimensions with $N = 100$ and $p = 12$ for selected sampling techniques. The candidate set has a size of $M = 1000$. Figure 7 presents experimental designs for uniform input in the interval $[-1, 1]$, while Figure 8 presents experimental designs for standard Gaussian input.

Note that in the standard Gaussian case, the asymptotic distribution, the coherence-optimal distribution, and the matrix-optimal distributions based on a coherence-optimal candidate set all have a very large spread that grows with the total degree of the basis. For degree $p = 12$, some points are seven standard deviations away from the mean. Engineering models are typically calibrated only for a certain region of the input domain corresponding to nonnegligible probability, and they may be less accurate (or even fail) outside of this region.

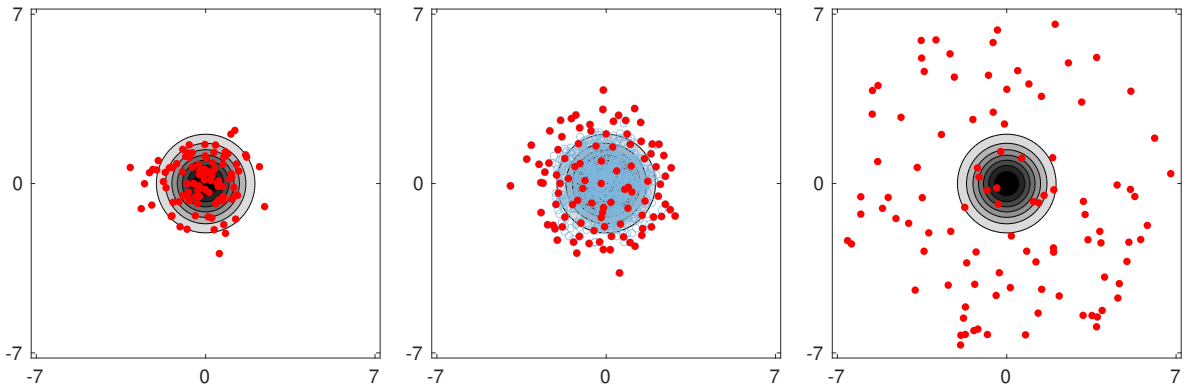


(a) MC **(b)** D-opt(MC) **(c)** asymptotic

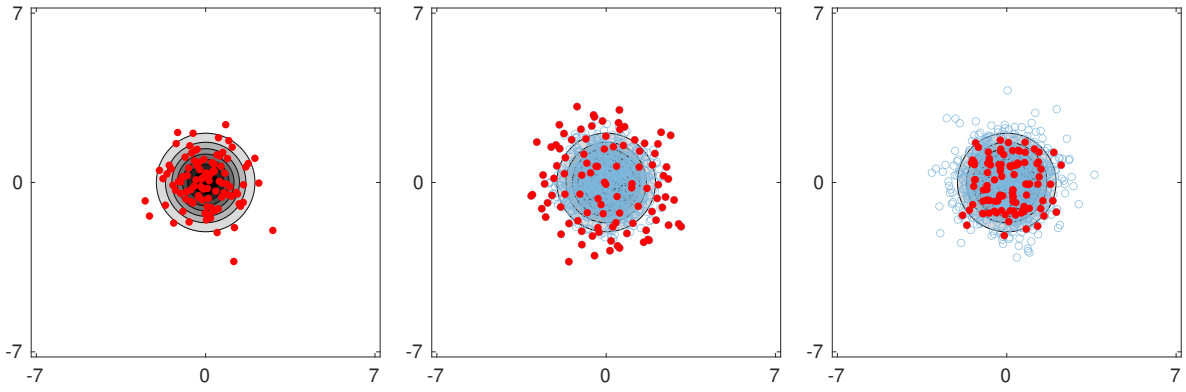
(d) LHS **(e)** D-opt(LHS) **(f)** near-opt(LHS)

(g) coh-opt **(h)** D-opt(coh-opt) **(i)** near-opt(coh-opt)

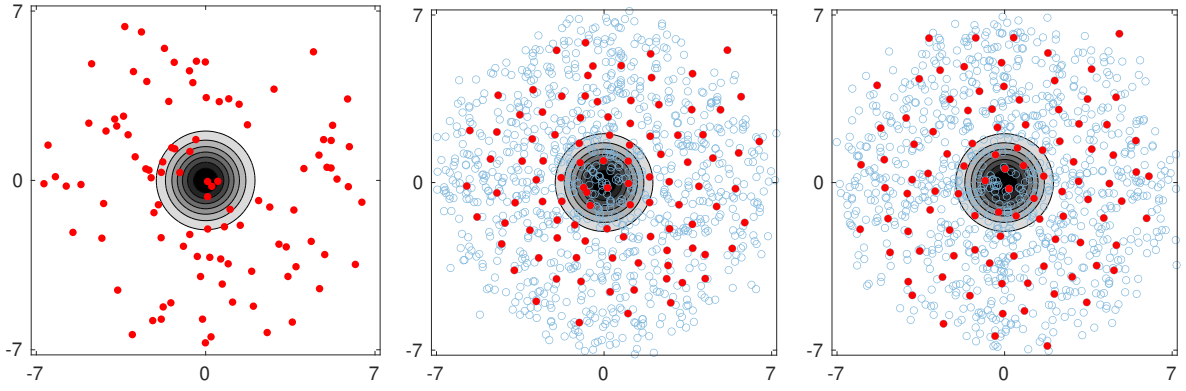
Figure 7: Visualization of experimental designs constructed for uniform input in $[-1, 1]^2$ for degree $p = 12$. Red filled points denote the chosen experimental design, while blue circles denote the candidate set. Size of the ED: $N = 100$, size of the candidate set: $M = 1000$.



(a) MC **(b)** D-opt(MC) **(c)** asymptotic



(d) LHS **(e)** D-opt(LHS) **(f)** near-opt(LHS)



(g) coh-opt **(h)** D-opt(coh-opt) **(i)** near-opt(coh-opt)

Figure 8: Visualization of the experimental design constructed for standard Gaussian input in $d = 2$ dimensions for degree $p = 12$. The gray surface plot illustrates the Gaussian probability density function. Red filled points denote the chosen experimental design, while blue circles denote the candidate set. Size of the ED: $N = 100$; size of the candidate set: $M = 1000$. The support of the asymptotic and the coherence-optimal distribution is the ball of radius $r = \sqrt{2}\sqrt{2p+2} \approx 7.2$. Note that engineering models may be less accurate in regions where the input distribution has negligible mass.

B Details on sparse regression solvers

In this appendix, we describe the sparse solvers used in our benchmark in more detail: LARS, OMP, subspace pursuit, SPGL1 and FastLaplace (BCS). In addition, we present an overview of greedy stepwise regression solvers for sparse PCE.

There exist various formulations for the sparse regression problem. The typical form minimizes the ℓ^2-norm of the empirical error under an additional constraint that is designed to enforce sparsity.

Sparsity is measured by the number of nonzero entries in a vector, formally denoted by $\|\boldsymbol{c}\|_0 = \sum_i \mathbf{1}_{\{c_i \neq 0\}}$ (even though this expression is not a norm). This results in the sparse regression problem

$$\hat{\boldsymbol{c}} = \arg\min_{\boldsymbol{c}} \|\boldsymbol{\Psi}\boldsymbol{c} - \boldsymbol{y}\|_2^2 + \lambda \|\boldsymbol{c}\|_0 \tag{24}$$

called ℓ^0-minimization. The only way to solve this problem exactly is by a combinatorial search through all possible nonzero patterns for $\boldsymbol{c}$, which is infeasible for large problem sizes.

The convex relaxation of this problem is ℓ^1-minimization, where $\|\boldsymbol{c}\|_0$ is replaced by $\|\boldsymbol{c}\|_1 = \sum_i |c_i|$. There are several equivalent formulations of the relaxed problem, namely

$$\hat{\boldsymbol{c}} = \arg\min_{\boldsymbol{c}} \|\boldsymbol{\Psi}\boldsymbol{c} - \boldsymbol{y}\|_2^2 + \lambda \|\boldsymbol{c}\|_1 \tag{25}$$

$$\hat{\boldsymbol{c}} = \arg\min_{\boldsymbol{c}} \|\boldsymbol{c}\|_1 \text{ s.t. } \|\boldsymbol{\Psi}\boldsymbol{c} - \boldsymbol{y}\|_2 \leq \sigma \tag{26}$$

$$\hat{\boldsymbol{c}} = \arg\min_{\boldsymbol{c}} \|\boldsymbol{\Psi}\boldsymbol{c} - \boldsymbol{y}\|_2 \text{ s.t. } \|\boldsymbol{c}\|_1 \leq \tau \tag{27}$$

called Lagrangian formulation, basis pursuit denoising (BPDN), and least absolute shrinkage and selection operator (LASSO), respectively. It has been shown that under certain conditions, the solutions to ℓ^0-minimization and ℓ^1-minimization coincide (Bruckstein et al., 2009). However contrary to (24), formulations (25)–(27) are convex problems and allow a numerical solution with considerably smaller cost. The three formulations (25), (26), and (27) are equivalent in the sense that if $\hat{\boldsymbol{c}}$ is solution to one of the formulations, there exists a value of constraint parameter σ, τ, or λ so that $\hat{\boldsymbol{c}}$ is also a solution to the other formulations. However, the relationship between the parameters σ, τ, λ that makes the problems equivalent depends on $\boldsymbol{\Psi}$ and $\boldsymbol{y}$ and is not known in advance (van den Berg and Friedlander, 2008).

There exist other sparsity-enforcing formulations, such as ℓ^p-norms (Bruckstein et al., 2009), $\ell^1 - \ell^2$-minimization (Yin et al., 2015), or elastic net (Tarakanov and Elsheikh, 2019). One example that we will describe is Bayesian compressive sensing, where a sparsity-enforcing prior is used for the coefficients of the PCE, resulting in a formulation that is equivalent to a sparse regression problem with a different kind of sparsity constraint, e.g., one related to the Student-t distribution (Tipping, 2001).

In the following descriptions of the algorithms, $\mathcal{A} \subset \mathbb{N}^d$ with various sub- or superscripts denotes a set of multi-indices, which by definition of PCE can be identified with a set of basis polynomials. With the notation of Section 2.1, $\boldsymbol{y} \in \mathbb{R}^N$ denotes the vector of model responses, $\boldsymbol{\Psi} \in \mathbb{R}^{N \times P}$ denotes the regression matrix of basis polynomials evaluated at the N experimental design

points, and $\boldsymbol{c} \in \mathbb{R}^P$ denotes the coefficients of a PCE. The residual is defined by $\boldsymbol{r} = \boldsymbol{y} - \boldsymbol{\Psi c}$, so that the norm of the residual is the empirical error.

The polynomials used for building the PCE are sometimes also called basis functions, regressors, or predictors. A sparse PCE is a PCE for which only some of the basis functions have nonzero coefficients: these basis functions are called *active*. We assume the regressors are normalized, so that the correlation between two regressors is equivalent to their inner product (and to the cosine of the angle between them).

B.1 Orthogonal matching pursuit (OMP)

Orthogonal matching pursuit (OMP), also called forward stepwise regression, is a classical greedy technique for finding approximate solutions to the ℓ^0-minimization problem (24) (Pati et al., 1993; Bruckstein et al., 2009). Despite its being a heuristic method, under certain assumptions there are theoretical guarantees for the solutions returned by OMP (Tropp and Gilbert, 2007; Bruckstein et al., 2009). OMP is an iterative algorithm that starts out with an empty model and adds the regressors one by one to the set of active regressors. In each iteration, OMP selects the regressor that is most correlated with the current residual, adds it to the set of active regressors, and then updates the coefficients of all active regressors to make sure the new residual is orthogonal to all of them and has smallest possible norm. The updating of the coefficients can be done through an update formula (Berchier, 2015) or by computing the least-squares solution to the system of equations involving only the active regressors (Marelli and Sudret, 2019).

The technique is presented in Algorithm 2. The iterations are continued until $\min\{N, P\}$ basis functions are in the active set (then either all polynomials are selected, or there are not enough points in the experimental design to use least-squares anymore).

OMP does not, per se, return a sparse solution. If a desired level of sparsity K is known a priori, the algorithm can be stopped after K iterations. Another possibility is to stop the algorithm as soon as the residual norm is smaller than some error threshold (Bruckstein et al., 2009; Doostan and Owhadi, 2011; Jakeman et al., 2015), where the best error threshold is determined through cross-validation. A third possibility is to determine the best number of active basis functions through a model selection criterion, e.g. the LOO error (Marelli and Sudret, 2019). Since the coefficients are computed by OLS on the active basis, the LOO can be computed cheaply (Chapelle et al., 2002; Blatman and Sudret, 2011). Typically, for an increasing sequence of basis functions the LOO error first decreases (reduction of underfitting), then increases (overfitting). This can be utilized to terminate the algorithm early once the LOO error starts rising (early-stop criterion) (Marelli and Sudret, 2019).

The computational complexity of OMP is $\mathcal{O}(mNP)$ (Tropp and Gilbert, 2007; Dai and Milenkovic, 2009), where $m \leq \min\{N, P\}$ is the number of iterations. The computation of the correlations of the current residual with all regressors is $\mathcal{O}(NP)$ and has to be performed m times. The computation of the least-squares solution in step i can be done in $\mathcal{O}(iN)$, e.g., by maintaining a QR factorization of the information matrix (Tropp and Gilbert, 2007), or by using Schur's complement to update the information matrix inverse whenever a new regressor is added.

Algorithm 2 Orthogonal matching pursuit (OMP) (Pati et al., 1993; Tropp and Gilbert, 2007; Marelli and Sudret, 2019).

Given a set of candidate basis functions $\mathcal{A}_{\text{cand}}$
Initialize all coefficients to zero: $\boldsymbol{c}_0 = 0$. Set $\mathcal{A}_0 = \emptyset$
Set the residual vector $\boldsymbol{r} := \boldsymbol{y}$
for $i = 1, \dots, m$ **do** ▷ $m \leq \min\{N, P\}$
▷ OMP can be stopped early when the error did not decrease anymore for a while
Find $\boldsymbol{\alpha}^* \in \mathcal{A}_{\text{cand}} \setminus \mathcal{A}_{i-1}$ with maximal correlation with the residual by solving

$$\boldsymbol{\alpha}^* = \arg \max_{\boldsymbol{\alpha} \in \mathcal{A}_{\text{cand}} \setminus \mathcal{A}_{i-1}} |\boldsymbol{r}^T \boldsymbol{\psi}_{\boldsymbol{\alpha}}|$$

▷ The entries of vector $\boldsymbol{\psi}_{\boldsymbol{\alpha}}$ are evaluations of the basis function $\psi_{\boldsymbol{\alpha}}$ at the ED
$\mathcal{A}_i = \mathcal{A}_{i-1} \cup \{\boldsymbol{\alpha}\}$ ▷ Current set of active predictors
Compute the coefficients $\boldsymbol{c}_i$ by least-squares using only the active indices $\mathcal{A}_i$
▷ This can be done in $\mathcal{O}(iN)$ when maintaining a QR factorization (Tropp and Gilbert, 2007)
Update the residual $\boldsymbol{r} = \boldsymbol{y} - \boldsymbol{\Psi}_{\mathcal{A}_i} \boldsymbol{c}_i$
end for

From the authors' experience, OMP often suffers from overfitting and can produce an unreliable LOO error estimate, which can be detrimental in basis-adaptive settings (see also (Lüthen et al., 2021)).

OMP is available in many software packages, among them UQLab (Marelli and Sudret, 2014).

B.2 Least angle regression (LARS)

Least-angle regression (LARS, sometimes also abbreviated LAR) is a greedy technique that finds an approximate solution to the ℓ^1-minimization problem (Efron et al., 2004). It is similar to OMP in that the algorithm starts out with an empty model and adds regressors one by one based on their correlation with the residual. However, unlike OMP, which updates the coefficients using least-squares (making the residual orthogonal to all active regressors in each step), LARS updates the coefficients in such a way that all active regressors have equal correlation with the residual. LARS can be interpreted as producing a path of solutions to (27), corresponding to increasing τ. The coefficients are increased in the equiangular direction until a nonactive regressor has as much correlation with the residual as all the active regressors. This regressor is then added to the set of active regressors and the new equiangular direction is computed. The optimal stepsize between the addition of subsequent regressors can be computed analytically (Efron et al., 2004). This algorithm solves (27) approximately. A slightly modified version of LARS, called LARS-LASSO, removes regressors whenever the sign of their coefficient changes, and it has been proven to solve (27) (or its noiseless counterpart) exactly under certain conditions (Efron et al., 2004; Bruckstein et al., 2009).

Algorithm 3 Least angle regression (LARS) (Efron et al., 2004; Blatman and Sudret, 2011).

Given a set of candidate basis functions $\mathcal{A}_{\text{cand}}$
Initialize all coefficients to zero: $\boldsymbol{c}_0 = 0$. Set $\mathcal{A}_0 = \emptyset$
Set the residual vector $\boldsymbol{r} := \boldsymbol{y}$
for $i = 1, \ldots, m$ **do** ▷ $m = \min\{N, P\}$
    Find $\boldsymbol{\alpha} \in \mathcal{A}_{\text{cand}} \setminus \mathcal{A}_{i-1}$ with maximal correlation with the residual
        ▷ For $i > 1$, $\boldsymbol{\alpha}_i$ is the element "responsible for" γ_i
    $\mathcal{A}_i = \mathcal{A}_{i-1} \cup \{\boldsymbol{\alpha}\}$ ▷ Current set of active predictors
    Compute $\boldsymbol{c}_i$ ▷ Equiangular direction for all $\boldsymbol{\alpha} \in \mathcal{A}_i$
        ▷ $\boldsymbol{c}_1$ is equal to the first selected predictor
        ▷ (Efron et al., 2004, Eq. 2.6)
    Compute γ_i ▷ Optimal stepsize: using this, there is a new regressor that is as much correlated with r as all regressors in $\mathcal{A}_i$ are
        ▷ (Efron et al., 2004, Eq. 2.13)
    Compute the new coefficients $\boldsymbol{c}_i = \boldsymbol{c}_{i-1} + \gamma_i \boldsymbol{c}_i$ ▷ Move the coefficients jointly into the direction of the least-squares solution until one of the other predictors in $\mathcal{A}_{\text{cand}} \setminus \mathcal{A}_i$ has as much correlation with the residual as the predictors in $\mathcal{A}_i$ (ensured by choice of γ_i and $\boldsymbol{c}_i$)
    Update the residual $\boldsymbol{r} = \boldsymbol{y} - \boldsymbol{\Psi}_i \boldsymbol{c}_i$
end for

The LARS technique is presented in Algorithm 3. It returns a sequence $\mathcal{A}_1 \subset \mathcal{A}_2 \subset \ldots \subset \mathcal{A}_m$ of sets containing indices of active basis functions, with $m = \min\{N, P\}$. Just like OMP, LARS can be stopped when a predefined sparsity K is reached or when the norm of the residual $\|\boldsymbol{r}\|_2$ falls below a predefined error threshold.

A modified version of LARS, called *ybrid LARS*, uses the equicorrelated approach to select the predictors, but computes the coefficients of the metamodel by least-squares (Efron et al., 2004; Blatman and Sudret, 2011). Once the LARS algorithm has finished and returned the sequence of basis sets $\mathcal{A}_1, \ldots, \mathcal{A}_m$, the corresponding coefficients are recomputed by least squares, $\boldsymbol{c}_i = \boldsymbol{c}_i^{\text{LSQ}}$, which ensures minimal empirical error for every metamodel $(\mathcal{A}_i, \boldsymbol{c}_i)$. Hybrid LARS facilitates another way to choose the best sparsity level: as for OMP, a model selection criterion (e.g. LOO) for each metamodel can be evaluated, and the best one is chosen. This procedure is detailed in Algorithm 4. Cheap OLS-based computation of LOO (Chapelle et al., 2002; Blatman and Sudret, 2011) and the early-stop criterion (Marelli and Sudret, 2019) can be applied as well.

As for OMP, the computational complexity of LARS (in the case $N < P$) is $\mathcal{O}(mNP)$, where $m \leq \min\{N, P\}$ is the number of iterations. This is due to matrix-vector multiplication and matrix inversion which have to be performed in every iteration. The latter can be computed in $\mathcal{O}(mN)$, when using techniques such as Schur's complement to update the information matrix inverse whenever a new regressor is added.

LARS is available in many software packages, e.g., as MATLAB implementation in UQLab (Marelli and Sudret, 2014).

Algorithm 4 Hybrid LARS with LOO-CV (Blatman and Sudret, 2011; Marelli and Sudret, 2019)

1: Initialization as in LARS (Algorithm 3)
2: **for** $i = 1, \ldots, m$ **do**
3: Run one step of LARS and obtain $(\mathcal{A}_i, \boldsymbol{c}_i)$ ▷ Algorithm 3
4: Recompute the coefficient vector using least-squares on the selected basis $\mathcal{A}_i$ only, obtaining $\boldsymbol{c}_i^{\mathrm{OLS}}$ (the coefficients corresponding to $\mathcal{A} \setminus \mathcal{A}_i$ are set to zero) ▷ Hybrid LARS
5: Compute the LOO error $\epsilon_{\mathrm{LOO}}(i)$ for $\boldsymbol{c}_i^{\mathrm{OLS}}$ ▷ OLS-based LOO computation (Chapelle et al., 2002; Blatman and Sudret, 2011)
6: **end for** ▷ early stopping possible by monitoring the LOO error (Marelli and Sudret, 2019)
7: Return the metamodel $(\mathcal{A}_{i^*}, \boldsymbol{c}_{i^*})$ with $i^* = \arg\min_i \epsilon_{\mathrm{LOO}}(i)$

B.3 Subspace pursuit (SP)

Another formulation of the ℓ^0-minimization problem is

$$\min_{\boldsymbol{c} \in \mathbb{R}^P} \|\boldsymbol{\Psi}\boldsymbol{c} - \boldsymbol{u}\|_2 \text{ s.t. } \|\boldsymbol{c}\|_0 = K \tag{28}$$

which is equivalent to (24) for a certain choice of λ.

Subspace pursuit (SP) seeks to identify a solution to (28) by iteratively and greedily enlarging and shrinking the set of active basis functions (Dai and Milenkovic, 2009). As with LARS and OMP, regressors are added to the set of active basis functions according to their correlation with the residual. However, the regressors are not added one by one, but batchwise. More precisely, SP maintains at all times an active basis of size K, where K denotes the desired sparsity. In each iteration, it adds K regressors at once and computes the coefficients of the active regressors by OLS. Then, it removes the K regressors with the smallest-in-magnitude coefficients. This is continued until convergence. Under certain assumptions, there are theoretical guarantees for the solution that SP returns (Dai and Milenkovic, 2009). To make the augmentation of the basis and the OLS regression feasible, it must hold that $2K \leq \min\{N, P\}$.

The technique is described in Algorithm 5 for a fixed value of sparsity K. The residual of a vector and a regression matrix is defined as

$$\mathrm{residual}(\boldsymbol{y}, \boldsymbol{\Psi}) = \boldsymbol{y} - \boldsymbol{\Psi}\boldsymbol{\Psi}^{\dagger}\boldsymbol{y} \tag{29}$$

where $\boldsymbol{\Psi}^{\dagger}$ denotes the pseudoinverse of $\boldsymbol{\Psi}$ and $\boldsymbol{\Psi}^{\dagger}\boldsymbol{y} = \boldsymbol{c}$ is the least-squares solution to $\boldsymbol{\Psi}\boldsymbol{c} \approx \boldsymbol{y}$ (the case of an overdetermined system). The algorithm returns a set $\mathcal{A}$ containing K multi-indices.

For arbitrary sparse vectors, the computational complexity is $\mathcal{O}(N(P+K^2)K)$ (Dai and Milenkovic, 2009). For very sparse vectors with $K^2 \in \mathcal{O}(P)$, the complexity thus becomes $\mathcal{O}(NPK)$, comparable to the runtime of OMP. The number of iterations that the SP algorithm performs can be shown to be $\mathcal{O}(K)$ in general and even $\mathcal{O}(\log K)$ in certain cases (Dai and Milenkovic, 2009).

Algorithm 5 Subspace pursuit (SP) (Dai and Milenkovic, 2009).

1: Given desired sparsity $K \leq \min\{\frac{N}{2}, \frac{P}{2}\}$
2: Given the experimental design and the candidate basis $\mathcal{A}_{\text{cand}}$, compute the associated regression matrix $\boldsymbol{\Psi}$ and the right-hand-side $\boldsymbol{y}$
3: $\mathcal{A}^0 = \{K \text{ indices corresponding to the largest magnitude entries in } \boldsymbol{\Psi}^T\boldsymbol{y}\}$
▷ Scalar product of columns of $\boldsymbol{\Psi}$ with $\boldsymbol{y}$
4: $\boldsymbol{y}_{\text{res}}^0 = \text{residual}(\boldsymbol{y}, \boldsymbol{\Psi}_{\mathcal{A}^0})$ ▷ Residual of least-squares solution based on active basis
5: **for** $l = 1, 2, \ldots$ **do**
6: $\mathcal{S}^l = \mathcal{A}^{l-1} \cup \{K \text{ indices corresponding to the largest magnitude entries in } \boldsymbol{\Psi}^T\boldsymbol{y}_{\text{res}}^{l-1}\}$
▷ Augment by indices of full basis that correlate best with the residual
7: $\boldsymbol{c} = \boldsymbol{\Psi}_{\mathcal{S}^l}^{\dagger}\boldsymbol{y}$ ▷ Least-squares solution based on set $\mathcal{S}^l$ of size $2K$
8: $\mathcal{A}^l = \{K \text{ indices corresponding to the largest magnitude entries in } \boldsymbol{c}\}$
9: $\boldsymbol{y}_{\text{res}}^l = \text{residual}(\boldsymbol{y}, \boldsymbol{\Psi}_{\mathcal{A}^l})$ ▷ Residual of least-squares solution based on $\mathcal{A}^l$
10: **if** $\left\|\boldsymbol{y}_{\text{res}}^l\right\|_2 \geq \left\|\boldsymbol{y}_{\text{res}}^{l-1}\right\|_2$ **then** ▷ if new K-sparse approx. is worse than the previous one
11: STOP iteration and **return** $\mathcal{A}^{l-1}$.
12: **end if**
13: **end for**

Remark: In line 10, the original publication (Dai and Milenkovic, 2009) uses ">" instead of "≥", but we also want to stop when the set has converged.

When the optimal sparsity level K is unknown, it can be determined e.g. by cross-validation: Diaz et al. (2018) suggest running Algorithm 5 for a range of $N_K = 10$ different values for K and choosing the one with the smallest 4-fold cross-validation error. In this paper, we propose to use leave-one-out cross-validation instead of 4-fold cross-validation, resulting in the SP variant SP_{LOO}.

A related algorithm is CoSAMP (Needell and Tropp, 2009), which differs from SP mainly in the number of regressors added in each iteration.

Subspace pursuit is available as MATLAB implementation in the software package `DOPT_PCE` (Diaz et al., 2018; Diaz, 2018).

B.4 SPGL1

ℓ^1-minimization is a convex problem, since both the objective function and the constraint are convex functions. Therefore, convex optimization methods can be used to find a solution. In this section, we describe the algorithm SPGL1 (van den Berg and Friedlander, 2008).

For a given value of τ, formulation (27) (LASSO) can be solved by spectral projected gradient (SPG) descent (Birgin et al., 2000; van den Berg and Friedlander, 2008).[13] However, for real-world problems, we often do not know a priori an appropriate value for τ. On the other hand,

[13] SPG is a gradient-based optimization algorithm with several enhancements (Barzilai–Borwein spectral step length and the Grippo–Lampariello–Lucidi scheme of nonmonotone line search) and projection onto the feasible set $\Omega_\tau = \{\boldsymbol{c} \in \mathbb{R}^P : \|\boldsymbol{c}\|_1 \leq \tau\}$

a sensible range of values for σ in formulation (26) (BPDN) can typically be estimated based on the noise level in the data and the expected model fit. In the case of PCE metamodelling, σ can be related to an estimate of the relative MSE through $\text{RelMSE} = \frac{\sigma^2}{N\hat{\text{Var}}[\boldsymbol{y}]}$, whose values are for engineering models typically between 10^{-10} and $10^0 = 1$.

The main idea of the solver SPGL1 is to solve BPDN through a detour over LASSO. Let $\boldsymbol{c}_\tau$ be the solution to LASSO for a given τ. Define a function $\phi : \mathbb{R}_+ \to \mathbb{R}_+$ by

$$\phi(\tau) := \|\boldsymbol{\Psi}\boldsymbol{c}_\tau - \boldsymbol{y}\|_2 \,. \tag{30}$$

Then the solution to BPDN with $\sigma := \phi(\tau)$ is $\boldsymbol{c}_\tau$. In other words, ϕ is the functional relationship between σ and τ that makes the two formulations BPDN and LASSO equivalent for given $\boldsymbol{\Psi}$ and $\boldsymbol{y}$. ϕ is the Pareto front of LASSO and BPDN and shows the trade-off between the minimal achievable ℓ^1-norm of the coefficients and the minimal ℓ^2-norm of the corresponding residual. The Pareto front is convex, nonincreasing and differentiable with an analytically computable derivative (van den Berg and Friedlander, 2008).

To find a solution to BPDN with a given σ, LASSO is solved with SPG several times for a sequence of τ until one is found with $\phi(\tau) = \sigma$. The sequence of τ is created by performing Newton's root finding algorithm on the function $f(\tau) = \sigma - \phi(\tau)$.

Each SPG iteration has a computational complexity of $\mathcal{O}(NP + P \log P)$ (from matrix-vector multiplication and ℓ^1-projection). Multiplying this with the number of SPG steps and the number of Newton steps yields the computational complexity of SPGL1.

This algorithm is available as MATLAB package `SPGL1` (van den Berg and Friedlander, 2008; Van den Berg and Friedlander, 2015).

In our numerical benchmarks computing sparse PCE for compressible models, SPGL1 was among the slowest solvers and often returned rather dense solutions.

B.5 Sparse Bayesian learning

Methods from the class of Bayesian compressive sensing (BCS), also known as sparse Bayesian learning (SBL), embed the regression problem in a probabilistic framework (Tipping, 2001; Ji et al., 2008; Babacan et al., 2010; Sargsyan et al., 2014; Tsilifis et al., 2020). The goal is to compute, for a given model response vector $\boldsymbol{y}$ and a regression matrix $\boldsymbol{\Psi}$, the coefficient vector $\boldsymbol{c}^{\text{MAP}}$ which maximizes the posterior distribution $p(\boldsymbol{c}|\boldsymbol{y})$. Another quantity of interest could be the most probable value y^* at a new point $\boldsymbol{x}^*$ maximizing $p(y^*|\boldsymbol{y})$.

In BCS, it is assumed that the "measurements" $\boldsymbol{y}$ are generated by adding zero-mean, finite-variance noise to the evaluations of the true model. This noise is often assumed to be Gaussian white noise with standard deviation σ, which, for a given input $\boldsymbol{x}$, results in a Gaussian distribution for its output $\boldsymbol{y}$ with mean $\boldsymbol{\Psi}\boldsymbol{c}$ and covariance matrix $\sigma^2\mathbb{1}$, i.e., $\boldsymbol{y}|\boldsymbol{c}, \boldsymbol{x}, \sigma \sim \mathcal{N}(\boldsymbol{\Psi}\boldsymbol{c}, \sigma^2\mathbb{1})$. Note that in the case of PCE, this is generally not a valid assumption: when an important term is missing from the PCE model, the discrepancy between measurements and PCE model evaluations can be highly correlated, heteroscedastic, and non-Gaussian, and have nonzero mean.

However, even though the assumptions might not be fulfilled, this framework can still be useful for finding sparse solutions.

The class of BCS algorithms comprises several methods that differ in the assumptions on the distributions of the various hyperparameters and in the (usually iterative, approximate) techniques for computing the posterior quantities.

In Figure 9a we present the general setup of sparse Bayesian learning. The measurements $\boldsymbol{y}$ are assumed to follow a Gaussian distribution as described above. The noise variance σ^2 is assumed to be a random variable whose distribution has to be specified (e.g. fixed, uniform, or inverse-Gamma). The coefficients c_i are assumed to be random variables as well, drawn from a normal distribution with mean zero and variance γ_i, i.e., each weight has its own variance. $\boldsymbol{\gamma}$ is a so-called *hyperparameter*, parametrizing the distribution of a parameter.

(a) General setup of BCS

(b) Setup by (Babacan et al., 2010)

Figure 9: Illustration of the general setup of BCS (a) and the hierarchical generalization of (Babacan et al., 2010) (b). (a): The likelihood and the prior on the coefficients are usually Gaussian, but the choice of $p(\sigma^2)$ and $p(\boldsymbol{\gamma})$ differs between publications, as well as the resulting solution algorithm. (b): Babacan et al. (2010) makes a specific choice for $p(\boldsymbol{\gamma})$ and includes an additional layer of hyperparameters. Shaded variables are held fixed.

So far, the described setup with fixed σ^2 and γ_i would yield (weighted) ridge regression. The sparsity comes into play through an assumption on the distribution of the hyperparameter $\boldsymbol{\gamma}$. For specific choices of $p(\gamma_i)$, it can be shown that the resulting *effective prior* on the coefficients $p(\boldsymbol{c}) = \int p(\boldsymbol{c}|\boldsymbol{\gamma})p(\boldsymbol{\gamma})\mathrm{d}\boldsymbol{\gamma}$ is a sparsity-encouraging distribution, i.e., one that has a sharp peak at zero, encouraging zero values, while at the same time a heavy tail, allowing for large coefficient values as well. Examples are the Laplace distribution and the generalized Student-t distribution (Wipf et al., 2004; Babacan et al., 2010; Figueiredo and Nowak, 2001).

Such sparsity-encouraging distributions are often intractable to use, because they do not allow for analytical computation of the desired values (such as the most likely coefficients given the data, or the prediction of the measurement value at a new point). However, feasible algorithms can be developed based on a suitable approximation step. Various frameworks for sparse Bayesian learning have been proposed whose setup follows the general structure of Figure 9a, but which

differ in the choice of priors for the hyperparameters and employ different solution algorithms for the MAP estimate of these hyperparameters (Tipping, 2001; Faul and Tipping, 2002; Figueiredo, 2003; Tipping and Faul, 2003; Wipf and Rao, 2004; Ji et al., 2008; Seeger and Nickisch, 2008; Babacan et al., 2010; Sargsyan et al., 2014; Tsilifis et al., 2020; Bhattacharyya, 2020). The MAP estimate of the hyperparameters is inserted into the distribution for $\boldsymbol{c}$. Because $p(\boldsymbol{c}|\boldsymbol{\gamma})$ and $p(\boldsymbol{y}|\boldsymbol{c}, \sigma^2)$ are normal distributions, all subsequent computations can be carried out analytically (Tipping, 2001; Wipf et al., 2004; Wipf and Rao, 2004). The sparsity of $\boldsymbol{c}$ is enforced because by the choice of $p(\boldsymbol{\gamma})$ and the other distributions, many of the components γ_i^{MAP} of $\boldsymbol{\gamma}^{\text{MAP}}$ will actually be zero, forcing the corresponding c_i to be zero as well.

BCS in the implementation of (Babacan et al., 2010) was suggested for sparse PCE by (Sargsyan et al., 2014). This approach employs an additional layer of hyperparameters as displayed in Figure 9b. The prior on the coefficient variances is an exponential distribution $p(\gamma_i|\lambda) = \text{Exp}\left(\gamma_i|\frac{\lambda}{2}\right)$ with shared hyperparameter λ. The hyperparameter λ follows a Gamma distribution $p(\lambda|\nu) = \Gamma(\lambda|\frac{\nu}{2}, \frac{\nu}{2})$ with hyperparameter ν. $\nu \to 0$ implies $p(\lambda) \propto \frac{1}{|\lambda|}$ (improper prior) and $\nu \to \infty$ implies the certain value $\lambda = 1$. In practice, (Babacan et al., 2010) find that $\nu = 0$ gives the best results. The prior on $\beta = \sigma^{-2}$ is a Gamma distribution $p(\beta) = \Gamma(\beta|a, b)$ with hyperparameters a, b. In practice, the algorithm does not estimate β well, which is, however, crucial; therefore, it is set to a fixed value (e.g. $\beta^{-1} = 0.01\,\|\boldsymbol{y}\|_2^2$ in (Babacan et al., 2010); in our benchmark, we use cross-validation to determine the best value for this parameter, similarly to the strategy for SPGL1). The objective function is the logarithm of the joint distribution $\mathcal{L}(\boldsymbol{\gamma}, \lambda, \beta) = \log p(\boldsymbol{y}, \boldsymbol{\gamma}, \lambda, \beta)$, which is an analytical expression. To maximize it, Babacan et al. (2010) adapt the fast approximate algorithm of Tipping and Faul (2003); Faul and Tipping (2002) to their generalized hierarchical setting. Here, the derivatives of the objective function with respect to the hyperparameters λ, β, and $\gamma_i, i = 1, \ldots, P$ are computed. This results in an iterative scheme where these parameters are optimized one at a time while the other ones are held fixed. The algorithm is explained in detail in Babacan et al. (2010, Algorithm 1) and has been implemented in MATLAB under the name FastLaplace (Babacan, 2011).

B.6 Greedy stepwise regression solvers

Many of the sparse regression solvers that have been proposed for computing sparse PCE belong to the class of greedy stepwise regression. Here, starting from an empty model, the regressors are added one by one according to a selection criterion (forward selection). Some methods also include a backward elimination step. Then the coefficients of the selected regressors are computed. The procedure is iterated until a stopping criterion is reached. Alternatively, several models are built and one is selected in the end using a model selection criterion. We summarize some greedy stepwise regression techniques proposed for sparse PCE, together with their choices for selection criterion, coefficient computation, and stopping criterion, in Table 3, including the well-known methods OMP and LARS.

New greedy methods in the fashion of Table 3 can easily be derived by pairing other methods for the regressor selection, the coefficient computation method, and the model selection crite-

rion. Note that except for LARS and OMP, these greedy methods are heuristic (no theoretical guarantee of convergence) and often depend on a number of tuning parameters.

Table 3: A selection of greedy stepwise regression algorithms proposed for sparse PCE. F = forward selection, FB = forward selection and backward elimination.

Ref.	**F/B**	**Regressor selection**	**Computation of the coefficients**	**Model selection / stopping criterion**	**Comment**
OMP (Tropp and Gilbert, 2007)	F	correlation with residual	OLS	*several choices:* given number of regressors (Tropp and Gilbert, 2007); (modified) LOO (Marelli and Sudret, 2019); norm of residual (Baptista et al., 2019); threshold on moving average of LOO (Baptista et al., 2019)	theoretical guarantees exist (Tropp and Gilbert, 2007)
LARS (Efron et al., 2004; Blatman and Sudret, 2011)	F	correlation with residual	least angle strategy	*several choices:* coefficient of determination (Efron et al., 2004); (modified) LOO (Blatman and Sudret, 2011)	theoretical guarantees exist (Efron et al., 2004; Bruckstein et al., 2009)
(Blatman and Sudret, 2008)	FB	coefficient of determination	OLS	coefficient of determination	degree-adaptive
(Blatman and Sudret, 2010a)	FB	coefficient of determination	OLS	LOO	degree-adaptive; with ED enrichment
(Hu and Youn, 2011)	F	testing bivariate interaction	stepwise moving least-squares	coefficient of determination	Only for interaction order ≤ 2. ED-adaptive. Degree of interaction terms determined in inner loop.
(Abraham et al., 2017)	FB	F: one-predictor regression criterion B: confidence intervals of coefficients	OLS	given number of regressors or iterations	-
(Shao et al., 2017)	F	partial correlation coefficient	variant of BCS (fixed priors)	KIC	prior depends on interaction order and degree of the regressor; degree-adaptive
(Cheng and Lu, 2018a)	FB	variance contribution of coefficient	support vector regression (SVR)	coefficient of determination	-
AFBS (Zhao et al., 2019)	FB	correlation with residual	OLS	threshold for residual norm	Main difference with OMP: backward selection. Employs several elimination checks.
HSPLSR-PCE (Zhao and Bu, 2019)	FB	F: partial least-squares (PLS) (regressors partitioned into blocks) B: soft thresholding	linear PLS on the regression matrix	modified (pseudo) LOO	Note that the input is assumed to be Gaussian, but the (nonlinear) regressors do not follow a Gaussian distribution.
(Zhou et al., 2019)	F	(partial) distance correlation	PLS	coefficient of determination	adaptive in degree and interaction order
(Zhou et al., 2019)	FB	correlation with residual	BCS	KIC	nesting iterative BCS procedure with forward-backward-selection scheme

C Additional results

In this appendix, we display additional results that complement the results shown in Sections 3.4–3.6. For a detailed description of the setup, we refer the reader to Section 3.

C.1 Comparison of sparse solvers

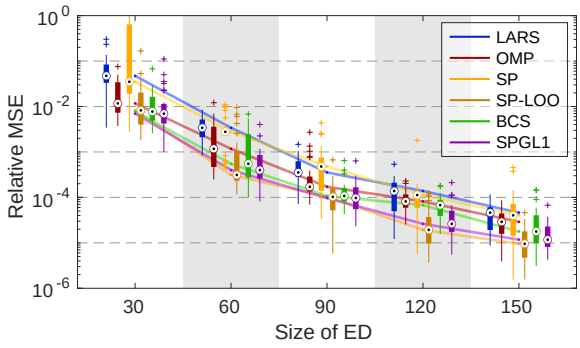
In Figure 10 we display the boxplots of relative MSE for the seven additional models presented in Table 1.

C.2 Comparison of sampling schemes together with solvers

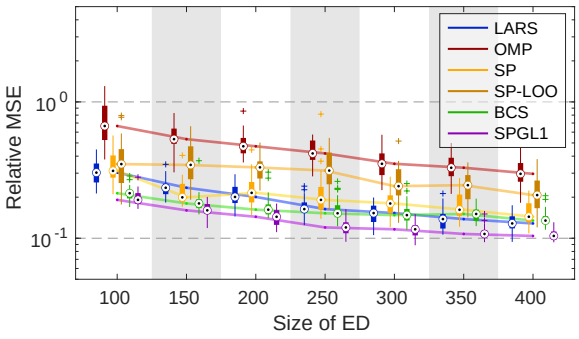
In Section 3.5, Figure 4 we showed aggregated results for the benchmark of solvers and sampling schemes. To give a more tangible impression of the data, in Figures 11–14 we display the boxplots of relative MSE against ED size for the four models Ishigami, borehole, two-dimensional diffusion, and 100D function. We show all combinations of solvers and sampling schemes, resulting in 16–20 combinations. Solvers are denoted by different colors. Sampling schemes are shown in varying shades and line styles. We also show the same results sliced at small and large ED sizes to compare the performance between solvers.

C.3 Comparison of sampling schemes together with solvers, using a smaller candidate basis

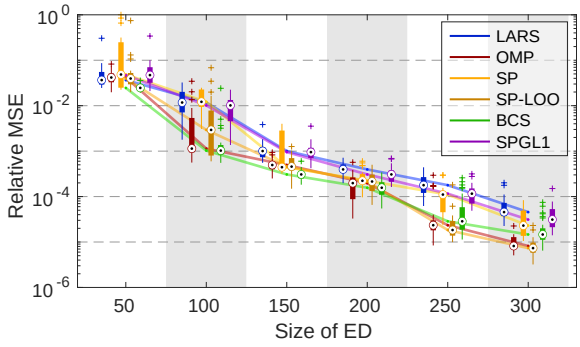
Due to space limitations, in Section 3.6 (Figures 5 and 6) we only showed results for two of the five solvers (OMP and $\mathrm{SP}_{\mathrm{LOO}}$). In Figure 15, we show boxplots of relative MSE against ED size for the three remaining solvers LARS, SP, and BCS.



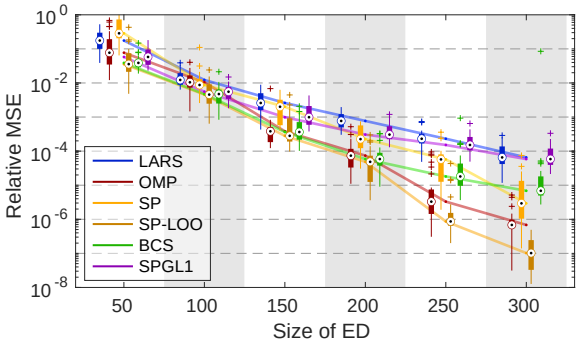
(a) Undamped oscillator ($d = 6$)



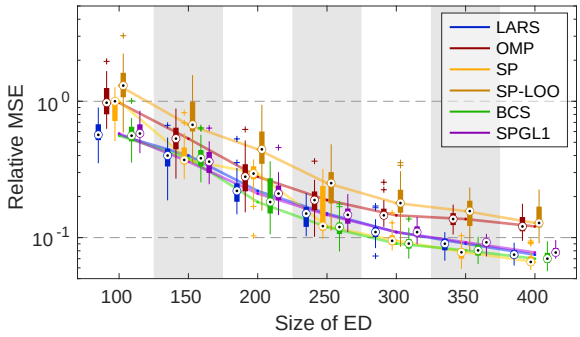
(b) Damped oscillator ($d = 8$)



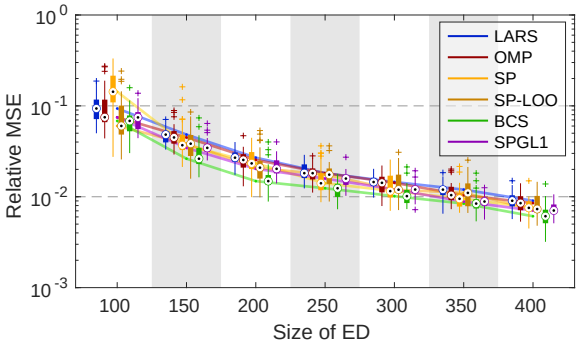
(c) Wingweight function ($d = 10$)



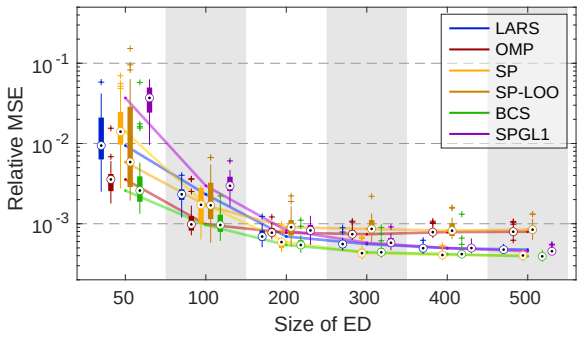
(d) Truss model ($d = 10$)



(e) Morris function ($d = 20$)

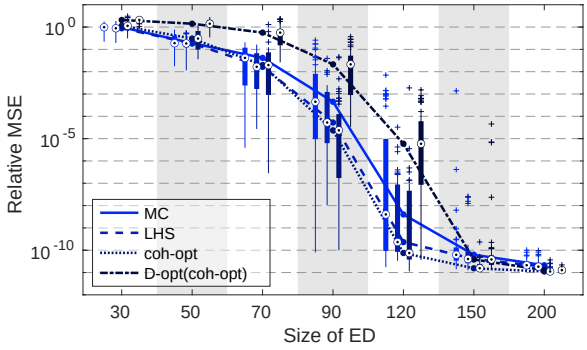


(f) Structural frame model ($d = 21$)

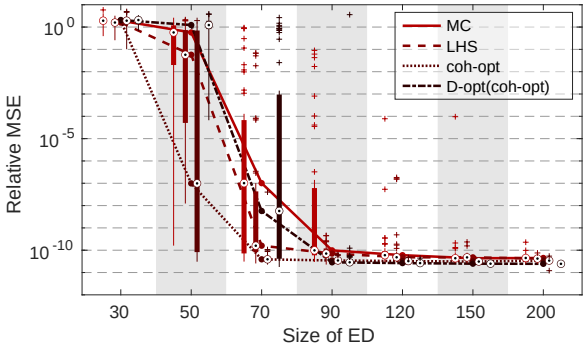


(g) 1-dim diffusion model ($d = 62$)

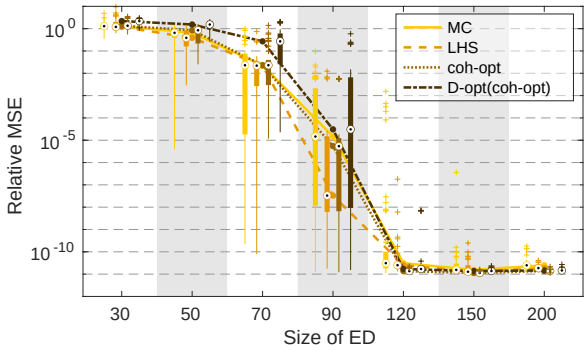
Figure 10: Results for seven additional models (see Table 1 in Section 3.4 for more details), complementing the results in Figure 2. Boxplots of relative MSE against experimental design for six sparse solvers and LHS design. Thirty replications. Note that the damped oscillator and the Morris function are very challenging for PCE: no solver achieves a relative MSE significantly smaller than 0.1, even when large EDs are used.



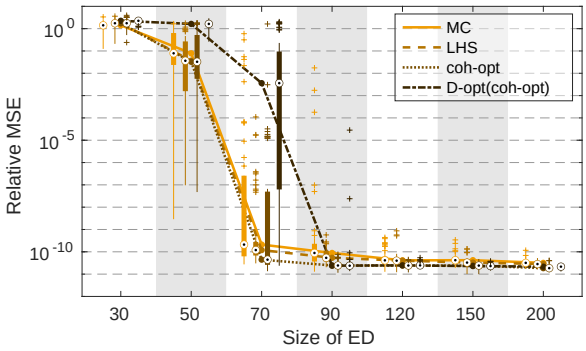
(a) LARS



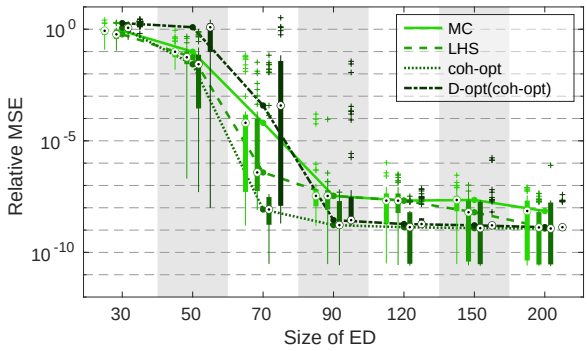
(b) OMP



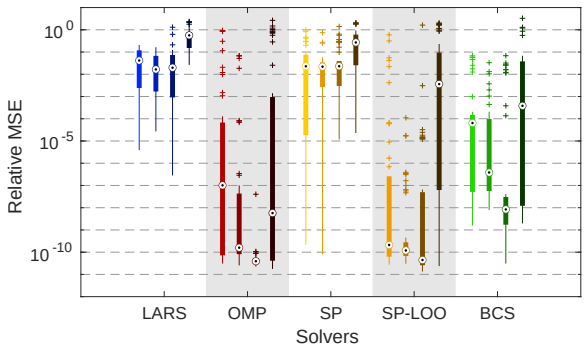
(c) SP



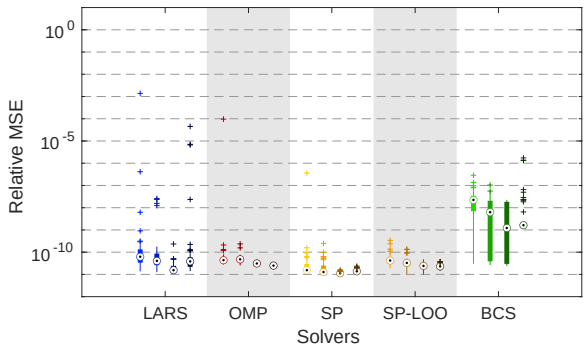
(d) SP_{LOO}



(e) BCS

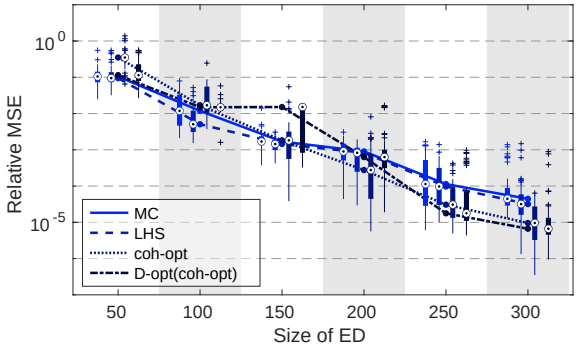


(f) Small ED (70 points)

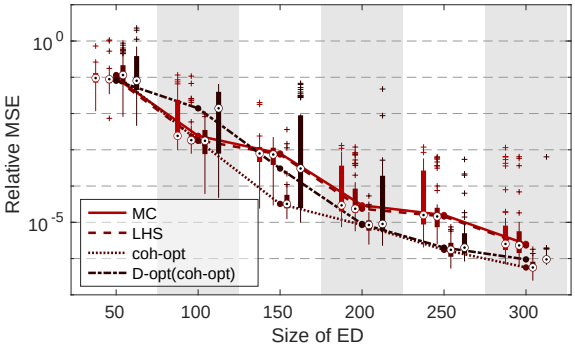


(g) Large ED (150 points)

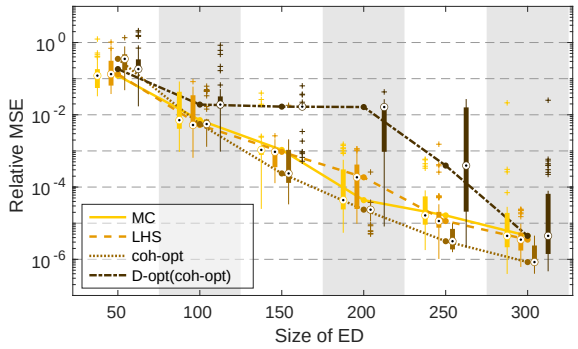
Figure 11: Boxplots of relative MSE from the benchmark of five solvers and four sampling schemes for the Ishigami model ($d = 3, p = 14, q = 1$). Solvers are coded by colors. Sampling schemes are shown in varying shades and line styles. In (f) and (g), we show the relative MSE of each of the solvers combined with each sampling scheme in the order MC–LHS–coh-opt–D-opt(coh-opt).



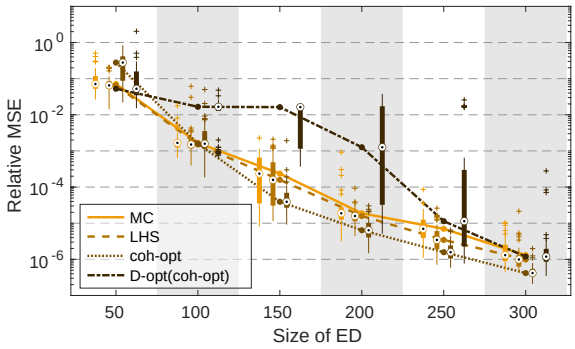
(a) LARS



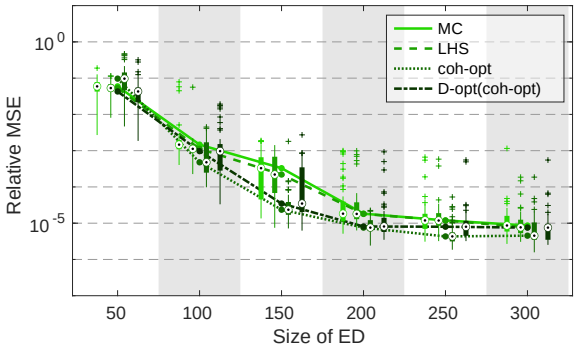
(b) OMP



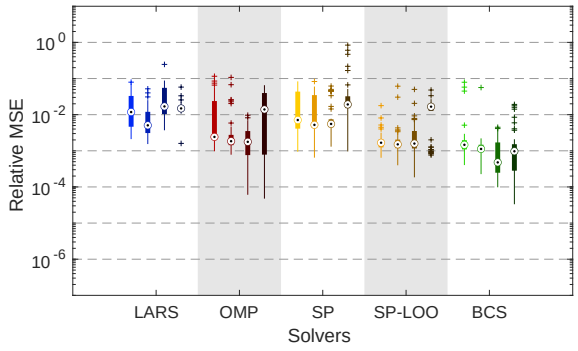
(c) SP



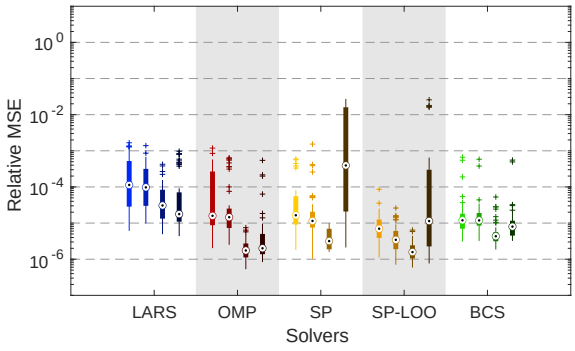
(d) SP_{LOO}



(e) BCS

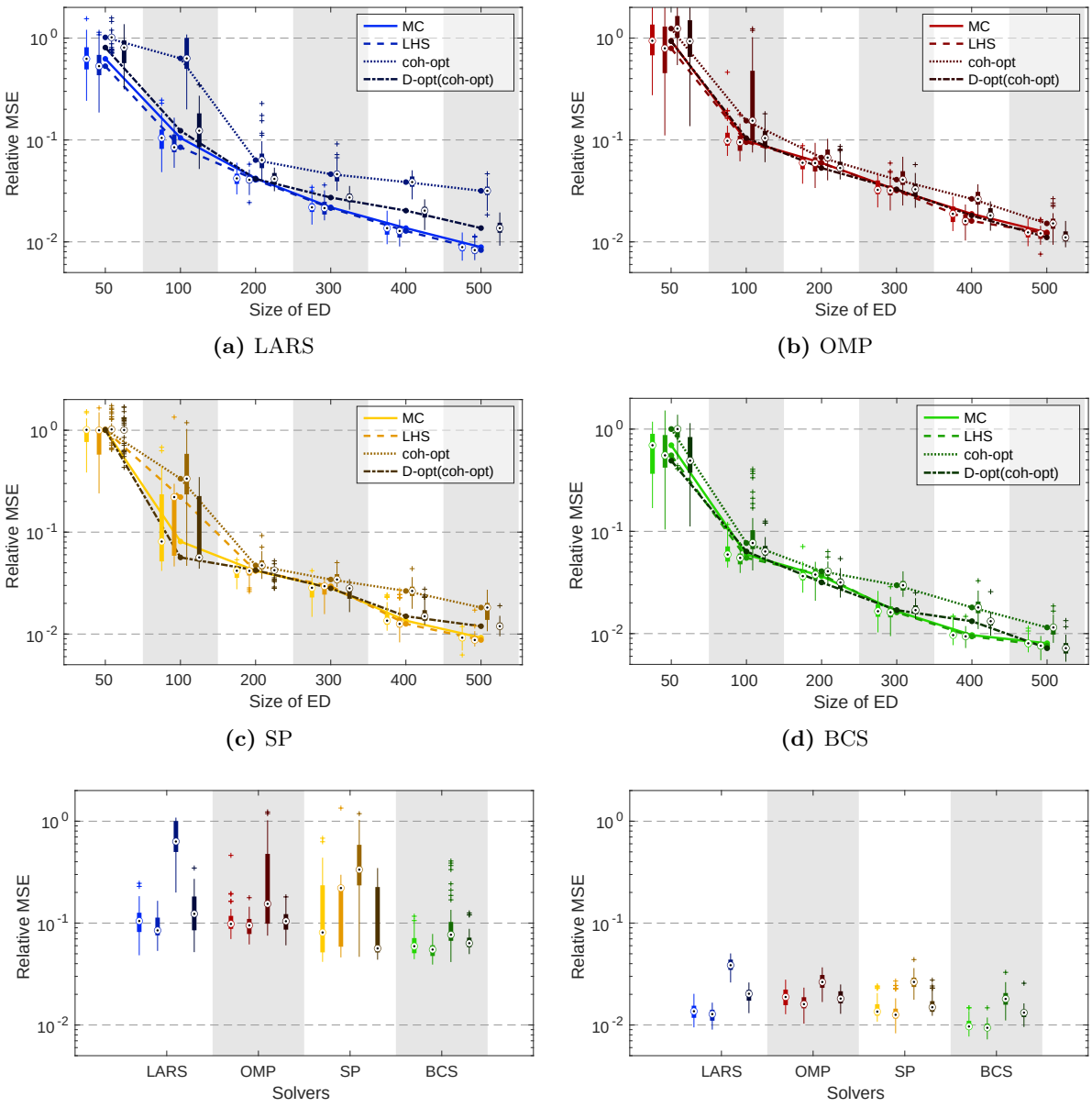


(f) Small ED (100 points)



(g) Large ED (250 points)

Figure 12: Boxplots of relative MSE from the benchmark of five solvers and four sampling schemes for the borehole model ($d = 8, p = 4, q = 1$). Solvers are coded by colors. Sampling schemes are shown in varying shades and line styles. In (f) and (g), we show the relative MSE of each of the solvers combined with each sampling scheme in the order MC–LHS–coh-opt–D-opt(coh-opt).

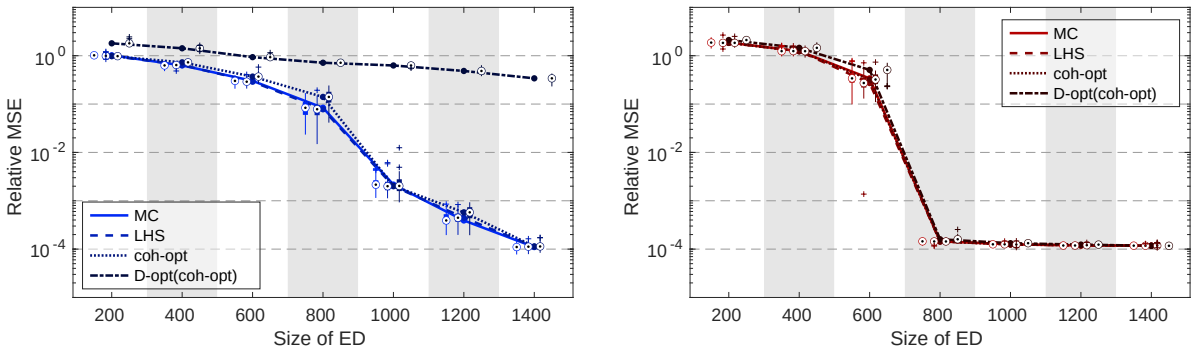


(a) LARS **(b)** OMP

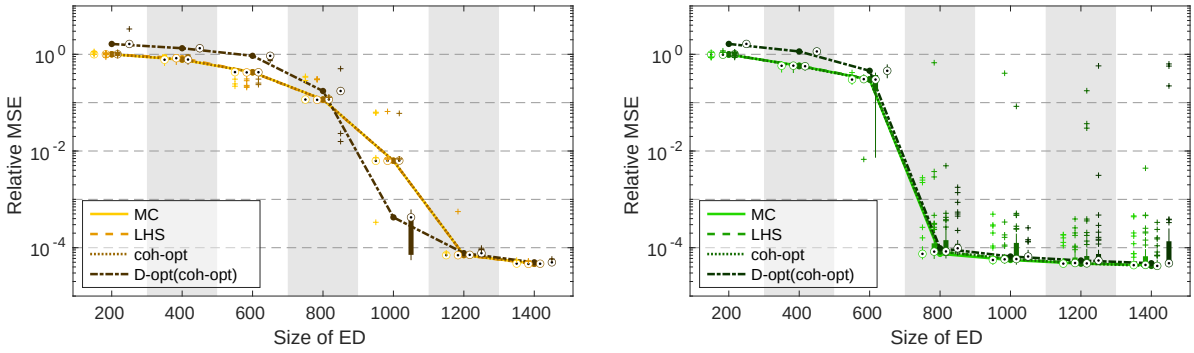
(c) SP **(d)** BCS

(e) Small ED (100 points) **(f)** Large ED (400 points)

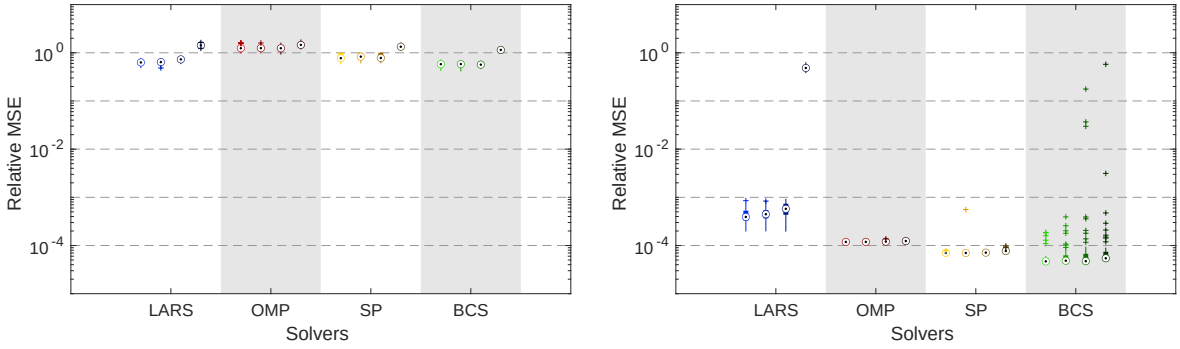
Figure 13: Boxplots of relative MSE from the benchmark of four solvers and four sampling schemes for the two-dimensional diffusion model ($d = 53, p = 4, q = 0.5$). Solvers are coded by colors. Sampling schemes are shown in varying shades and line styles. In (e) and (f), we show the relative MSE of each of the solvers combined with each sampling scheme in the order MC–LHS–coh-opt–D-opt(coh-opt).



(a) LARS **(b)** OMP

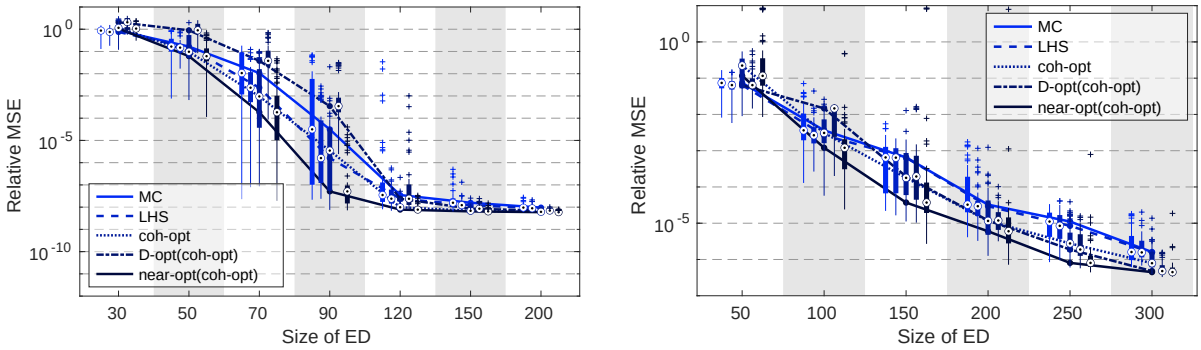


(c) SP **(d)** BCS



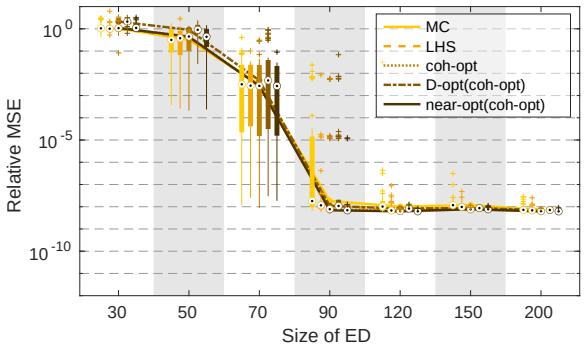
(e) Small ED (400 points) **(f)** Large ED (1200 points)

Figure 14: Boxplots of relative MSE from the benchmark of four solvers and four sampling schemes for the 100D function ($d = 100, p = 4, q = 0.5$). Solvers are coded by colors. Sampling schemes are shown in varying shades and line styles. In (e) and (f), we show the relative MSE of each of the solvers combined with each sampling scheme in the order MC–LHS–coh-opt–D-opt(coh-opt).

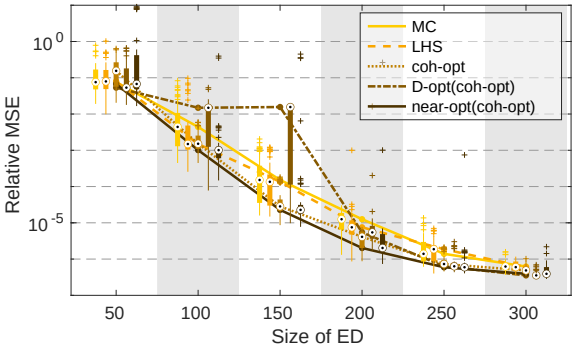


(a) Ishigami model, LARS

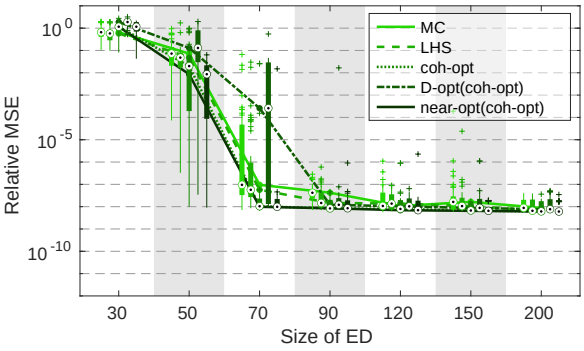
(b) Borehole model, LARS



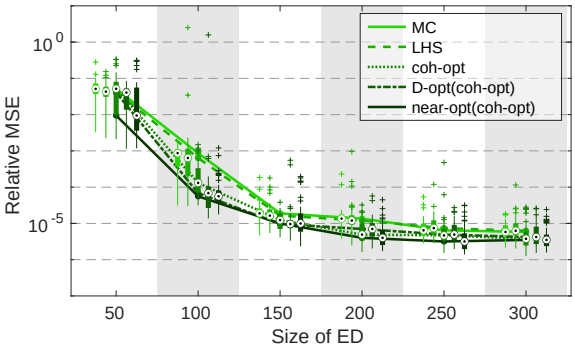
(c) Ishigami model, SP



(d) Borehole model, SP



(e) Ishigami model, BCS



(f) Borehole model, BCS

Figure 15: Left column: results for the Ishigami model with a smaller basis ($d = 3, p = 12, q = 1$), complementing the plots in Figure 5. Right column: results for the borehole model with a smaller basis ($d = 8, p = 4, q = 1$), complementing the plots in Figure 6. Results for three sparse solvers and five experimental design schemes. Fifty replications.

D Benchmark studies

An overview of articles and benchmark studies comparing sparse PCE methods, including their main results, is given in Table 4.

Table 4: Overview of some articles and benchmark studies comparing sparse PCE methods. The "best method" listed in the last column is the one delivering the smallest target error, as reported in the respective publications. Target quantities can be moments, Sobol' indices or the generalization error of the PCE surrogate. Abbreviations: '$\succ$' stands for 'better than'; for the acronyms of solvers and sampling schemes see Sections 2.5 and 2.6 or the respective publications; for the sampling schemes, the method given in parentheses indicates how the corresponding candidate set is created (e.g., D-opt(coh-opt) stands for D-optimal sampling based on a coherence-optimal candidate set). If the cited paper proposed a new method, this method is marked by a star (*).

Ref.	**Type**	**Methods compared**	**Result**
(Hampton and Doostan, 2015b,a)	sampling	MC, asymptotic*, and coh-opt*	coh-opt best
(Fajraoui et al., 2017)	sampling	LHS, Sobol, D-opt(LHS), S-opt(LHS)*; sequential sampling*	sequential S-opt(LHS) best; D-opt(LHS) worst
(Hadigol and Doostan, 2018)	sampling (OLS)	MC, LHS, coh-opt, A-opt(coh-opt)*, D-opt(coh-opt)*, E-opt(coh-opt)*	D-opt(coh-opt) best for $p > d$; MC, LHS best for $p < d$
(Jakeman et al., 2017)	sampling	MC, asymptotic, CSA*	for $p > d$: CSA better than MC and asymptotic
(Alemazkoor and Meidani, 2018a)	sampling	MC, coh-opt, near-opt(coh-opt)*	near-opt(coh-opt) $\succ$ coh-opt $\succ$ MC
(Diaz et al., 2018)	sampling	coh-opt, D-opt(coh-opt), sequential D-opt(coh-opt)*	seq. D-opt(coh-opt) $\succ$ D-opt(coh-opt) $\succ$ coh-opt
(Dutta and Gandomi, 2020)	sampling	MC, LHS, Sobol, Importance Sampling	LHS best
(Hu and Ludkovski, 2017)	solvers	OMP, SPGL1, BCS(Babacan et al., 2010)	BCS $\succ$ OMP $\succ$ SPGL1
(Huan et al., 2018)	solvers	l1_ls, SpaRSA, CGIST, FPC_AS, ADMM with default parameters	all showed similar performance; ADMM slightly advantageous
(Liu et al., 2020a)	solvers	OMP, LARS, rPCE*	rPCE $\succ$ LARS $\succ$ OMP
(Baptista et al., 2019)	solvers	OMP, SPGL1, and two variants* of OMP (modified regressor selection, randomization)	best: OMP, and OMP with modified regressor selection
(Tarakanov and Elsheikh, 2019)	solvers	OMP, LARS, Rank-PCE*	best: Rank-PCE
(Zhou et al., 2019)	solvers	LARS, BCS (Wipf and Rao, 2004), BCS (Ji et al., 2008), D-MORPH-reweighted (Cheng and Lu, 2018b), stepwise regression based on Bayesian ideas*	best: stepwise regression (based on 1 Sobol' design)

Supplementary material

Sparse Polynomial Chaos Expansions: Literature Survey and Benchmark

Nora Lüthen[1], Stefano Marelli[1], and Bruno Sudret[1]

[1] *Chair of Risk, Safety and Uncertainty Quantification, ETH Zürich, Stefano-Franscini-Platz 5, 8093 Zürich, Switzerland*

May 19, 2021

This description refers to UQLabModules V1.4.0.

UQLab is designed to be augmented by external code. We describe briefly how to connect external sampling schemes and solvers with the UQLab framework.

To use a custom sampling scheme, create the experimental design and add it to the structure array of PCE options `MetaOpts` as follows:

```
MetaOpts.ExpDesign.Sampling = 'user'; % optional
MetaOpts.ExpDesign.X = X;
MetaOpts.ExpDesign.Y = Y;
```

Make sure to define a suitable input object so that UQLab knows which family of polynomials to use. If the sampling scheme has weights, use the undocumented feature

```
MetaOpts.ExpDesign.CY = diag(1./(myWeights.^2));
```

where `myWeights` is the vector of weights corresponding to the points in `X`. This results in the weighted problem $\boldsymbol{W}\boldsymbol{\Psi}\boldsymbol{c} \approx \boldsymbol{W}\boldsymbol{y}$ with the diagonal matrix $\boldsymbol{W} = \mathrm{diag}(\texttt{myWeights})$.

To use a custom solver "mySolver" for sparse PCE in UQLab, define the wrapper function `uq_PCE_mysolver.m` (all lowercase) and use the option `metaopts.Method = 'MYSOLVER'` (all uppercase) for the PCE. A template for the implementation of `uq_PCE_mysolver.m` is displayed in File 1. See also `uq_PCE_lars.m` for an example implementation.

The function `uq_PCE_mysolver` takes the variables `univ_p_val` and `current_model`, where the first contains the univariate polynomial evaluations. The second argument contains all information about the PCE that is being constructed. Inside the wrapper function, call the custom solver, perform hyperparameter selection and error estimation, etc. The return variable `results` is a struct with the following fields:

- `.coefficients` : all coefficients, including the zeros, in the same order as the candidate basis
- `.indices` : all basis indices, including the inactive ones, in the same order as the coefficients
- `.normEmpErr` : empirical error
- `.LOO` : leave-one-out error (or a similar criterion for model selection)
- Also, a field `.optErrorParams` is required that needs to have two fields `.loo` and `.normEmpErr` (same as before).

Since basis adaptivity is independent of the choice of solver, each solver can make use of degree adaptivity, q-norm truncation, interaction restriction etc. as usual in UQLab. If the sampling scheme has weights, the solver must take them into account as displayed in File 1.

```
function results = uq_PCE_mysolver(univ_p_val, current_model)

% Get the index of the current output (for models with vector-valued output)
current_output = current_model.Internal.Runtime.current_output;

% Generate the regression matrix
Psi = uq_PCE_create_Psi(current_model.PCE(current_output).Basis.Indices, univ_p_val);

% Get the experimental design model evaluations
Y = current_model.ExpDesign.Y(:,current_output);

% Take weights into account, if they exist
if isfield(current_model.ExpDesign, 'CY')
    CY = current_model.ExpDesign.CY;
    CYinv = CY \ eye(size(CY));
    L = chol(CYinv);
    Psi = L*Psi;
    Y = L*Y;
end

% Now implement your method here
% e.g. [coeffs, LOO] = mysolver(Psi, Y, some_other_params);
...

% Prepare results
results.coefficients = ...
results.indices = current_model.PCE.Basis.Indices; % ALL indices
results.LOO = ...
results.normEmpErr = ...

opt_results.loo = results.LOO;
opt_results.normEmpErr = results.normEmpErr;
results.optErrorParams = opt_results;

end
```

File 1: Template for the implementation of a custom sparse solver for use in UQLab