Comput. Methods Appl. Mech. Engrg. 245–246 (2012) 36–46

Contents lists available at SciVerse ScienceDirect

Comput. Methods Appl. Mech. Engrg.

journal homepage: www.elsevier.com/locate/cma

Stochastic collocation with kernel density estimation ☆

Howard C. Elman [a,*], Christopher W. Miller [b]

[a] Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742, United States
[b] Department of Applied Mathematics and Scientific Computation, University of Maryland, College Park, MD 20742, United States

ARTICLE INFO

Article history:
Received 7 September 2011
Received in revised form 25 June 2012
Accepted 26 June 2012
Available online 16 July 2012

Keywords:
Stochastic partial differential equation
Stochastic collocation
Kernel density estimation
Adaptive

ABSTRACT

The stochastic collocation method has recently received much attention for solving partial differential equations posed with uncertainty, i.e., where coefficients in the differential operator, boundary terms or right-hand sides are random fields. Recent work has led to the formulation of an adaptive collocation method that is capable of accurately approximating functions with discontinuities and steep gradients. These methods, however, usually depend on an assumption that the random variables involved in expressing the uncertainty are independent with marginal probability distributions that are known explicitly. In this work we combine the adaptive collocation technique with kernel density estimation to approximate the statistics of the solution when the joint distribution of the random variables is unknown.

1. Problem statement

Let (Ω, Σ, P) be a complete probability space with sample space Ω, σ-algebra $\Sigma \subset 2^{\Omega}$ and probability measure $P : \Sigma \to [0,1]$. Let $D \subset \mathbb{R}^d$ be a d-dimensional bounded domain with boundary ∂D. We investigate partial differential equations (PDEs) of the form

$$\begin{aligned} \mathcal{L}(\mathbf{x}, \omega; u) &= f(\mathbf{x}), \quad \forall \mathbf{x} \in D, \quad \omega \in \Omega \\ \mathcal{B}(\mathbf{x}, \omega; u) &= g(\mathbf{x}), \quad \forall \mathbf{x} \in \partial D, \quad \omega \in \Omega. \end{aligned} \tag{1.1}$$

Here $\mathcal{L}$ is a partial differential operator with boundary operator $\mathcal{B}$, both of which can depend on the random parameter ω. As a consequence of the Doob–Dynkin lemma, it follows that u is also a random field, dependent on both the spatial location $\mathbf{x}$ and the event ω. In order to work numerically with the expressions in (1.1), we must first represent the operators in terms of a finite number of random variables $\boldsymbol{\xi} = [\xi_1, \xi_2, \ldots, \xi_M]^T$. This is often accomplished using a truncated Karhunen–Loève (KL) expansion [17]. If we denote $\Gamma = \mathit{Image}(\boldsymbol{\xi})$, then we can write (1.1) as

$$\begin{aligned} \mathcal{L}(\mathbf{x}, \boldsymbol{\xi}; u) &= f(\mathbf{x}), \quad \forall \mathbf{x} \in D, \quad \boldsymbol{\xi} \in \Gamma \\ \mathcal{B}(\mathbf{x}, \boldsymbol{\xi}; u) &= g(\mathbf{x}), \quad \forall \mathbf{x} \in \partial D, \quad \boldsymbol{\xi} \in \Gamma. \end{aligned} \tag{1.2}$$

For a given realization of the random vector $\boldsymbol{\xi}$, the system (1.2) is a deterministic partial differential equation that can be solved using a deterministic solver. Throughout this paper we assume that $D, \mathcal{L}, \mathcal{B}, f$, and g are defined so that the above problem (1.2) is well posed for all values of $\boldsymbol{\xi} \in \Gamma$. In this paper we will explore several different sampling methods for solving the system (1.2).

One is typically interested in methods that allow statistical properties of u to be computed. If $\rho(\boldsymbol{\xi})$ denotes the joint probability density function of the random vector $\boldsymbol{\xi}$, then the kth moment of the solution u is defined as

$$\mathbb{E}(u^k) = \int_{\Gamma} u^k \rho(\boldsymbol{\xi}) d\boldsymbol{\xi}. \tag{1.3}$$

One may also be interested in computing probability distributions associated with u, for example $P(u(\mathbf{x}, \boldsymbol{\xi}) \geqslant c)$.

Several methods have been developed for computing approximations to the random field u and the associated statistical quantities. The most widely known is the Monte–Carlo method, where the desired statistics are obtained by repeatedly sampling the distribution of $\boldsymbol{\xi}$, solving each of the resulting deterministic PDEs, and then estimating the desired quantities by averaging. Recently, much attention has been paid to alternative approaches such as the stochastic Galerkin and stochastic sparse grid collocation methods [2,9,12,22,21,27]. These methods typically approximate the solution u as a high-degree multivariate polynomial in $\boldsymbol{\xi}$. If this approximation is denoted $u_p(\mathbf{x}, \boldsymbol{\xi})$, then the error $u - u_p$ can be measured in terms of an augmented Sobolev norm

$$\|\cdot\|_{L^2_P;V} = \left(\int_{\Omega} \|\cdot\|_V^2 dP \right)^{\frac{1}{2}}. \tag{1.4}$$

Here V is an appropriate Sobolev space that depends on the spatial component of the problem and $\|\cdot\|_V$ is the norm over this space. It can be shown that as the total degree of the polynomial approximation is increased, the error in the above norm, $\|u - u_p\|_{L^2_P;V}$, decays very rapidly provided that the solution u is sufficiently smooth in

☆ This work was supported in part by the US Department of Energy under Grant DEFG0204ER25619 and the US National Science Foundation under Grants CCF0726017 and DMS1115317.

* Corresponding author.
E-mail addresses: elman@cs.umd.edu (H.C. Elman), cmiller@math.umd.edu (C.W. Miller).

0045-7825/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.cma.2012.06.020

ξ [22]. If u is not sufficiently smooth then the convergence of these methods can stall or they may not converge at all [18]. Several methods have been proposed for treating problems that are discontinuous in the stochastic space. One approach partitions the stochastic space into elements and approximates the solution locally within elements by polynomials, continuous on the domain [3,25]. Another approach is to use a hierarchical basis method developed in [16], which approximates u using a hierarchical basis of piecewise linear functions defined on a sparse grid. This idea was used with stochastic collocation in [18] where the sparse grid is refined adaptively using an a posteriori error estimator.

If the truncated Karhunen–Loève expansion is used to express $\mathcal{L}$ and $\mathcal{B}$, then the random variables $\xi_1, \xi_2, \ldots, \xi_M$ have zero mean and are uncorrelated [17]. It is frequently assumed that the random variables are independent and that their marginal density functions $\rho_i(\xi_i)$ are known explicitly. In this case the joint density function is simply the product of the marginal densities $\rho(\xi) = \Pi_{k=1}^{M} \rho_i(\xi_i)$. This assumption simplifies the evaluation of the moments of the solution since the multidimensional integral in (1.3) can be written as the product of one-dimensional integrals. It is not the case, however, that uncorrelated random variables are necessarily independent, and in the worst case the support of the product of the marginal densities may contain points that are not in the support of the true joint density. Thus, it may not be appropriate to define the joint density function as the product of the marginal density functions. See [13] for further discussion of this point. In this paper we explore a method for approximating the statistics of the solution u when an explicit form of the joint distribution is not available and we only have access to a finite number of samples of the random vector ξ. In particular, we are able to treat the case where information on the parameters of the problem is only available in the form of experimental data. The method works by constructing an approximation $\hat{\rho}(\xi)$ to the joint probability distribution $\rho(\xi)$ using kernel density estimations [23]. This construction is then combined with an adaptive collocation strategy similar to the one derived in [18] to compute an approximation to the random field u. Moments can then be efficiently evaluated by integrating this approximation with respect to the approximate probability measure $\hat{\rho}(\xi)$.

The remainder of this paper proceeds as follows. Section 2 discusses the adaptive collocation method in [18]. Section 3 presents an overview of the kernel density estimation technique used for approximating the unknown distribution of ξ. Section 4 presents the method developed in this paper for approximating solutions to problems of the form (1.2). An error bound for the method is given in Section 4.1, and Section 4.2 presents techniques for extracting solution statistics. Section 5 presents the results of numerical experiments showing the performance of the new method and comparing this performance with that of the Monte Carlo method. Finally in Section 6 we draw some conclusions.

2. The adaptive collocation method

Collocation methods work by solving the Eq. (1.2) for a finite number of pre-determined parameters $\{\xi^{(1)}, \ldots, \xi^{(N_c)}\}$ using a suitable deterministic solver. The solutions at each sample point are then used to construct an interpolant to the solution for arbitrary choices of the random vector ξ. We denote such an approximation generally as $\mathcal{A}(u)(\xi)$. Collocation methods were first used for solving PDEs with random coefficients in [2]. The interpolant was formed using a Lagrangian polynomial basis defined on tensor product grids. The cardinality of these grids is exponential in the dimension of the random vector so that this method is not viable for problems with high-dimensional random inputs. Sparse grid collocation methods were developed in [27] and an error analysis of the method was presented in [22]. These methods use the Smolyak interpolation formula [24] to construct a high-order polynomial interpolant using many fewer points than the full tensor grid. A refinement of this method for problems where the solution depends on the parameters in an anisotropic manner was presented in [21]. For all of these methods, the solution random field is expressed globally as a polynomial in the random vector ξ. These methods are therefore only useful when the random field u is sufficiently regular in ξ.

An adaptive collocation method was developed in [18]. This method is designed to compute approximations of random fields that possess discontinuities or strong gradients, and for which the image set Γ is bounded.[1] In the following, we present an overview of this method and our proposed modifications. To simplify the presentation we describe the case of a function u defined by a single random parameter whose image is a subset of $[0,1]$. This can be generalized in a straightforward manner to a function defined by M parameters with image contained in any M-dimensional hypercube. Define

$$m_i = \begin{cases} 1 & \text{if } i = 1, \\ 2^{i-1} + 1 & \text{if } i > 1, \end{cases} \tag{2.1}$$

$$\xi_j^i = \begin{cases} \frac{j-1}{m_i - 1} & \text{for } j = 1, \ldots, m_i, \quad \text{if } m_i > 1, \\ 0.5 & \text{for } j = 1, \quad \text{if } m_i = 1. \end{cases} \tag{2.2}$$

For $i = 1, 2, \ldots$, we have that $\theta^i = \{\xi_j^i\}_{j=1}^{m_i}$ consists of m_i distinct equally spaced points on $[0,1]$. We also have that $\theta^i \subset \theta^{i+1}$. Since these points are equidistant, the use of global polynomial interpolation as in [27] is not appropriate due to the Runge phenomenon. We make no assumptions on the smoothness of u; for example, it may contain singularities that global polynomial approximations will not resolve. To address these issues, a hierarchical basis of piecewise linear functions is used to construct the interpolant. Define $\theta^0 = \emptyset$ and $\Delta\theta^i = \theta^i \setminus \theta^{i-1}$. Note that $|\Delta\theta^i| = m_i - m_{i-1}$. Let the members of $\Delta\theta^i$ be denoted $\{\xi_j^{\Delta i}\}_{j=0}^{|\Delta\theta^i|-1}$. The hierarchical basis is defined on the interval $[0,1]$ as

$$a_0^1(\xi) = 1 \tag{2.3}$$

$$a_j^i(\xi) = \begin{cases} 1 - (m_i - 1)|\xi - \xi_j^{\Delta i}| & \text{if } |\xi - \xi_j^{\Delta i}| < 1/(m_i - 1), \\ 0 & \text{otherwise}, \end{cases} \tag{2.4}$$

for $i > 1$ and $j = 0, \ldots, |\Delta\theta^i| - 1$; see Fig. 2.1. These functions are piecewise linear and have the property that $a_j^i(\xi_k^{\Delta i}) = \delta_{jk}$, and $a_j^i(\xi_k^s) = 0$ for all $s < i$. Note that there is a binary tree structure on the nodes in θ^i. That is, we can define the set of children of a point $\xi_j^{\Delta i}$ as

$$child(\xi_j^{\Delta i}) = \begin{cases} \{\xi_j^{\Delta i+1}\} & \text{if } i = 2 \\ \{\xi_{2j}^{\Delta i+1}, \xi_{2j+1}^{\Delta i+1}\} & \text{otherwise}. \end{cases} \tag{2.5}$$

We also denote the parent of a point in this tree as $par(\xi_j^{\Delta i})$.

Algorithm 1 defines an interpolation scheme using the hierarchical basis functions.

Algorithm 1. Interpolation with hierarchical basis functions

Define $\mathcal{A}_0(u)(\xi) = 0$.
Define $k = 1$
repeat
  Construct $\Delta\theta^k$
  Evaluate $u(\xi_j^{\Delta k})\ \forall \xi_j^{\Delta k} \in \Delta\theta^k$
  $w_j^k = u(\xi_j^{\Delta k}) - \mathcal{A}_{k-1}(u)(\xi_j^{\Delta k})\ \forall \xi_j^{\Delta k} \in \Delta\theta^k$
  Define $\mathcal{A}_k(u)(\xi) = \sum_{i=1}^{k} \sum_{j=0}^{|\Delta\theta^i|-1} w_j^i a_j^i(\xi)$.
  $k = k + 1$
until $max(|w_j^{k-1}|) < \tau$

[1] For unbounded Γ, interpolation is carried out on a bounded subset of Γ, see e.g. [26].

The quantities $\{w_j^k\}$ are referred to as the hierarchical surplus. They represent the correction to the interpolant $\mathcal{A}_{i-1}(u)$ at the points in $\Delta\theta_i$. For functions with values that vary dramatically at neighboring points, the hierarchical surpluses $\{w_j^i\}$ remain large for several iterations. This provides us with a natural error indicator as well as a convergence criterion for the method, whereby we require that the largest hierarchical surplus be smaller than a given tolerance. The hierarchical surpluses also provide a mechanism to implement adaptive grid refinement. The grid is adaptively refined at points with large hierarchical surpluses. For such a point, its children are added to the next level of the grid. Algorithm 2 defines such an adaptive interpolation algorithm that is similar to the one appearing in [18].

Algorithm 2. Adaptive interpolation with hierarchical basis functions

Define $\mathcal{A}_0(u)(\xi) = 0$.
Define $k = 1$
Initialize $\Delta\theta^1_{adaptive} = \theta^1$.
repeat
  $\Delta\theta^{k+1}_{adaptive} = \emptyset$
  for $\xi_j^{\Delta k} \in \Delta\theta^k_{adaptive}$ **do**
    Evaluate $u(\xi_j^{\Delta k})$
    $w_j^k = u(\xi_j^{\Delta k}) - \mathcal{A}_{k-1}(u)(\xi_j^{\Delta k})$
    if $\|w_j^k\| > \tau$ **then**
      $\Delta\theta^{k+1}_{adaptive} = \Delta\theta^{k+1}_{adaptive} \cup child(\xi_j^{\Delta k})$
    end if
  end for
  Define $\mathcal{A}_k(u)(\xi) = \sum_{i=1}^k \sum_j w_j^i a_j^i(\xi)$.
  $k = k + 1$
until $max(|w_j^{k-1}|) < \tau$

This method can be generalized in a straightforward way to functions defined on $[0,1]^M$. All that is needed is to define a multidimensional hierarchical basis set and a method for generating the children of a given grid point. The multidimensional hierarchical basis consists of tensor products of the one-dimensional hierarchical basis functions. Given $\mathbf{i} = [i_1, \ldots, i_M] \in \mathbb{N}^M$ and $\mathbf{j} = [j_1, \ldots, j_M] \in \mathbb{N}^M$, let

$$a^{\mathbf{i}}_{\mathbf{j}}(\boldsymbol{\xi}) = a^{i_1}_{j_1}(\xi_1) \otimes \cdots \otimes a^{i_M}_{j_M}(\xi_M). \tag{2.6}$$

We can define the multidimensional interpolation grids by

$$\theta_1 = [0.5, 0.5, \ldots, 0.5] \tag{2.7}$$
$$child(\xi^{\Delta\mathbf{i}}_{\mathbf{j}}) = \{\boldsymbol{\xi} | \exists! j \in 1, \ldots, M \text{ s.t. } [\xi_1, \ldots, \xi_{j-1}, par(\xi_j), \xi_{j+1}, \ldots \xi_M] = \xi^{\Delta\mathbf{i}}_{\mathbf{j}}\}.$$

From this we can see that each grid point has at most $2M$ children.

It was shown in [15] that the interpolation error associated with this method bounded by

$$\mathcal{O}(|\theta^k|^{-2} log(|\theta^k|^{3(M+1)})). \tag{2.8}$$

This bound grows rapidly with increasing dimension. Numerical experiments presented in [18] show that, in practice, the interpolation error is significantly smaller than this bound, both for smooth functions and functions that contain steep gradients or discontinuities.

This method can be used to approximate the solutions to (1.2) by applying a suitable deterministic solver to the equations at collocation points $\xi^{\Delta\mathbf{i}}_{\mathbf{j}}$. We can then construct an interpolant of u, $\mathcal{A}_k(u)$ using the formula in Algorithm 2. In principle, the expected value of u can be approximated by

$$\mathbb{E}(u) \approx \int_\Gamma \mathcal{A}_k(u)\rho(\boldsymbol{\xi})d\boldsymbol{\xi} = \sum_{\mathbf{i}}\sum_{\mathbf{j}} w^{\mathbf{i}}_{\mathbf{j}} \int_\Gamma a^{\mathbf{i}}_{\mathbf{j}}(\boldsymbol{\xi})\rho(\boldsymbol{\xi})d\boldsymbol{\xi}, \tag{2.9}$$

although in the cases under discussion ρ will not be known explicitly. Even in the case where ρ is known explicitly and can be expressed as the product of univariate functions, the integral in (2.9) can still be difficult to calculate when it is of high dimension.

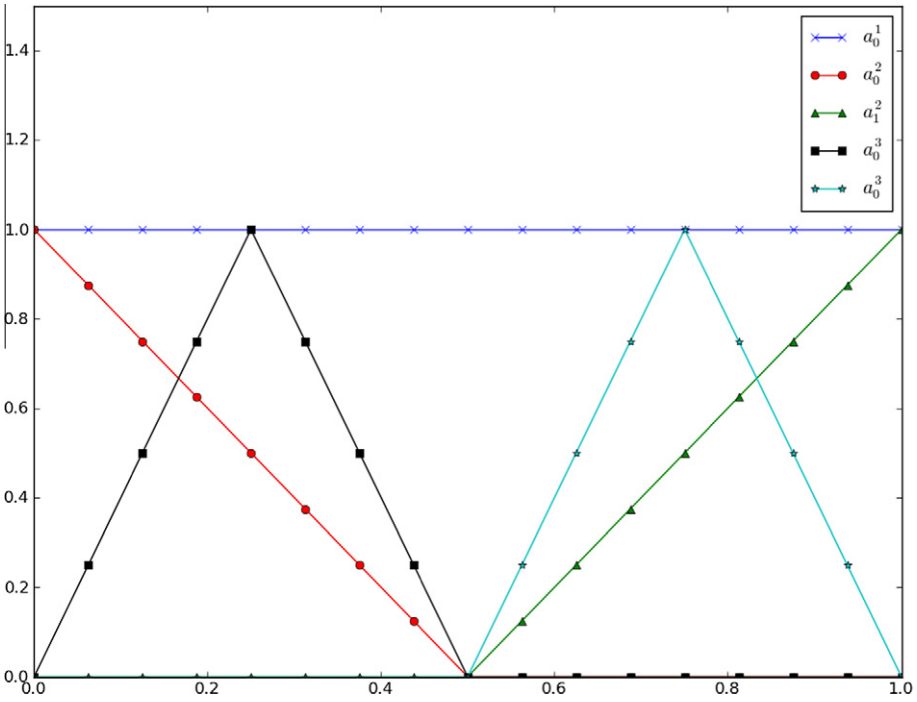

Fig. 2.1. The hierarchical basis functions for $i = 1, 2, 3$.

3. Kernel density estimation

Let $K(\xi)$ be a function satisfying the following conditions:

$$\int_{\mathbb{R}^M} K(\xi)d\xi = 1, \tag{3.1}$$
$$\int_{\mathbb{R}^M} K(\xi)\xi d\xi = 0,$$
$$\int_{\mathbb{R}^M} K(\xi)\|\xi\|^2 d\xi = k_2 < \infty,$$
$$K(\xi) \geqslant 0,$$

where $\|\xi\|$ is the Euclidean norm of the M-dimensional vector ξ. Let $\xi^{(1)}, \xi^{(2)}, \ldots, \xi^{(N)}$ be N independent realizations of the random vector ξ. The kernel density approximation to the joint distribution of ξ is given by

$$\hat{\rho}(\xi) = \frac{1}{Nh^M}\sum_{k=1}^{N} K\left(\frac{\xi - \xi^{(i)}}{h}\right), \tag{3.2}$$

where h is a user-defined parameter called the bandwidth. It is straightforward to verify that the function $\hat{\rho}$ defined above satisfies the conditions for being a probability density function. The main challenge here lies in the selection of an appropriate value for h. If h is chosen to be too large then the resulting estimate is said to be oversmoothed and important features of the data may be obscured. If h is chosen to be too small then the resulting estimate is said to be undersmoothed and the approximation may contain many spurious features not present in the true distribution. Fig. 3.1 shows kernel density estimates of a bimodal distribution for a small and large value of h. The oversmoothed estimate does not detect the bimodality of the data whereas the undersmoothed estimate introduces spurious oscillations into the estimate.

One method for specifying h is to choose the value that minimizes the approximate mean integrated square error (*AMISE*). For a given value of h, the *AMISE* is given by

$$AMISE(h, N) = \frac{1}{4}h^4\alpha^2 \int_{\mathbb{R}^M} (\Delta\rho(\xi))^2 d\xi + N^{-1}h^{-M}\beta, \tag{3.3}$$

where

$$\alpha = \int_{\mathbb{R}^M} \|\xi\|_1^2 K(\xi)d\xi, \quad \beta = \int_{\mathbb{R}^M} K(\xi)^2 d\xi, \tag{3.4}$$

and Δ here denotes the Laplace operator [23]. From this expression the optimal value of h can be derived as [23]

$$h_{opt}^{M+4} = M\beta\alpha^{-2}\left\{\int (\Delta\rho(\xi))^2 d\xi\right\}^{-1} N^{-1}. \tag{3.5}$$

It can be shown that the optimal bandwidth is of magnitude $\mathcal{O}(N^{-1/(M+4)})$ as the number of samples N increases. If the optimal value of h is used it can also be shown that the *AMISE* decays like $\mathcal{O}(N^{-\frac{4}{4+M}})$.

For numerical computations, choosing h to minimize the *AMISE* is impractical since it requires a priori knowledge of the exact distribution. Many techniques have been proposed for choosing the smoothing parameter h without a priori knowledge of the underlying distribution, including least-squares cross-validation and maximum likelihood cross-validation [23]. In the numerical experiments below we employ maximum likelihood cross-validation (MLCV). This method proceeds as follows. Given a finite set of samples, $\xi^{(1)}, \xi^{(2)}, \ldots, \xi^{(N)}$, of the random vector ξ, define

$$\hat{\rho}_{-i}(\xi) = \frac{1}{Nh^M}\sum_{k=1, k\neq i}^{N} K\left(\frac{\xi - \xi^{(k)}}{h}\right) \tag{3.6}$$

to be the kernel density estimate constructed by omitting the ith sample. The maximum likelihood cross-validation method is to choose h that maximizes

$$CV(h) \equiv \frac{1}{N}\sum_{i=1}^{N} log(\hat{\rho}_{-i}(\xi^{(i)})). \tag{3.7}$$

Note that this value of h only depends on the data. The intuition behind this method is that if we are given an approximation to the true density based on $N-1$ samples and we draw another sample, then the approximate density should be large at this new sample point. In the numerical experiments described below, we solved this optimization problem using Brent's method [6]. The asymptotic cost of evaluating (3.7) is $\mathcal{O}(N^2)$. Thus as the number of samples grows large this method can become costly. In this case one typically only uses a randomly selected subset of the samples to evaluate (3.7) [14]. In the numerical experiments described below, we observed that for the sample sizes used, the cost of this optimization was significantly lower than the cost of repeatedly solving the algebraic systems of equations that arise from the spatial discretization of the PDE (1.2).

In [23] it is shown that the choice of kernel does not have a strong effect on the error associated with kernel density estimation. In our experiments we use the multivariate Epanechnikov kernel

$$K(\xi) = \left(\frac{3}{4}\right)^M \prod_{i=1}^{M}(1 - \xi_i^2)\mathbb{1}_{\{-1\leqslant\xi_i\leqslant 1\}}. \tag{3.8}$$

This kernel is frequently used in the case of univariate data as it minimizes the asymptotic mean integrated square error over all choices of kernels satisfying (3.1). It also has the advantage that it is compactly supported. This causes the approximate density function $\hat{\rho}$ to be compactly supported, which is important in assuring the well-posedness of some stochastic partial differential equations.

4. Adaptive collocation with KDE driven grid refinement

The interpolation method in [18] distributes interpolation nodes so that discontinuities and steep gradients in the solution function are resolved; however the method does not take into account how significant a given interpolation node is to the statistics of the solution function since the refinement process does not depend on ρ. The kernel density estimate described above can also be used to drive refinement of the adaptive sparse grid in Algorithm 2. The algorithm we propose is as follows. First construct an estimate $\hat{\rho}$ to the true density ρ using a finite number of samples $\{\xi^{(i)}\}_{i=1}^N$. Second, replace the refinement criterion in Algorithm 2 with

$$|w_j^k|\hat{\rho}(\xi_j^{\Delta k}) > \tau. \tag{4.1}$$

A similar approach is used in [19] to drive the refinement. However in that study it is again assumed that one has access to an explicit form of the joint density function. With the refinement criterion (4.1), the grid is only adaptively refined at points near the data $\{\xi^{(i)}\}_{i=1}^N$ since the kernel density estimate is only supported near the samples. In the sequel we refer to this proposed method, i.e., Algorithm 2 with refinement criterion (4.1), as *adaptive KDE collocation*. The refinement criterion (4.1) could also be employed in any method where the stochastic domain can be refined locally, e.g., the multi-element stochastic collocation method [10,11]. The remainder of this section is divided into two parts. In Section 4.1 we present interpolation error estimates associated with adaptive KDE collocation and in Section 4.2 we present methods for approximating the solution statistics of the random field u. Note that throughout this discussion we can ignore the spatial component of the problem.

4.1. Error analysis of adaptive KDE collocation

For simplicity we present the results for the case where the problem only depends on a single parameter and interpolation is

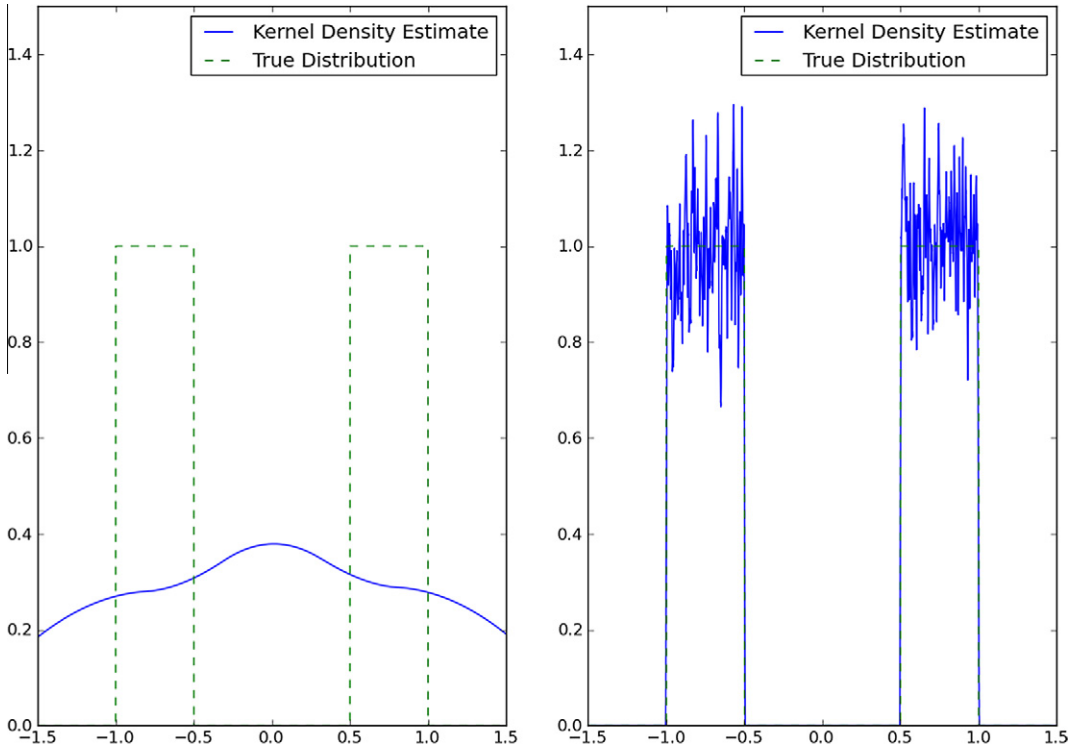

Fig. 3.1. Under-smoothed kernel density estimate (left) and over-smoothed (right).

carried out on $[0,1]$. Extension of the argument to multi-parameter problems defined on an arbitrary hypercube is straightforward. Also we ignore the spatial component of the problem as it has no effect on the discussion of the errors resulting from the discretization of the stochastic portion of the problem. Assume that $\mathcal{A}_k(u)$ is an interpolant generated using adaptive KDE collocation with tolerance τ. Let $\hat{\rho}$ be the kernel density estimate used in computing $\mathcal{A}_k$ and let $\hat{\Gamma}$ be the support of $\hat{\rho}$. Let $\mathcal{A}_k^{complete}(u)$ be the interpolant constructed by Algorithm 1 with grid points $\Delta\theta^k = \{\xi_j^{\Delta i}\}$ and set of hierarchical surpluses $\{w_j^i\}$ at those grid points. By definition, $\Delta\theta^k_{adaptive} \subset \Delta\theta^k$. Define $\Delta\theta^k_{remaining} = \Delta\theta^k - \Delta\theta^k_{adaptive}$. Then if $\xi_j^{\Delta i} \in \Delta\theta^k_{remaining}$, it follows from (4.1) that $|w_j^i \hat{\rho}(\xi_j^{\Delta i})| \leqslant \tau$. We can bound the difference between u and $\mathcal{A}_k(u)$ on $\hat{\Gamma}$ as

$$\|(u-\mathcal{A}_k(u))\rho\|_{L_\infty(\hat{\Gamma})} \leqslant \left\|\frac{\rho}{\hat{\rho}}\right\|_{L_\infty(\hat{\Gamma})} (\underbrace{\|(u-\mathcal{A}_k^{complete}(u))\hat{\rho}\|_{L_\infty(\hat{\Gamma})}}_{\epsilon_1} + \underbrace{\|(\mathcal{A}_k^{complete}(u)-\mathcal{A}_k(u))\hat{\rho}\|_{L_\infty(\hat{\Gamma})}}_{\epsilon_2}). \tag{4.2}$$

The term ϵ_1 is the interpolation error associated with piecewise multilinear approximation on a full grid. This case is studied in [15]. The interpolation error is bounded by

$$\|u-\mathcal{A}_k^{complete}(u)\|_{L_\infty(\Gamma)} = \mathcal{O}(|\Delta\theta^k|^{-2}|log_2(|\Delta\theta^k|)|^{3(M-1)}) \tag{4.3}$$

Since $\hat{\rho}$ is bounded it follows that the bound on ϵ_1 decays at the same rate.

Bounding ϵ_2 depends on counting the points in $\Delta\theta^k_{remaining}$ and using the fact that at those points $|w_j^i\hat{\rho}| \leqslant \tau$. We have that

$$\|(\mathcal{A}_k^{complete}(u)-\mathcal{A}_k(u))\hat{\rho}\|_{L_\infty(\Gamma)} \leqslant \sum_{\Delta\theta^k_{remaining}} |w_j^i| \|a_j^i(\xi)\hat{\rho}(\xi)\|_{L_\infty(\Gamma)}. \tag{4.4}$$

Expanding $\hat{\rho}$ in a Taylor series around $\xi_j^{\Delta i}$ and noting that $a_j^i(\xi)\hat{\rho}(\xi)$ is only supported on an interval of size $\frac{1}{2^i}$ gives

$$\|(\mathcal{A}_k^{complete}(u)-\mathcal{A}_k(u))\hat{\rho}\|_{L_\infty(\Gamma)} \leqslant \sum_{\Delta\theta^k_{remaining}} |w_j^i\hat{\rho}(\xi_j^{\Delta i})| + |w_j^i|\frac{\|\hat{\rho}'\|_{L_\infty(\Gamma)}}{2^i} \leqslant \tau|\Delta\theta^k_{remaining}| + \sum_{\Delta\theta^k_{remaining}} |w_j^i|\frac{\|\hat{\rho}'\|_{L_\infty(\Gamma)}}{2^i}. \tag{4.5}$$

The sums here are over all $\mathbf{i},\mathbf{j}$ such that $\xi_{\mathbf{j}}^{\Delta\mathbf{i}} \in \Delta\Theta^k_{remaining}$. For decreasing τ, the number of points in $\Delta\theta^k_{remaining}$ decreases, since more points are locally refined and those points that remain in $\Delta\theta^k_{remaining}$ for large k correspond to basis functions with very small support. If τ is chosen to be small and k is allowed to grow so that the refinement criterion (4.1) is satisfied at every leaf node, the term ϵ_2 will converge to zero.

4.2. Estimation of solution statistics

Computation of the moments of the solution via the methods presented in [2,3,12,18,22,27] all require that the joint density function ρ be explicitly available in order to evaluate the integral $\int_\Gamma \hat{u}(\mathbf{x},\xi)\rho(\xi)d\xi$ where $\hat{u}$ is an approximation to u computed by either the stochastic Galerkin method [3,12] or by the stochastic collocation method [2,18,22,27]. In practice this may be an unrealistic assumption since we often only have access to a finite sample from the distribution of ξ. This section describes two ways of approximating the solution statistics when only a random sample from the distribution of ξ is available. The first is the well-known Monte–Carlo method [20]; the second is a variant of the Monte–Carlo predictor method presented in [26].

Given a random field $u(x,\xi)$ and a finite number of samples $\{\xi^{(i)}\}_{i=1}^N$, the Monte–Carlo method approximates the mean of u by the sample mean

$$\mathbb{E}(u)(\mathbf{x}) \approx \frac{1}{N}\sum_{i=1}^N u(\mathbf{x},\xi^{(i)}) \equiv \bar{u}(\mathbf{x}). \tag{4.6}$$

This method has the advantage that the convergence is independent of the dimension of the random parameter. The error in the expected value can be approximated by first noting that the estimate is unbiased,

$$Bias_{MC} = \mathbb{E}(u)(\mathbf{x}) - \mathbb{E}\left(\frac{1}{N}\sum_{i=1}^{N} u(\mathbf{x}, \boldsymbol{\xi}^{(i)})\right) = 0, \tag{4.7}$$

and that

$$Var(\bar{u}(\mathbf{x})) = \frac{Var(u(\mathbf{x}, \xi))}{N}, \tag{4.8}$$

where $Var(\bar{u}(\mathbf{x}))$ is the variance of the sample mean. An application of Chebyshev's inequality then gives a standard probabilistic estimate, that for $a > 0$,

$$P\left(\left|\mathbb{E}(u)(\mathbf{x}) - \frac{1}{N}\sum_{i=1}^{N} u(\mathbf{x}, \boldsymbol{\xi}^{(i)})\right| \geqslant a\right) \leqslant \frac{Var(u)}{Na^2}. \tag{4.9}$$

Note that a factor of 2 error reduction requires an increase of the sample size by a factor of 4. This slow rate of convergence is often cited as the chief difficulty in using the Monte–Carlo method [2,12]. It is also important to note that this bound is probabilistic in nature and that it is possible for the Monte–Carlo method to perform much worse (or much better) than expected. For a fixed choice of the quantity on the left hand side of (4.9), which we call P here, say $P = .05$, we have that

$$a \leqslant \sqrt{\frac{Var(u)}{.05N}}, \tag{4.10}$$

and from this we can conclude with 95% percent confidence that the Monte–Carlo estimate is bounded by $\sqrt{\frac{Var(u)}{.05N}}$. Smaller values of P lead to looser bounds but greater confidence in those bounds.

The method presented in [26] is to construct an approximation $\hat{u}$ of the solution function in the stochastic space using conventional sparse grid collocation and then, given a finite number of samples $\{\boldsymbol{\xi}^{(i)}\}_{i=1}^{N}$, to approximate the expected value by

$$\mathbb{E}(u)(x) \approx \frac{1}{N}\sum_{i=1}^{N} \hat{u}(\mathbf{x}, \boldsymbol{\xi}^{(i)}). \tag{4.11}$$

Instead of using conventional sparse grid collocation, we construct an approximation $\hat{u}$ using the adaptive KDE collocation method. Assuming that one has already constructed the interpolant, computation of the expected value can be carried out very quickly this way since the interpolant is simple to evaluate. Note also that while the standard Monte–Carlo method was used to evaluate (4.11), adaptive KDE collocation is also compatible with other sampling methods such as quasi-Monte Carlo [7], multilevel Monte–Carlo [4,8]. In the case of quasi-Monte Carlo, the sample points used in (4.11) are simply chosen to be the quasi-Monte Carlo sample points, and in the case of multilevel Monte–Carlo an expression similar to (4.11) is computed at each level of the computation. We expect sampling strategies would yield combined benefits; we do not explore this issue here.

The error associated with this method separates into two terms as follows,

$$\begin{aligned}|\epsilon_{sparse}| &= \left|\mathbb{E}(u)(\mathbf{x}) - \frac{1}{N}\sum_{i=1}^{N} \mathcal{A}(u)(\mathbf{x}, \boldsymbol{\xi}^{(i)})\right| \leqslant \left|\mathbb{E}(u)(\mathbf{x}) - \frac{1}{N}\sum_{i=1}^{N} u(\mathbf{x}, \boldsymbol{\xi}^{(i)})\right| \\ &\quad + \left|\frac{1}{N}\sum_{i=1}^{N}((\mathbf{x}, \boldsymbol{\xi}^{(i)}) - \mathcal{A}(u)(\mathbf{x}, \boldsymbol{\xi}^{(i)}))\right| = \epsilon_{MC} + \epsilon_{interp}.\end{aligned} \tag{4.12}$$

The first term is statistical error and depends only on the number of samples taken and the variance of u, and decays according to (4.9). The second term is the interpolation error and is bounded since the infinity norm of the interpolation error is bounded in the neighborhood of the sample points using (4.2).

Given N samples of ξ, evaluation of (4.6) requires N evaluations of the random field u. In the case where u is defined by a system such as (1.2), this requires N solutions of a discrete PDE. In contrast, evaluation of (4.11) requires N_{interp} evaluations of u to construct $\mathcal{A}(u)$ and then it requires N evaluations of $\mathcal{A}(u)$. The relative computational efficiency of (4.11) then depends on two factors: first, whether an accurate interpolant $\mathcal{A}(u)$ can be constructed using $N_{interp} \ll N$ function evaluations, and second, whether the cost of evaluating $\mathcal{A}(u)$ is significantly less than the cost of evaluating u. The first condition, as shown by (4.3), depends on the dimension of the problem as well as the number of samples we have access to. For most problems of interest the second condition is satisfied in that it is much less expensive to evaluate a piecewise polynomial than it is to solve a discrete algebraic system associated with a complex physical model. Note that in order for ϵ_{interp} to be small the interpolation error only needs to be small near the sample points. For adaptive KDE collocation the kernel density estimate is designed to make the interpolant more accurate in the neighborhoods of these points by indicating where large clusters of points are located.

4.3. Analysis of errors arising from spatial discretization

Thus far, we have focused on the statistical errors associated with the adaptive KDE collocation method in the absence of errors arising from spatial discretization. Within the context of stochastic partial differential equations, however, errors are also introduced by discretizing the equation in space at each collocation point. Generally this error is analyzed by separating the error into spatial and stochastic components [2,3,22]. We will present an outline of this approach here.

As above, let $u(\mathbf{x}, \xi)$ be the solution to (1.2). Let $u_h(\mathbf{x}, \xi)$ be an approximation to $u(\cdot, \xi)$ obtained by a discrete deterministic PDE solver. Let $\mathcal{A}(u_h)$ be the approximation to $u_h(\cdot, \xi)$ obtained by adaptive KDE collocation, that is, a discrete PDE solver is used to solve (1.2) at each collocation point. The approximation error in both the spatial and probabilistic dimensions can be written as

$$\begin{aligned}&\|[u(\mathbf{x}, \xi) - \mathcal{A}(u_h)(\mathbf{x}, \xi)]\rho\|_{L_\infty(\Gamma);V} \\ &\leqslant \|[u(\mathbf{x}, \xi) - u_h(\mathbf{x}, \xi)]\rho\|_{L_\infty(\Gamma);V} + \|[u_h - \mathcal{A}(u_h)(\mathbf{x}, \xi)]\rho\|_{L_\infty(\Gamma);V},\end{aligned} \tag{4.13}$$

where $\|\cdot\|_{L_\infty(\Gamma);V}$ is the tensor product norm induced by the L_∞ norm on Γ and a suitable norm defined on D, e.g., a Sobolev norm. The first term in (4.13) is associated with the spatial discretization and can be bounded using standard techniques for deterministic problems [1,5]. The second term is the interpolation error on Γ and is bounded by (4.2). In a practical computation, the spatial and stochastic discretizations should be configured so that these two errors are approximately equal. Our concern in this study is the stochastic component and we restrict our attention to this in the sequel.

5. Numerical experiments

In this section we assess the performance of adaptive KDE collocation applied to several test problems. We aim to measure quantitatively the two terms in the estimate (4.2) and to compare the computational efficiency of our method with the Monte–Carlo method.

5.1. Example 1: interpolation of a highly oscillatory function

Before exploring our main concern, the solution of PDEs with stochastic coefficients, we first examine the utility of adaptive collocation for performing a simpler task, to interpolate a

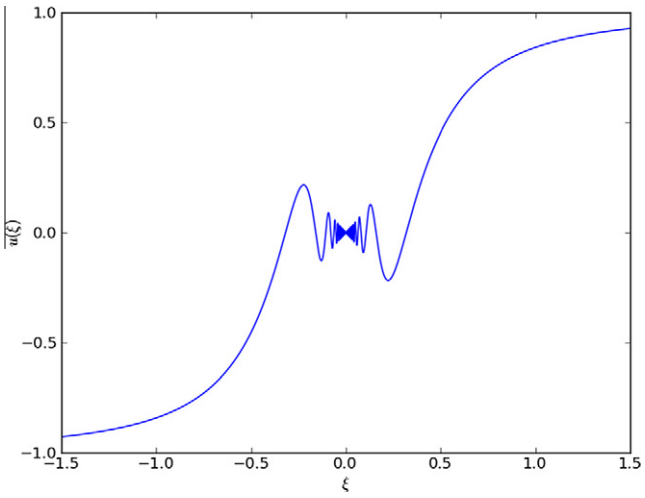

Fig. 5.1. $u(\xi) = |\xi| sin(1/\xi)$.

scalar-valued function whose argument is a random vector. We use adaptive KDE collocation to construct an approximation to the function

$$u(\boldsymbol{\xi}) = \begin{cases} \prod_{k=1}^{M} |\xi_k| sin(1/\xi_k) & \text{if } \xi_k \neq 0 \\ 0 & \text{otherwise,} \end{cases} \tag{5.1}$$

where $\boldsymbol{\xi}$ is a random variable uniformly distributed over the set $[-1,-0.5]^M \cup [0.5,1]^M$. Fig. 5.1 shows a plot of the function $u(\xi)$ for the single parameter case. The density of $\boldsymbol{\xi}$ is given explicitly by

$$\rho(\xi) = 2^{M-1} \mathbf{1}_{[-1,-0.5]^M \cup [0.5,1]^M}. \tag{5.2}$$

The function u is everywhere continuous but infinitely oscillatory along each axis of $\boldsymbol{\xi}$. The axes however are not contained in the support of ρ so the oscillations do not have any effect on the statistics of u with respect to the measure on $\boldsymbol{\xi}$. Algorithm 2 with the refinement criterion used in [18] would place many collocation points near the origin in an attempt to resolve the oscillatory behavior. Provided that the approximate density $\hat{\rho}$ is a good approximation to the true density, adaptive KDE collocation will only place collocation points near the support of ρ.

In our experiments, the density estimate for each choice of M will be constructed from 5000 samples of $\boldsymbol{\xi}$ with the bandwidth h chosen by maximum likelihood cross validation. For a given value of ξ let $|(u(\xi) - \mathcal{A}_k(u)(\xi))\rho(\xi)|$ be the interpolation error scaled by ρ. First we measure the scaled interpolation error at 500 equally spaced points on $[-1.5, 1.5]$ and use the maximum observed error as an estimate for the infinity norm of the error $\|(u(\xi) - \mathcal{A}_k(u)(\xi))\rho(\xi)\|_{L_\infty(\Gamma)}$ for the one-parameter (i.e. $M = 1$ in (5.1)) problem. We denote this estimate by $\|(u(\xi) - \mathcal{A}_k(u)(\xi))\rho(\xi)\|_{l_\infty}$ Fig. 5.2 shows the interpolation error in the mesh-norm $\|\cdot\rho(\xi)\|_\infty$. This norm only indicates the error on the support of ρ. Fig. 5.2 shows that the interpolation error decays rapidly where the random variable ξ is supported. Fig. 5.2 shows that adaptive KDE collocation converges significantly faster than Algorithm 2. The reason is that Algorithm 2 places many points near the origin, attempting to resolve the oscillations. After a few initial global refinements of the grid the new method concentrates all of the new collocation points inside the support of ξ.[2] Fig. 5.3 shows the collocation nodes used by the adaptive method with KDE driven refinement.

Now we examine the performance for the same task when u depends on multiple parameters in (5.1). Fig. 5.4 shows the number of collocation points required as a function of the convergence criterion τ and the number of parameters. The figure shows that as the number of parameters is increased, the efficiency of the proposed method slows. This is due to the factor $log_2(|\Delta\theta^k|)^{3(M-1)}$ appearing in the estimate (4.3). Note however that for any fixed value of M, the asymptotic interpolation error bound (4.3) decays faster than the Monte–Carlo error bound (4.9). The results in Section 5.3 indicate that the asymptotic bound (4.3) may be pessimistic for problems of interest.

[2] Algorithm 2 with the refinement criterion (4.1) indicates that a node is not refined if $\hat{\rho}\|w_j^k\|$ is small. In practice however it is necessary to perform some initial global grid refinements to achieve a minimum level of resolution.

5.2. Example 2: two-parameter stochastic diffusion equation

Next, we use the method derived in Section 4 to compute statistics associated with the solution to the stochastic diffusion equation

$$-\nabla \cdot (a(\mathbf{x}, \xi_1, \xi_2)\nabla u(\mathbf{x}, \xi_1, \xi_2)) = 1, \quad \forall \mathbf{x} \in D \tag{5.3}$$

$$u(\mathbf{x}, \xi_1, \xi_2) = 0, \quad \forall \mathbf{x} \in \partial D \tag{5.4}$$

where $D = [0,1]^2$. The diffusion coefficient a is defined for this example as follows. Define the set $LL = \{\mathbf{x} : 0 < x_1, x_2 \leqslant 0.5\}$ and the set $UR = \{\mathbf{x} : 0.5 < x_1, x_2 < 1.0\}$. Let $\mathbb{1}_{LL}(\mathbf{x})$ and $\mathbb{1}_{UR}(\mathbf{x})$ be the indicator functions on LL and UR respectively. The diffusion coefficient is piecewise constant and is given by

$$a(\mathbf{x}, \xi_1, \xi_2) = 1 + \mathbb{1}_{LL}(\mathbf{x})\xi_1 + \mathbb{1}_{UR}(\mathbf{x})\xi_2. \tag{5.5}$$

Here ξ_1 and ξ_2 are assumed to be independently distributed log-normal random variables. The PDF of ξ_i for $i = 1, 2$ is given by

$$\rho_i(\xi_i) = \frac{1}{\xi_i\sqrt{2\pi\sigma^2}} e^{-\frac{(log(\xi_i)-\mu)^2}{2\sigma^2}}, \tag{5.6}$$

with $\sigma = 1$ and $\mu = 2$. Since ξ_1 and ξ_2 are assumed to be independent, their joint distribution is given by

$$\rho(\xi_1, \xi_2) = \frac{1}{2\pi\xi_1\xi_2} e^{\frac{-(log(\xi_1)-2)^2-(log(\xi_2)-2)^2}{2}}. \tag{5.7}$$

Note that ξ_1 and ξ_2 take on values in the range $(0, \infty)$. This, combined with the definition of the diffusion coefficient in (5.5) ensures that the diffusion coefficient will be positive at all points in D for all possible values of the random variables ξ_1 and ξ_2. This is sufficient to ensure the well-posedness of (5.3) [2]. In the numerical experiments, interpolation was carried out on the domain $[1 \times 10^{-6}, 6]^2$. This computational domain contained all of the samples of (ξ_1, ξ_2) generated by the log-normal random number generator.

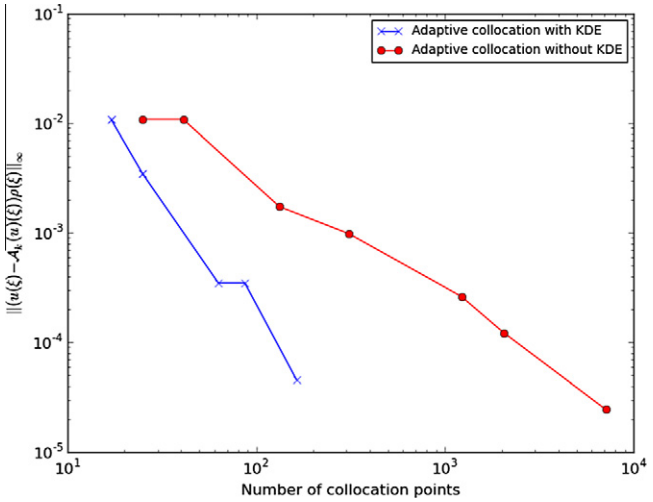

Fig. 5.2. $\|(u(\xi) - \mathcal{A}_k(u)(\xi))\rho(\xi)\|_\infty$ versus the number of collocation points.

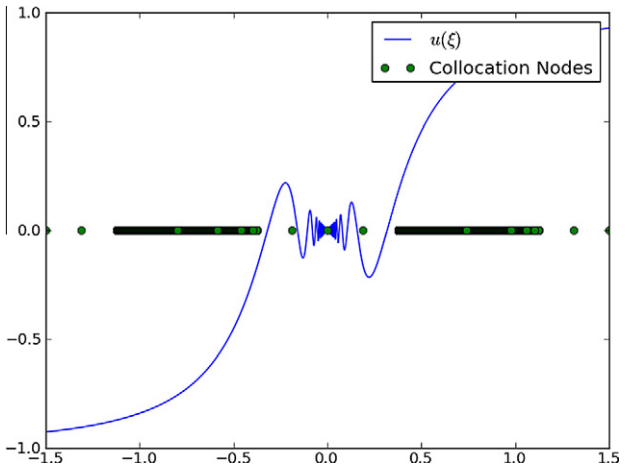

Fig. 5.3. $u(\xi)$ and the collocation points used in constructing approximate solution.

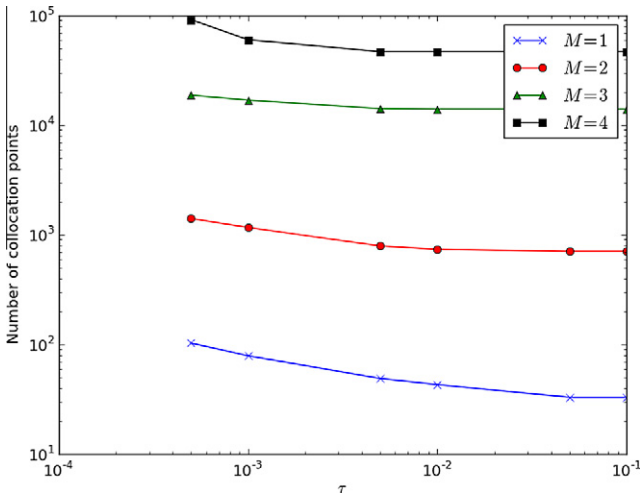

Fig. 5.4. The tolerance τ vs the number of collocation points.

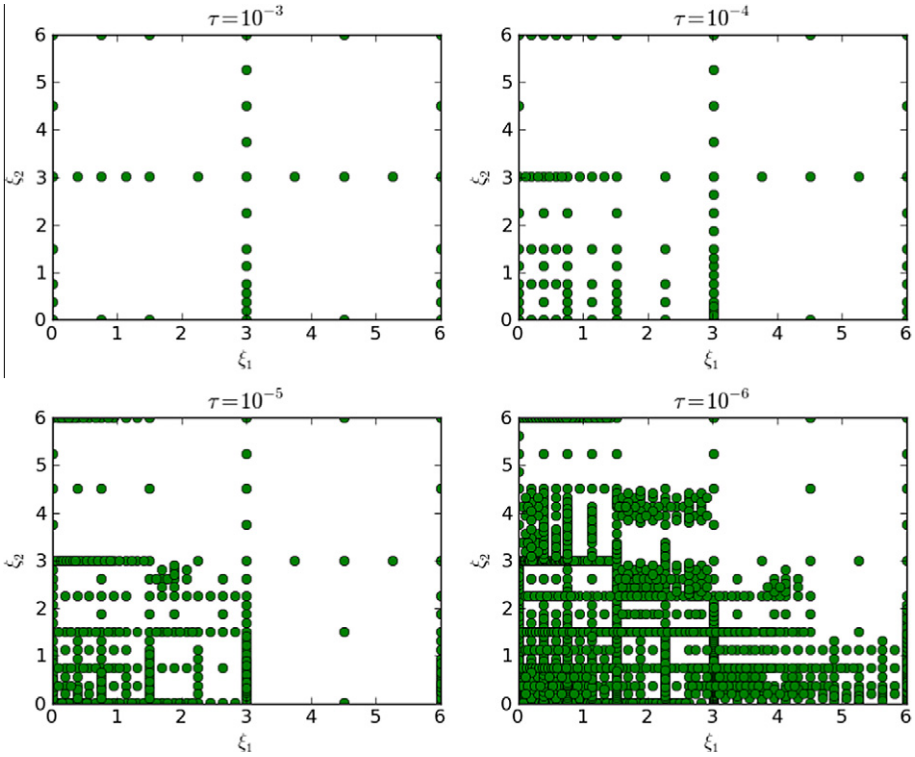

Fig. 5.5. Collocation points for various values of the error tolerance τ.

The method described above generates a set of collocation points in the stochastic space. At each of these points (5.3) must be solved by using a suitable deterministic solver. In this example the spatial discretization is accomplished using finite differences on a uniform 32×32 mesh. The discrete difference operators are formed using the five point stencil

$$\begin{bmatrix} & a(x, y+\frac{h_D}{2}, \xi_1, \xi_2) & \\ a(x-\frac{h_D}{2}, y, \xi_1, \xi_2) & a(x, y, \xi_1, \xi_2) & a(x+\frac{h_D}{2}, y, \xi_1, \xi_2) \\ & a(x, y-\frac{h_D}{2}, \xi_1, \xi_2) & \end{bmatrix}, \tag{5.8}$$

for $\mathbf{x} = [x, y]^T \in D$, and where h_D is the spatial discretization parameter. For this example the resulting linear systems are solved using a direct solver, although an iterative solver may also be used as in [9]. Although the spatial discretization of the problem introduces an additional source of error, it is known that the error resulting from the spatial discretization of the problem separates from the error associated with discretization of the stochastic component [2,3]. Thus we can focus solely on the error introduced by interpolating in the stochastic space and by approximating the true joint density by a kernel density estimate.

First we proceed as in Section 5.1 and evaluate the interpolation error. Since the exact solution is not known we compute $\mathcal{A}(u)$ with a very tight error tolerance $\tau = 10^{-9}$. We treat this as an accurate solution and observe the decay in error for interpolants obtained using a looser error tolerance. For each interpolant, the kernel density estimate is derived from 5,000 samples of $\boldsymbol{\xi} = [\xi_1, \xi_2]$ where ξ_1 and ξ_2 are independently distributed log-normal random variables as described above. The bandwidth for the kernel density estimates

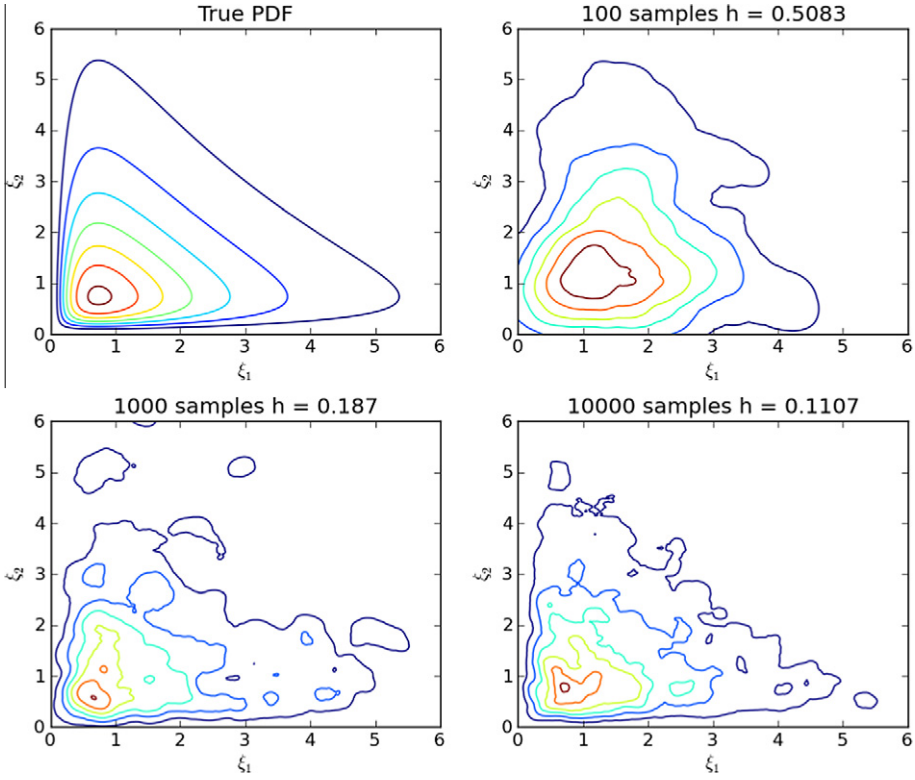

Fig. 5.6. Kernel density estimates for varying numbers of samples.

is chosen using the maximum likelihood cross-validation method described in Section 3.

Fig. 5.5 shows the collocation points used for several values of the error tolerance τ. Comparing these with the contour plot of the true joint density function in Fig. 5.6, it can be seen that the method is concentrating collocation points in regions where the estimated joint PDF is large. Thus the method is only devoting resources towards computing an accurate interpolant in regions that are significant to the statistics of u. Fig. 5.7 shows the interpolation error as a function of the number of collocation points. Since an exact solution to (5.3) is not available we treat the solution obtained by using the method with $\tau = 10^{-10}$ as an exact solution. As opposed to the first example, the solution u here depends on both the spatial location and the value of the random parameter. We report the error in the discrete norm $\|\cdot\rho\|_{l^2(D)\times l^\infty(\Gamma)}$, where the space $l^2(D)$ consists of square summable mesh-functions defined on the spatial grid and $l^\infty(\Gamma)$ consists of bounded mesh-functions defined on a 500×500 uniform grid on Γ. Fig. 5.7 shows that the interpolation error decays quickly for the two parameter problem. The apparent slowdown in convergence rate is attributable to the fact that the exact solution is not available and the error is being measured with respect to an approximate solution.

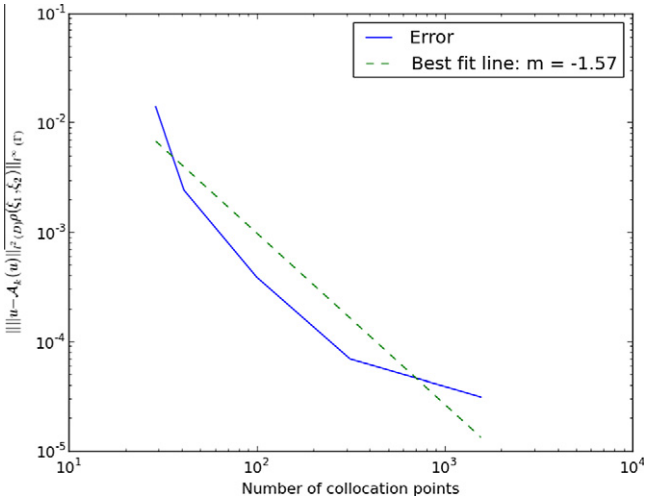

Fig. 5.7. $\|(u_h(\mathbf{x},\xi) - \mathcal{A}(u_h)(\mathbf{x},\xi))\rho(\xi)\|_{l^2(D)\times l^\infty(\Gamma)}$ versus the number of collocation points

5.3. High-dimensional stochastic diffusion

We now examine the performance of adaptive KDE collocation for evaluating the statistics of a random field that depends on a large number of parameters. The problem is given by

$$-\frac{d}{dx}(a_M(x,\xi)\frac{d}{dx}u(x,\xi)) = 1, \quad \forall x \in (0,1) \tag{5.9}$$

$$u(0,\xi) = u(1,\xi) = 0. \tag{5.10}$$

The diffusion coefficient a_M is defined for even M by

$$a_M = \mu + \sum_{k=0}^{M/2-1} \lambda_k(\xi_{2k}cos(2\pi kx) + \xi_{2k+1}sin(2\pi kx)), \tag{5.11}$$

where $\lambda_k = exp(-k), \mu = 3$ and ξ_k is uniformly distributed on $[0,1]$. The problem (5.9) is well posed on the image of ξ. The system (5.9) was solved at each collocation point by using central finite differences on a uniform mesh with 128 degrees of freedom. Experimental results for these problems are shown in Tables 5.1 (for $M = 4$ random variables), 5.2 ($M = 10$), and 5.3 ($M = 20$). The contents of the tables are as follows.

First, for each M, we performed a Monte–Carlo simulation with several choices of number of samples N. This sample size is shown in the first column of the tables. In addition, for each value of

Table 5.1
Monte–Carlo error (left) and $\|\frac{1}{N}\sum_{i=1}^{N} u_h(x,\xi^{(i)}) - \mathcal{A}(u_h)(x,\xi^{(i)})\|_{l^2(D)}$, 4 parameter problem.

N	τ				
	5×10^{-2}	1×10^{-3}	5×10^{-4}	1×10^{-4}	5×10^{-5}
100	5.25×10^{-3}	2.23×10^{-4}	1.18×10^{-4}	9.42×10^{-6}	9.42×10^{-7}
8.43×10^{-2}	**(28)**	(212)	(301)	(813)	(1169)
500	5.47×10^{-3}	2.71×10^{-4}	9.84×10^{-5}	1.12×10^{-5}	1.76×10^{-6}
3.78×10^{-2}	**(28)**	**(211)**	**(315)**	(777)	(1210)
1000	4.29×10^{-3}	2.36×10^{-4}	1.24×10^{-4}	9.78×10^{-6}	2.61×10^{-6}
2.67×10^{-2}	**(33)**	**(200)**	**(297)**	**(762)**	(1207)
5000	4.36×10^{-3}	3.88×10^{-4}	1.36×10^{-4}	1.67×10^{-5}	4.73×10^{-6}
1.19×10^{-2}	**(33)**	**(172)**	**(286)**	**(745)**	**(1104)**
20000	4.32×10^{-3}	2.73×10^{-4}	1.30×10^{-4}	1.09×10^{-5}	3.58×10^{-6}
5.96×10^{-3}	**(33)**	**(180)**	**(294)**	**(780)**	**(1107)**

Table 5.2
Monte–Carlo error (left) and $\|\frac{1}{N}\sum_{i=1}^{N} u_h(x,\xi^{(i)}) - \mathcal{A}(u_h)(x,\xi^{(i)})\|_{l^2(D)}$, 10 parameter problem.

N	τ				
	5×10^{-2}	1×10^{-3}	5×10^{-4}	1×10^{-4}	5×10^{-5}
100	7.66×10^{-3}	8.86×10^{-4}	4.41×10^{-4}	4.48×10^{-5}	8.28×10^{-6}
9.08×10^{-2}	**(76)**	(1026)	(1655)	(5026)	(8111)
500	7.13×10^{-3}	6.08×10^{-4}	3.36×10^{-4}	2.34×10^{-5}	1.01×10^{-5}
4.06×10^{-2}	**(92)**	(1170)	(1189)	(5773)	(9404)
1000	9.19×10^{-3}	6.03×10^{-4}	2.65×10^{-4}	1.95×10^{-5}	1.77×10^{-5}
2.87×10^{-2}	**(59)**	(1216)	(1989)	(5996)	(9664)
5000	7.16×10^{-3}	6.62×10^{-4}	3.03×10^{-4}	2.04×10^{-5}	1.02×10^{-5}
1.28×10^{-2}	**(93)**	**(1120)**	**(2041)**	(6095)	(9787)
20000	7.25×10^{-3}	6.27×10^{-4}	2.66×10^{-4}	1.96×10^{-5}	5.67×10^{-6}
6.42×10^{-3}	**(93)**	**(1187)**	**(2127)**	**(6050)**	**(9942)**

$M, var[u(\mathbf{x},\boldsymbol{\xi})]$ was estimated at the spatial grid points using 20,000 samples. Eq. (4.10) can then be used to compute a 95% confidence bound of the Monte–Carlo error. This estimate is shown in the first column of Tables 5.1, 5.2, and 5.3 beneath the number of samples used to construct the Monte–Carlo estimate.

The other columns of the tables contain results for adaptive KDE collocation where the kernel density estimates are generated using the same set of sample points used for the Monte–Carlo simulation. The total error for this method is bounded by (4.12). The term $\|\epsilon_{MC}\|_{l^2 D}$ is estimated by the 95% confidence bound in the first column of the tables, as discussed in the previous paragraph. The other quantities in the table are the $l^2(D)$-norm of the sample mean interpolation error, $\|\epsilon_{interp}\|_{l^2(D)}$, in the top of each box, together with (in parentheses) the number of collocation points N_{interp} used to construct $\mathcal{A}(u)$. For example, the second from left entry in the bottom row of Table 5.3 shows that for the 20-parameter problem and the 20,000 sample set, $\mathcal{A}(u)$ was constructed using 3,108 collocation points and $\|\epsilon_{interp}\|_{l^2(D)} = 6.52\times10^{-4}$.

The costs of the two methods are essentially determined by the number of PDE solves required, N for the Monte–Carlo simulation and N_{interp} for adaptive KDE collocation. In the tables, the number of collocation points N_{interp} in parentheses are shown in bold typeface when they are smaller than the number of samples. For such cases, if $\|\epsilon_{interp}\|_{l^2(D)}$ is significantly smaller than $\|\epsilon_{MC}\|_{l^2(D)}$, then adaptive KDE collocation is less expensive than Monte–Carlo simulation. It can be seen from the results that the savings can be significant when the number of samples increases. For example, the second from left entry in the bottom row of Table 5.3 shows that (by (4.12)) the error in mean for the adaptive collocation method is bounded by $\|\epsilon_{interp}\|_{l^2(D)} + \|\epsilon_{MC}\|_{l^2(D)} = 7.11\times10^{-3}$ while only

Table 5.3
Monte–Carlo error (left) and $\|\frac{1}{N}\sum_{i=1}^{N} u_h(x,\xi^{(i)}) - \mathcal{A}(u_h)(x,\xi^{(i)})\|_{l^2(D)}$, 20 parameter problem.

N	τ				
	5×10^{-2}	1×10^{-3}	5×10^{-4}	1×10^{-4}	5×10^{-5}
100	1.64×10^{-2}	1.65×10^{-3}	2.15×10^{-3}	5.81×10^{-4}	2.39×10^{-4}
9.14×10^{-2}	**(41)**	(878)	(1299)	(4126)	(6958)
500	1.45×10^{-2}	2.77×10^{-3}	1.38×10^{-3}	3.75×10^{-4}	1.67×10^{-4}
4.09×10^{-2}	**(41)**	(1045)	(1738)	(5545)	(9106)
1000	8.45×10^{-3}	1.46×10^{-3}	9.02×10^{-4}	1.66×10^{-4}	7.13×10^{-5}
2.89×10^{-2}	**(119)**	(1618)	(2622)	(8580)	(14012)
5000	8.70×10^{-3}	9.58×10^{-4}	4.99×10^{-4}	7.88×10^{-5}	2.59×10^{-5}
1.29×10^{-2}	**(156)**	**(2459)**	**(4169)**	(13389)	(22276)
20000	7.25×10^{-3}	6.52×10^{-4}	3.38×10^{-4}	3.48×10^{-5}	2.35×10^{-5}
6.46×10^{-3}	**(193)**	**(3108)**	**(4991)**	**(15963)**	(26081)

requiring 3108 PDE solves, an error comparable in magnitude to that obtained with the Monte–Carlo method (6.46×10^{-3}) with 20,000 solves.

We also note that these results suggest that the factor $log_2(|\Delta\theta^k|)^{3(M-1)}$ in the estimate (4.3) may be pessimistic for many problems of interest. Care must be taken when using the predictor method not to over-resolve the interpolant when one only has access to only a small amount of data. Doing so results in an interpolant that is too accurate given the number of samples available and results in wasted computation. This is the case in the right-hand columns of the tables where the interpolant is being resolved to a much higher level of accuracy than the associated Monte–Carlo error bound.

6. Conclusions

We have presented a new adaptive sparse grid collocation method based on the method proposed in [18] that can be used when the joint PDF of the stochastic parameters is not available and all one has access to is a finite set of samples from that distribution. It is shown that in this case a kernel density estimate can provide a mechanism for driving the refinement of an adaptive sparse grid collocation strategy. Numerical experiments show that in cases involving a large number of samples it can be economical to construct a surrogate to the unknown function using fewer function evaluations and then to perform the Monte–Carlo method on that surrogate.

Acknowledgments

The authors thank Elisabeth Ullmann for her careful reading and helpful comments during the preparation of this work.

References

[1] M. Ainsworth, J. Oden, A Posteriori Error Estimation in Finite Element Analysis, Wiley, New York, 2000.
[2] I. Babuška, F. Nobile, R. Tempone, A stochastic collocation method for elliptic partial differential equations with random input data, SIAM J. Numer. Anal. 45 (2007) 1005–1034.
[3] I. Babuška, R. Tempone, G.E. Zouraris, Galerkin finite element approximations of stochastic elliptic partial differential equations, SIAM J. Numer. Anal. 42 (2004) 800–825.
[4] A. Barth, C. Schwab, N. Zollinger, Multi-level Monte Carlo Finite Element method for elliptic PDEs with stochastic coefficients, Numer. Math. 119 (2011) 123–161.
[5] S.C. Brenner, L.R. Scott, The Mathematical Theory of Finite Element Methods, Springer-Verlag, New York, 1994.
[6] R.P. Brent, Algorithms for Minimization Without Derivatives, Prentice-Hall, New Jersey, 1973.
[7] R.E. Caflisch, Monte Carlo and quasi-Monte Carlo methods, Acta Numer. 7 (1998) 1–49.
[8] [8] K.A. Cliffe, M.B. Giles, R. Scheichl, A.L. Teckentrup, Multilevel Monte Carlo methods and applications to elliptic PDEs with random coefficients, newblock Accepted for publication in Comput. Visualizat. Sci. 14 (1) (2011) 3–15.
[9] H.C. Elman, C.W. Miller, E.T. Phipps, R.S. Tuminaro, Assessment of collocation and galerkin approaches to linear diffusion equations with random data, Int. J. Uncertainty Quantificat. 1 (2011) 19–33.
[10] J. Foo, G.E. Karniadakis, Multi-element probabilistic collocation method in high dimensions, J. Comput. Phys. 229 (2010) 1536–1557.
[11] J. Foo, X. Wan, G.E. Karniadakis, The multi-element probabilistic collocation method (ME-PCM): error analysis and applications, J. Comput. Phys. 227 (2008) 9572–9595.
[12] R. Ghanem, P. Spanos, Stochastic Finite Elements: A Spectral Approach, Springer-Verlag, New York, 1991.
[13] M. Grigoriu, Probabilistic models for stochastic partial differential equations, J. Comput. Phys. 229 (2010) 8406–8429.
[14] J. Han, M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufman, San Fransisco, 2000.
[15] A. Klimke, Uncertainty modeling using fuzzy arithmetic and sparse grids, Ph.D. Thesis, Universität Stuttgart, 2006.
[16] A. Klimke, B. Wohlmuth, Algorithm 847: spinterp: piecewise multilinear hierarchical sparse grid interpolation in MATLAB, ACM Trans. Math. Softw. 31 (2005) 561–579.
[17] M. Loeve, Probability Theory, 4th ed., 2, Springer-Verlag, New York, 1978.
[18] X. Ma, N. Zabaras, An adaptive hierarchical sparse grid collocation algorithm for the solution of stochastic differential equations, J. Comput. Phys. 228 (2009) 3084–3113.
[19] X. Ma, N. Zabaras, High-dimensional stochastic model representation technique for the solution of stochastic pdes, J. Comput. Phys. 229 (2010) 3884–3915.
[20] N. Metropolis, S. Ulam, The Monte–Carlo method, J. Amer. Statist. Assoc. 44 (1949) 335–341.
[21] F. Nobile, R. Tempone, C. Webster, An anisotropic sparse grid collocation algorithm for the solution of stochastic differential equations, SIAM J. Numer. Anal. 46 (2008) 2411–2442.
[22] F. Nobile, R. Tempone, C. Webster, A sparse grid stochastic collocation method for partial differential equations with random input data, SIAM J. Numer. Anal. 45 (2008) 2309–2345.
[23] B.W. Silverman, Density Estimation for Statistics and Data Analysis, Chapman and Hall, London, 1986.
[24] S. Smolyak, Quadrature and interpolation formulas for tensor products of certain classes of functions, Soviet Math. Dokl. 4 (1963) 240–243.
[25] X. Wan, G.E. Karniadakis, An adaptive multi-element generalized polynomial chaos method for stochastic differential equations, J. Comput. Phys. 209 (2005) 617–642.
[26] X. Wan, G.E. karniadakis, Solving elliptic problems with non-Gaussian spatially-dependent random coefficients, Comput. Methods Appl. Mech. Engrg. 198 (2009) 1985–1995.
[27] D. Xiu, J. Hesthaven, High-order collocation methods for differential equations with random inputs, SIAM J. Sci. Comput. 27 (2005) 1118–1139.