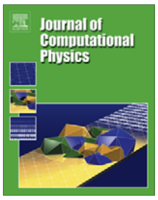
Journal of Computational Physics 228 (2009) 3084–3113

Contents lists available at ScienceDirect

Journal of Computational Physics

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



An adaptive hierarchical sparse grid collocation algorithm for the solution of stochastic differential equations

Xiang Ma, Nicholas Zabaras *

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

ARTICLE INFO

Article history:
Received 31 January 2008
Received in revised form 3 December 2008
Accepted 12 January 2009
Available online 20 January 2009

Keywords:
Collocation
Stochastic partial differential equations
Sparse grid
Hierarchical multiscale method
Adaptive sparse grid
Discontinuities
Smolyak algorithm

ABSTRACT

In recent years, there has been a growing interest in analyzing and quantifying the effects of random inputs in the solution of ordinary/partial differential equations. To this end, the spectral stochastic finite element method (SSFEM) is the most popular method due to its fast convergence rate. Recently, the stochastic sparse grid collocation method has emerged as an attractive alternative to SSFEM. It approximates the solution in the stochastic space using Lagrange polynomial interpolation. The collocation method requires only repetitive calls to an existing deterministic solver, similar to the Monte Carlo method. However, both the SSFEM and current sparse grid collocation methods utilize global polynomials in the stochastic space. Thus when there are steep gradients or finite discontinuities in the stochastic space, these methods converge very slowly or even fail to converge. In this paper, we develop an adaptive sparse grid collocation strategy using piecewise multi-linear hierarchical basis functions. Hierarchical surplus is used as an error indicator to automatically detect the discontinuity region in the stochastic space and adaptively refine the collocation points in this region. Numerical examples, especially for problems related to long-term integration and stochastic discontinuity, are presented. Comparisons with Monte Carlo and multi-element based random domain decomposition methods are also given to show the efficiency and accuracy of the proposed method.

1. Introduction

To accurately predict the performance of physical systems, it becomes essential for one to include the effects of input uncertainties into the model system and understand how they propagate and alter the final solution. The presence of uncertainties can be modeled in the system through reformulation of the governing equations as stochastic ordinary/partial differential equations (SODEs/SPDEs). The traditional approach for solving SODEs/SPDEs is the Monte Carlo (MC) method. This approach gives access to the complete statistics of the solution. It does not approximate the solution space and its convergence rate does not depend on the number of independent input random variables. Furthermore, MC methods are very easy to implement given a working deterministic code. However, the statistical approach becomes quickly intractable for complex problems in multiple random dimensions. This is because the number of realizations required to acquire good statistics is usually quite large. Furthermore, the number of realizations changes with the variance of the input parameters and the truncation errors are hard to estimate. This has in part been alleviated by improved sampling techniques like Latin hypercube sampling [1].

* Corresponding author. Tel.: +1 607 255 9104; fax: +1 607 255 1222.
E-mail address: zabaras@cornell.edu (N. Zabaras).
URL: http://mpdc.mae.cornell.edu/ (N. Zabaras).

0021-9991/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2009.01.006

A more recent approach in modeling uncertainty is based on the spectral stochastic finite element method (SSFEM) [2]. In this method, we project the dependent variables of the model onto a stochastic space spanned by a set of complete orthogonal polynomials and then a Galerkin projection scheme is used to transform the original stochastic problem into a system of coupled deterministic equations. These polynomials are functions of a set of random variables $\xi(\theta)$ where θ is a realization of the random event space. In the original work of Wiener [3], Gaussian random variables were used with Hermite polynomials. Some of the early applications of SSFEM are presented in [2,4–6]. This scheme has been extended to include other random distributions leading to generalized polynomial chaos expansions (gPC) [7]. The gPC was successfully applied to model uncertainty propagation in various applications [8–10]. Error bounds and convergence studies [11] have shown that these methods exhibit fast convergence rates with increasing orders of expansions. These convergence studies assume that the solution is sufficiently smooth in the random space. Also, the computed absolute error may become unacceptably large during long-term integration. In addition, when the solution exhibits a discontinuous dependence on the input random parameters, the gPC may converge slowly or even fail to converge. This is due to the global polynomial expansion used in the gPC which cannot resolve the local discontinuity in the random space, the well-known Gibbs phenomenon which occurs in spectral decompositions of discontinuous functions.

Thus, more efficient and robust schemes are needed to address the presence of discontinuities in the solution in the random space. In [11–13], finite element basis functions were used in the random space to approximate locally the stochastic dependence of the solution. In [14], the authors have successfully applied this method to capture unstable equilibrium in natural convection. The wavelet basis expansion method was also utilized to address this problem [15,16]. The multi-element generalized polynomial chaos method (ME-gPC) was introduced to address discontinuities in the random space while preserving the convergence rate of the gPC method [17–19]. The main idea of the ME-gPC method is to decompose the space of random inputs into disjoint random elements, then employ a gPC expansion in each element. All of the above methods employ a Galerkin projection in the random space to transform the corresponding stochastic equations to a set of deterministic algebraic equations. The coupled nature of the resulting equations for the unknown coefficients in the spectral expansion makes the solution of the stochastic problem extremely complex as the number of stochastic dimensions and/or the number of expansion terms increase, the so called *curse of dimensionality*. In fact, computational complexity of the problem increases combinatorially with the number of stochastic dimensions and the number of expansion terms. In addition, it is required to develop a stochastic simulator, which is a non-trivial task especially if the underlying ODEs/PDEs have complicated non-linear terms.

There have been recent efforts to couple the fast convergence of the Galerkin methods with the decoupled nature of MC sampling, the so called stochastic collocation method. This framework represents the stochastic solution as a polynomial approximation. This interpolant is constructed via independent function calls to the deterministic problem at different interpolation points. This strategy has emerged as a very attractive alternative to the spectral stochastic paradigm. However, the construction of the set of interpolation points is non-trivial, especially in multi-dimensional random spaces. In [20], a methodology was proposed wherein the Galerkin approximation is used to model the physical space and a collocation scheme is used to sample the random space. A tensor product rule was used to interpolate the variables in stochastic space using products of one-dimensional (1D) interpolation functions based on Gauss quadrature points. Though this scheme leads to the solution of uncoupled deterministic problems as in the MC method, the number of realizations required to build the interpolation scheme increases as power of the number of random dimensions. On the other hand, the sparse grid resulting from the Smolyak algorithm depends weakly on dimensionality [21]. Sparse grids has been applied in many fields, such as high-dimensional integration [22], interpolation [23–25] and solution of PDEs [26]. For an in depth review, the reader may refer to [27]. In [28–30], the authors used the Smolyak algorithm to build sparse grid interpolants in high-dimensional stochastic spaces based on Lagrange interpolation polynomials. Using this method, interpolation schemes can be constructed with orders of magnitude reduction in the number of sampled points to give the same level of approximation (up to a logarithmic factor) as interpolation on a uniform grid.

Error estimates for Smolyak algorithm based stochastic collocation methods have been given in [29,30], where assuming smoothness of the solution in random space they were shown to achieve fast convergence, similar to stochastic Galerkin methods. However, it is noted that some stochastic sparse grid collocation methods, e.g. [28,30], utilize the Lagrange polynomial interpolant, which is a global polynomial basis in the random space. Therefore, as is the case with gPC that uses orthogonal global polynomials, these methods fail to capture local behavior in the random space. To this end, we concentrate on stochastic collocation strategies which utilize basis functions with local support, the same idea as in [11–13,15–17], in order to resolve successfully discontinuities in the random space. In addition, we also seek for an adaptive collocation strategy which can refine the sparse grid only locally around the discontinuity region. It is noted that, for the current existing polynomial interpolation methods, e.g. [28,30], the set of interpolation points are either Clenshaw–Curtis or Gaussian quadrature points, which are pre-determined. So this leads to grids with no substantial room for adaptivity.

Therefore, an adaptive framework utilizing local interpolant/basis functions offers greater promise in efficiently and accurately representing high-dimensional non-smooth stochastic functions. Towards this idea, the authors in [17] proposed an adaptive version of the ME-gPC, where decay rate of local variance was used as an error indicator to adaptively split the random element into two parts along each dimension similar to the *h*-adaptive approach in the deterministic finite element method. In order to utilize the decoupled nature of the collocation algorithm, they later extended this method to the multi-element probabilistic collocation method (ME-PCM), where tensor product or sparse grid collocation is used in each random element [31]. Then the collocation solution is projected back onto the PC basis such that one can employ the same adaptive criterion as in ME-gPC. These are still dimension-dependent methods, where the number of random elements increases fast with the number of random dimensions. The same problem also exists for the Stochastic Galerkin [13] and Wiener–Haar expansion [16] methods. Thus, there is also a need for an adaptive framework that scales linearly ($O(N)$) with increasing dimensionality instead of the ($O(2^N)$) scaling of current adaptive stochastic methods, where N is the stochastic dimension. In this paper, we utilize a piecewise multi-linear hierarchical basis sparse grid interpolation approach towards adaptivity that addresses the issues of locality and curse-of-dimensionality. This borrows ideas directly from wavelet-based representation of functions [26,32–35], where the coefficients of the representation are used as error indicators. However, in [32–35], the multi-dimensional interpolation grid is constructed through tensor product of one-dimensional wavelet expansions and therefore it is not suitable for high dimensions. The method introduced in this work is different from the adaptive wavelet method since it employs a different adaptation strategy based on the Smolyak algorithm for constructing the interpolation grid. The basic idea here is to use a piecewise linear hat function as a hierarchical basis function by dilation and translation on equidistant interpolation nodes. Then the stochastic function can be represented by a linear combination of these basis functions. The corresponding coefficients are just the hierarchical increments between two successive interpolation levels (hierarchical surpluses) [24,27]. The magnitude of the hierarchical surplus reflects the local regularity of the function. For a smooth function, this value decreases to zero quickly with increasing interpolation level. On the other hand, for a non-smooth function, a singularity is indicated by the magnitude of the hierarchical surplus. The larger this magnitude is, the stronger the singularity. Thus, the hierarchical surplus serves as a natural error indicator for the sparse grid interpolation. When this value is larger than a predefined threshold, we simply add the $2N$ neighboring points to the current point. A key motivation towards using this framework is its linear scaling with dimensionality, in contrast to the N-dimensional tree (2^N) scaling of the h-type adaptive framework (e.g. the framework in [17]). In addition, such a framework guarantees that a user-defined error threshold is met. We will also show that it is rather easier with this approach to extract realizations, higher-order statistics, and the probability density function (PDF) of the solution.

It is noted here that, in previous works, there exists the so called dimension-adaptive (anisotropic) sparse grid methods employing the concept of generalized sparse grids, which was originally developed in [36] and further extended to interpolation in [25]. In recent papers [37,38], the authors have applied this method to various stochastic problems. In this framework, the structure of the solution was detected on-the-fly to sample the space in a non-isotropic way. The most sensitive dimension is detected and adaptively sampled. Then *all* of the interpolation points from the next level are added along this dimension. Error bounds and convergence issues for the anisotropic sparse grid collocation technique are discussed in [38]. However, this framework requires the underlying discontinuity aligned along the lines of the underlying sparse grid, which is not the case in most problems. The method introduced in this paper is different from the above adaptive strategy. We only add locally around the current point the $2N$ neighboring points from the next interpolation level instead of all of the interpolation points along only one dimension. In this way, besides the detection of important dimensions, additional singularities and local variations in a stochastic function can be found and resolved [26,27,39]. It is also noted that the work in [38] uses Lagrange polynomial interpolation and thus cannot resolve discontinuities.

The contribution of this work is as follows: (1) We utilize the concepts of hierarchical sparse grid collocation. This provides a new point of view on the sparse grid collocation method leading to the concept of adaptivity; (2) We develop a locally refined adaptive sparse grid collocation method with $2N$ linear scaling for the refinement, which further reduces the *curse of dimensionality*; (3) By purely based on the interpolation, it is shown that this method not only can calculate easily the mean and the variance, but also can extract the realization of the solution as a function of the random variables in order to examine its local behavior. This is another issue not addressed in earlier works [28–31,37,38].

This paper is organized as follows: In the next section, the mathematical framework of SODEs/SPDEs is formulated. In Section 3, the conventional sparse grid collocation (CSGC) and adaptive sparse grid collocation (ASGC) methodologies are detailed. The numerical examples are given in Section 4. Finally, concluding remarks are provided in Section 5.

2. Problem definition

In this section, we follow the notation in [28,30]. Define a complete probability space $(\Omega, \mathcal{F}, \mathcal{P})$ with sample space Ω which corresponds to the outcomes of some experiments, $\mathcal{F} \subset 2^{\Omega}$ is the σ-algebra of subsets in Ω (these subsets are called events) and $\mathcal{P} : \mathcal{F} \to [0,1]$ is the probability measure. Also, define D as a d-dimensional bounded domain $D \subset \mathbb{R}^d$ $(d = 1,2,3)$ with boundary ∂D. We are interested to find a stochastic function $u : \Omega \times D \to \mathbb{R}$ such that for $\mathcal{P}$-almost everywhere (a.e.) $\omega \in \Omega$, the following equation holds:

$$\mathcal{L}(\boldsymbol{x}, \omega; u) = f(\boldsymbol{x}, \omega), \quad \forall \boldsymbol{x} \in D \tag{1}$$

and

$$\mathcal{B}(\boldsymbol{x}; u) = g(\boldsymbol{x}), \quad \forall \boldsymbol{x} \in \partial D, \tag{2}$$

where $\boldsymbol{x} = (x_1, \ldots, x_d)$ are the coordinates in $\mathbb{R}^d$, $\mathcal{L}$ is (linear/non-linear) differential operator, and $\mathcal{B}$ is a boundary operator. In the most general case, the operators $\mathcal{L}$ and $\mathcal{B}$ as well as the driving terms f and g, can be assumed random. We assume that the boundary has sufficient regularity and that f and g are properly defined such that the problem in Eqs. (1) and (2) is well-posed $\mathcal{P}$ – a.e. $\omega \in \Omega$.

2.1. The finite-dimensional noise assumption and the Karhunen–Loève expansion

Any second-order stochastic process can be represented as a random variable at each spatial and temporal location. Therefore, we require an infinite number of random variables to completely characterize a stochastic process. This poses a numerical challenge in modeling uncertainty in physical quantities that have spatio-temporal variations, hence necessitating the need for a reduced-order representation (i.e., reducing the infinite-dimensional probability space to a finite-dimensional one). Such a procedure, commonly known as a 'finite-dimensional noise assumption' [20,28], can be achieved through any truncated spectral expansion of the stochastic process in the probability space. One such choice is the Karhunen–Loève (K–L) expansion [2].

For example, let the force term $f(\boldsymbol{x}, \omega)$ be a second-order stochastic process, and its correlation function be $R(\boldsymbol{x}_1, \boldsymbol{x}_2)$, where $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are spatial coordinates. By definition, the correlation function is real, symmetric, and positive definite. All its eigenfunctions are mutually orthonormal and form a complete set spanning the function space to which $f(\boldsymbol{x}, \omega)$ belongs. Then the truncated K–L expansion takes the following form:

$$f(\boldsymbol{x},\omega) = \mathbb{E}[f](\boldsymbol{x}) + \sum_{i=1}^{N}\sqrt{\lambda_i}\phi_i(\boldsymbol{x})Y_i(\omega), \tag{3}$$

where $\{Y_i(\omega)\}_{i=1}^{N}$ are uncorrelated random variables. If the process is a Gaussian process, then they are standard identically independent $N(0, 1)$ Gaussian random variables. Also, $\phi_i(\boldsymbol{x})$ and λ_i are the eigenfunctions and eigenvalues of the correlation function, respectively. They are the solutions of the following eigenvalue problem:

$$\int_D R(\boldsymbol{x}_1,\boldsymbol{x}_2)\phi_i(\boldsymbol{x}_2)d\boldsymbol{x}_2 = \lambda_i\phi_i(\boldsymbol{x}_1). \tag{4}$$

The number of terms needed to approximate a stochastic process depends on the decay rate of the eigenvalues. Generally, a higher correlation length would lead to a rapid decay of the eigenvalues.

Following a decomposition such as the K–L expansion, the random inputs can be characterized by a set of N random variables, e.g.

$$\begin{aligned}\mathcal{L}(\boldsymbol{x},\omega;u) &= \mathcal{L}(\boldsymbol{x},Y_1(\omega),\ldots,Y_N(\omega);u),\\ f(\boldsymbol{x},\omega) &= f(\boldsymbol{x},Y_1(\omega),\ldots,Y_N(\omega)).\end{aligned} \tag{5}$$

Hence, by using the Doob–Dynkin lemma [40], the solution of Eqs. (1) and (2) can be described by the same set of random variables $\{Y_i(\omega)\}_{i=1}^{N}$, i.e.,

$$u(\boldsymbol{x},\omega) = u(\boldsymbol{x},Y_1(\omega),\ldots,Y_N(\omega)). \tag{6}$$

Thus, the use of the spectral expansion guarantees that the finite-dimensional noise assumption is satisfied and effectively reduces the infinite probability space to a N-dimensional space.

When using the K–L expansion, we here assume that we obtain a set of mutually independent random variables. The issue of non-independent random variables can be resolved by introducing an auxiliary density function [20]. In this work, we assume that $\{Y_i(\omega)\}_{i=1}^{N}$ are independent random variables with probability density function ρ_j. Let Γ_i be the image of Y_i. Then

$$\rho(\mathbf{Y}) = \prod_{i=1}^{N}\rho_i(Y_i), \quad \forall\mathbf{Y}\in\Gamma \tag{7}$$

is the joint probability density of $\mathbf{Y} = (Y_1,\ldots,Y_N)$ with support

$$\Gamma \equiv \prod_{i=1}^{N}\Gamma_i \in \mathbb{R}^N. \tag{8}$$

Then the problem in Eqs. (1) and (2) can be restated as: Find the stochastic function $u : \Gamma \times D \to \mathbb{R}$ such that

$$\mathcal{L}(\boldsymbol{x},\mathbf{Y};u) = f(\boldsymbol{x},\mathbf{Y}), \quad (\boldsymbol{x},\mathbf{Y})\in D\times\Gamma \tag{9}$$

subject to the corresponding boundary conditions

$$\mathcal{B}(\boldsymbol{x},\mathbf{Y};u) = g(\boldsymbol{x},\mathbf{Y}), (\boldsymbol{x},\mathbf{Y})\in\partial D\times\Gamma. \tag{10}$$

We emphasize here that the dimensionality N of the space Γ is usually determined by the number of the independent random variables Y_i, for example from the K–L expansion in Eq. (3). In addition, we also assume without loss of generality that the support of the random variables Y_i is $\Gamma^i = [0, 1]$ for $i = 1,\ldots,N$ and thus the bounded stochastic space is a N-hypercube $\Gamma = [0, 1]^N$, since any bounded stochastic space can always be mapped to the above hypercube.

Therefore, the original infinite-dimensional stochastic problem is restated as a finite-dimensional problem. Then we can apply any stochastic method (gPC expansion or stochastic collocation) in the random space and the resulting equations become a set of deterministic equations in the physical space that can be solved by any standard deterministic discretization technique, e.g. the finite element method. The theory and properties of the gPC expansion have been well documented in various references [2,7,9]. The regularity of the solution with respect to the stochastic space Γ affects the convergence rate of such stochastic techniques, especially of approximations of global support. However, such regularity is usually not known *a priori* in many problems. This has motivated the development of the multi-element based stochastic domain decomposition method. The basic idea is to divide the stochastic space into several disjoint elements and solve each subproblem using gPC or conventional probabilistic collocation method, which results in the ME-gPC and ME-PCM methods, respectively. The adaptive procedure for these methods can be found in [17,18,31]. Local variance is used as an indicator for adaptivity. A parameter θ_1 is defined to identify which element is to be refined and another parameter θ_2 is set to determine in which dimension the element is going to split. These problems are addressed in this work using the adaptive stochastic sparse grid collocation method which is introduced in the following section.

3. Stochastic collocation method

The basic idea of this method is to have a finite element approximation for the spatial domain and approximate the multi-dimensional stochastic space using interpolation functions on a set of collocation points $\{\mathbf{Y}_i\}_{i=1}^{M} \in \Gamma$. Suppose we can find a finite element approximate solution u of the deterministic solution of the problem in Eqs. (9) and (10) for each point $\mathbf{Y}_i$, we are then interested in constructing an interpolant of u by using linear combinations of the solutions $u(\cdot, \mathbf{Y}_i)$. The multi-dimensional interpolation can be constructed through either full-tensor product of 1D interpolation rule or by the so called sparse grid interpolation method based on the Smolyak algorithm [21]. Since in the full-tensor product case the number of support points grows very quickly as the number of stochastic dimensions increases, we will mainly focus on the sparse grid method and discuss the proposed adaptivity algorithm.

3.1. Smolyak algorithm

The Smolyak algorithm provides a way to construct interpolation functions based on a minimal number of points in multi-dimensional space. Using the Smolyak method, univariate interpolation formulae are extended to the multivariate case by using tensor products in a special way. This provides an interpolation strategy with potentially orders of magnitude reduction in the number of support nodes required. The algorithm provides a linear combination of tensor products chosen in such a way that the interpolation error is nearly the same as for full-tensor product in higher dimensions.

Let us consider a smooth function $f : [0,1]^N \rightarrow \mathbb{R}$. In the 1D case ($N = 1$), we consider the following interpolation formula to approximate f:

$$\mathcal{U}^i(f) = \sum_{j=1}^{m_i} f(Y_j^i) \cdot a_j^i \tag{11}$$

with the set of support nodes

$$X^i = \{Y_j^i | Y_j^i \in [0,1] \quad \text{for } j = 1, 2, \ldots, m_i\}, \tag{12}$$

where $i \in \mathbb{N}, a_j^i \equiv a_j(Y_j^i) \in C([0,1])$ are the interpolation nodal basis functions, and m_i is the number of elements of the set X^i. We assume that a sequence of formulae Eq. (11) is given with different i. In the multivariate case ($N > 1$), the tensor product formulae are

$$(\mathcal{U}^{i_1} \otimes \cdots \otimes \mathcal{U}^{i_N})(f) = \sum_{j_1=1}^{m_1} \cdots \sum_{j_N=1}^{m_N} f(Y_{j_1}^{i_1}, \ldots, Y_{j_N}^{i_N}) \cdot (a_{j_1}^{i_1} \otimes \cdots \otimes a_{j_N}^{i_N}), \tag{13}$$

which serve as building blocks for the Smolyak algorithm.

The Smolyak algorithm constructs the sparse interpolant $A_{q,N}$ using products of 1D functions. $A_{q,N}$ is given as [23–25]

$$\mathcal{A}_{q,N}(f) = \sum_{q-N+1 \leqslant |\mathbf{i}| \leqslant q} (-1)^{q-|\mathbf{i}|} \cdot \binom{N-1}{q-|\mathbf{i}|} \cdot (\mathcal{U}^{i_1} \otimes \cdots \otimes \mathcal{U}^{i_N}) \tag{14}$$

with $q \geqslant N, \mathcal{A}_{N-1,N} = 0$ and where the multi-index $\mathbf{i} = (i_1, \ldots, i_N) \in \mathbb{N}^N$ and $|\mathbf{i}| = i_1 + \cdots + i_N$. Here i_k, $k = 1, \ldots, N$, is the level of interpolation along the kth direction. The Smolyak algorithm builds the interpolation function by adding a combination of 1D functions of order i_k with the constraint that the sum total ($|\mathbf{i}| = i_1 + \cdots + i_N$) across all dimensions is between $q - N + 1$ and q. The structure of the algorithm becomes clearer when one considers the incremental interpolant, Δ^i given by [23–25]

$$\mathcal{U}^0 = 0, \quad \Delta^i = \mathcal{U}^i - \mathcal{U}^{i-1}. \tag{15}$$

The Smolyak interpolation $\mathcal{A}_{q,N}$ is then given by

$$\mathcal{A}_{q,N}(f) = \sum_{|\mathbf{i}| \leqslant q} (\Delta^{i_1} \otimes \cdots \otimes \Delta^{i_N})(f) = \mathcal{A}_{q-1,N}(f) + \sum_{|\mathbf{i}|=q} (\Delta^{i_1} \otimes \cdots \otimes \Delta^{i_N})(f). \tag{16}$$

To compute the interpolant $\mathcal{A}_{q,N}(f)$ from scratch, one needs to compute the function at the nodes covered by the sparse grid $\mathcal{H}_{q,N}$

$$\mathcal{H}_{q,N} = \bigcup_{q-N+1\leqslant|\mathbf{i}|\leqslant q} (X^{i_1} \times \cdots \times X^{i_N}). \tag{17}$$

The construction of the algorithm allows one to utilize all the previous results generated to improve the interpolation (this is immediately obvious from Eq. (16)). By choosing appropriate points for interpolating the 1D function, one can ensure that the sets of points X^i are nested ($X^i \subset X^{i+1}$). To extend the interpolation from level $i-1$ to i, one only has to evaluate the function at the grid points that are unique to X^i, that is, at $X^i_\Delta = X^i \setminus X^{i-1}$. Thus, to go from an order $q-1$ interpolation to an order q interpolation in N dimensions, one only needs to evaluate the function at the differential nodes $\Delta\mathcal{H}_{q,N}$ given by

$$\Delta\mathcal{H}_{q,N} = \bigcup_{|\mathbf{i}|=q} (X^{i_1}_\Delta \otimes \cdots \otimes X^{i_N}_\Delta). \tag{18}$$

3.2. Choice of collocation points and the nodal basis functions

It is more advantageous to choose the collocation points in a nested fashion to obtain many recurring points with increasing q. One of the choice is the Clenshaw–Curtis grid at the non-equidistant extrema of the Chebyshev polynomials [28,30,38]. For any choice of $m_i > 1$, the sets $X^i = \{Y^i_1, \ldots, Y^i_{m_i}\}$ are given by

$$m_i = \begin{cases} 1, & \text{if } i = 1, \\ 2^{i-1}+1, & \text{if } i > 1, \end{cases} \tag{19}$$

$$Y^i_j = \begin{cases} (-\cos(\pi(j-1)/(m_i-1))+1)/2, & \text{for } j = 1,\ldots,m_i, \quad \text{if } m_i > 1, \\ 0.5, & \text{for } j = 1, \quad \text{if } m_i = 1. \end{cases} \tag{20}$$

With this selection, the resulting sets are nested, i.e., $\mathcal{H}_{q,N} \subset \mathcal{H}_{q+1,N}$. The corresponding univariate nodal basis functions are Lagrange characteristic polynomials.

$$a^i_j = \begin{cases} 1, & \text{for } i = 1, \quad \text{and} \\ \prod\limits_{\substack{k=1 \\ k\neq j}}^{m_i} \frac{Y - Y^i_k}{Y^i_j - Y^i_k}, & \text{for } i > 1 \quad \text{and} \quad j = 1,\ldots,m_i. \end{cases} \tag{21}$$

It is noted that by using this grid, the support nodes are pre-determined as in Eq. (20). Thus, this grid is not suitable if we want to use adaptivity. Therefore, we propose to use the Newton–Cotes grid using equidistant support nodes. By using equidistant nodes, it is easy to refine the grid locally. However, it is well known that for Lagrange polynomial interpolation on equidistant nodes, the error may not go to zero as the number of nodes increases due to the well-known Runge's phenomenon [25]. To this end, we propose to use the linear hat function as the univariate nodal basis function [22]. The piecewise linear hat function has a local support in contrast to the global support of the polynomial in Eq. (21), so it can be used to resolve discontinuities in the stochastic space.

We first consider the 1D interpolation rule Eq. (11) with the support nodes defined as

$$m_i = \begin{cases} 1, & \text{if } i = 1, \\ 2^{i-1}+1, & \text{if } i > 1, \end{cases} \tag{22}$$

$$Y^i_j = \begin{cases} \frac{j-1}{m_i-1}, & \text{for } j = 1,\ldots,m_i, \quad \text{if } m_i > 1, \\ 0.5, & \text{for } j = 1, \quad \text{if } m_i = 1. \end{cases} \tag{23}$$

It is noted that the resulting grid points are also nested and the grid has the same number of points as the Clenshaw–Curtis grid.

In the linear setting, the simplest choice of 1D basis function is the standard linear hat function [24,26,27]:

$$a(Y) = \begin{cases} 1 - |Y|, & \text{if } Y \in [-1,1], \\ 0, & \text{otherwise}. \end{cases} \tag{24}$$

This mother of all piecewise linear basis functions can be used to generate an arbitrary a^i_j with local support $[Y^i_j - 2^{1-i}, Y^i_j + 2^{1-i}]$ by dilation and translation, i.e.,

$$a^1_1 = 1 \quad \text{for } i = 1, \quad \text{and} \tag{25}$$

$$a^i_j = \begin{cases} 1 - (m_i - 1)\cdot|Y - Y^i_j|, & \text{if } |Y - Y^i_j| < 1/(m_i - 1), \\ 0, & \text{otherwise}, \end{cases} \tag{26}$$

for $i > 1$ and $j = 1,\ldots,m_i$. The N-dimensional multi-linear basis functions can be constructed using tensor products as follows:

$$a^{\mathbf{i}}_{\mathbf{j}}(\mathbf{Y}) := a^{i_1}_{j_1} \otimes \cdots \otimes a^{i_N}_{j_N} = \prod_{k=1}^{N} a^{i_k}_{j_k}, \tag{27}$$

where the multi-index $\mathbf{j} = (j_1, \ldots, j_N) \in \mathbb{N}^N$ and j_k, $k = 1, \ldots, N$, denotes the location of a given support node in the kth dimension from Eq. (23). Thus, we define the functional space

$$V^{\mathbf{i}} := \mathrm{span}\{a^{\mathbf{i}}_{\mathbf{j}} : \mathbf{i} \in \mathbb{N}^N, \mathbf{j} \in \mathbb{N}^N, j_k = 1, \ldots, m_{i_k}, k = 1, \ldots, N\} \tag{28}$$

as the space of piecewise multi-linear functions for a give multi-index $\mathbf{i}$. Then the family of functions $\{a^{\mathbf{i}}_{\mathbf{j}}\}$ is just the standard *nodal basis* of the finite-dimensional space $V^{\mathbf{i}}$. If we want to apply the Smolyak algorithm using nodal basis, it is straightforward to use Eq. (14). It can be rewritten as

$$\mathcal{A}_{q,N}(f) = \sum_{q-N+1 \leqslant |\mathbf{i}| \leqslant q} (-1)^{q-|\mathbf{i}|} \cdot \binom{N-1}{q-|\mathbf{i}|} \cdot \sum_{\mathbf{j}} f(Y^{i_1}_{j_1}, \ldots, Y^{i_N}_{j_N}) \cdot a^{\mathbf{i}}_{\mathbf{j}}. \tag{29}$$

Now we can define the sparse grid interpolation space V_Γ as

$$V_\Gamma := \bigoplus_{q-N+1 \leqslant |\mathbf{i}| \leqslant q} V^{\mathbf{i}}. \tag{30}$$

It is noted that the coefficients of the approximation in the stochastic space mainly depend on the function values at the interpolation points. Thus they do not give much information about the regularity of the solution in the random space. Therefore, the interpolation formulae provided above are not appropriate for an adaptive implementation.

3.3. From nodal basis to multivariate hierarchical basis

Let us return to the incremental interpolation formula Eq. (16). This formula takes advantage of the nested nature of the grid points, $X^{(i)} \subset X^{i+1}$ [24]. Here, we follow closely [24] to provide a clear development of the derivation of the hierarchical basis and the hierarchical surpluses.

We start from the 1D interpolation formula Eq. (11) using nodal basis as discussed in the previous section. By the definition of Eq. (15), we can write

$$\Delta^i(f) = \mathcal{U}^i(f) - \mathcal{U}^{i-1}(f). \tag{31}$$

With

$$\mathcal{U}^i(f) = \sum_{Y^i_j \in X^i} a^i_j \cdot f(Y^i_j), \quad \text{and} \quad \mathcal{U}^{i-1}(f) = \mathcal{U}^i(\mathcal{U}^{i-1}(f)), \tag{32}$$

we obtain [24]

$$\Delta^i(f) = \sum_{Y^i_j \in X^i} a^i_j \cdot f(Y^i_j) - \sum_{Y^i_j \in X^i} a^i_j \cdot \mathcal{U}^{i-1}(f)(Y^i_j) = \sum_{Y^i_j \in X^i} a^i_j \cdot (f(Y^i_j) - \mathcal{U}^{i-1}(f)(Y^i_j)), \tag{33}$$

and, since $f(Y^i_j) - \mathcal{U}^{i-1}(f)(Y^i_j) = 0, \forall Y^i_j \in X^{i-1}$, we obtain

$$\Delta^i(f) = \sum_{Y^i_j \in X^i_\Delta} a^i_j \cdot (f(Y^i_j) - \mathcal{U}^{i-1}(f)(Y^i_j)) \tag{34}$$

recalling that $X^i_\Delta = X^i \setminus X^{i-1}$. Clearly, X^i_Δ has $m^i_\Delta = m_i - m_{i-1}$ points, since $X_{i-1} \subset X_i$. By consecutively numbering the elements in X^i_Δ, and denoting the jth point of X^i_Δ as Y^i_j, we can re-write the above equation as [24]

$$\Delta^i(f) = \sum_{j=1}^{m^i_\Delta} a^i_j \cdot \underbrace{(f(Y^i_j) - \mathcal{U}^{i-1}(f)(Y^i_j))}_{w^i_j}. \tag{35}$$

Here, we define w^i_j as the 1D hierarchical surplus, which is just the difference between the function values at the current and the previous interpolation levels. We also define the set of functions a^i_j as the hierarchical basis functions. Fig. 1 shows the comparison of the nodal and the hierarchical basis functions [24]. Fig. 2 shows a comparison of the nodal and hierarchical interpolation in 1D [24].

For example, in Fig. 2, if we work in the nodal basis of interpolation level 3, then the function f is approximated as from Eq. (11)

$$f = f(Y^3_1)a^3_1 + f(Y^3_2)a^3_2 + f(Y^3_3)a^3_3 + f(Y^3_4)a^3_4 + f(Y^3_5)a^3_5. \tag{36}$$

On the other hand, the hierarchical basis for the same interpolation level from Eq. (35) is given as follows:

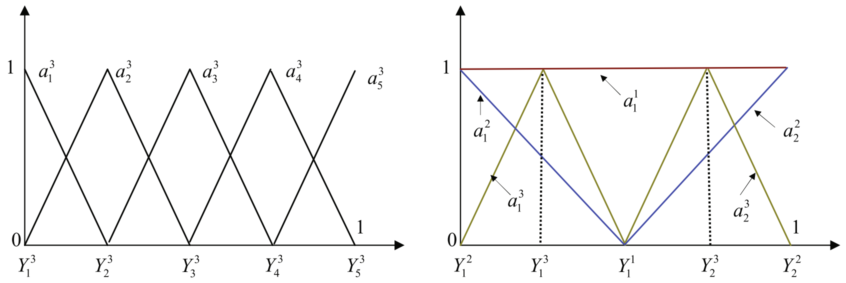


Fig. 1. Nodal basis functions $a_j^3, Y_j^3 \in X^3$ (left) and hierarchical basis functions a_j^i with the support nodes $Y_j^i \in X_{\Delta}^i, i = 1, 2, 3$ (right) for the Newton–Cotes grid.

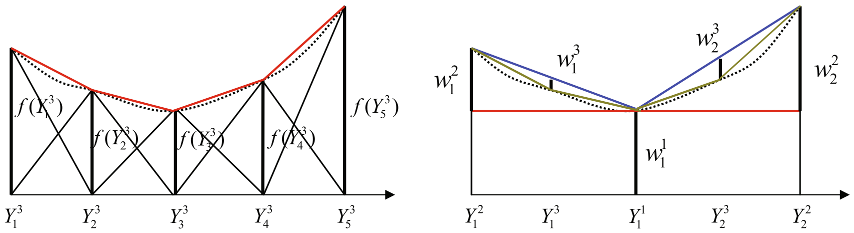


Fig. 2. Nodal (left) versus hierarchical (right) interpolation in 1D.

$$f = w_1^1 a_1^1 + w_1^2 a_1^2 + w_2^2 a_2^2 + w_1^3 a_1^3 + w_2^3 a_2^3. \tag{37}$$

Please note the different numbering used for the hierarchical and nodal basis functions. It is seen that the hierarchical basis utilizes only some of the nodal basis functions form level 1 to 3 instead of all of the nodal basis functions in level 3. It is for this reason that we refer to this representation as the 'hierarchical basis'.

For the multi-dimensional case, we define a hierarchical difference space

$$W^{\mathbf{i}} := V^{\mathbf{i}} \setminus \bigoplus_{k=1}^{N} V^{\mathbf{i}-\mathbf{e}_k}, \tag{38}$$

where $\mathbf{e}_k$ denotes the kth unit vector. To complete this definition, we formally set

$$V^{\mathbf{i}-\mathbf{e}_k} = 0, \quad \text{if } i_k = 0. \tag{39}$$

Thus, through a new multi-index set

$$B_{\mathbf{i}} := \{\mathbf{j} \in \mathbb{N}^N : Y^{i_k}_{j_k} \in X^{i_k}_{\Delta} \text{ for } j_k = 1, \ldots, m^{i_k}_{\Delta}, k = 1, \ldots, N\}, \tag{40}$$

we can obtain another basis of $V^{\mathbf{i}}$, the *hierarchical basis*

$$\{a^{\mathbf{i}}_{\mathbf{j}} : \mathbf{j} \in B_{\mathbf{k}}, \mathbf{k} \leqslant \mathbf{i}\}, \tag{41}$$

which also leads to

$$W^{\mathbf{i}} := \mathrm{span}\{a^{\mathbf{i}}_{\mathbf{j}} : \mathbf{j} \in B_{\mathbf{i}}\}. \tag{42}$$

It is clear that the following decomposition holds [26,27]:

$$V^{\mathbf{i}} := \bigoplus_{k_1=1}^{i_1} \cdots \bigoplus_{k_N=1}^{i_N} W^{\mathbf{k}} := \bigoplus_{\mathbf{k} \leqslant \mathbf{i}} W^{\mathbf{k}}. \tag{43}$$

This equation provides another view on the nodal basis function space $V^{\mathbf{i}}$. Note that in Eqs. (41) and (43), '$\leqslant$' refers to the element-wise relation for multi-indexes.

We next obtain the sparse grid interpolation formula for the multivariate case in a hierarchical form. From Eq. (16), we can write

$$\mathcal{A}_{q,N}(f) = \mathcal{A}_{q-1,N}(f) + \Delta\mathcal{A}_{q,N}(f), \tag{44}$$

$$\Delta\mathcal{A}_{q,N}(f) = \sum_{|\mathbf{i}|=q} (\Delta^{i_1} \otimes \cdots \otimes \Delta^{i_N}) \tag{45}$$

with $\mathcal{A}_{N-1,N} = 0$. This can be further simplified as

$$\mathcal{A}_{q-1,N}(f) = \sum_{|\mathbf{i}|\leqslant q-1} (\Delta^{i_1} \otimes \cdots \otimes \Delta^{i_N}) \tag{46}$$

and

$$\Delta\mathcal{A}_{q,N}(f) = \sum_{|\mathbf{i}|=q} \sum_{\mathbf{j}\in B_{\mathbf{i}}} (a^{i_1}_{j_1} \otimes \cdots \otimes a^{i_N}_{j_N}) \cdot (f(Y^{i_1}_{j_1}, \ldots, Y^{i_N}_{j_N}) - \mathcal{A}_{q-1,N}(f)(Y^{i_1}_{j_1}, \ldots, Y^{i_N}_{j_N})). \tag{47}$$

Here, we define

$$w^{\mathbf{i}}_{\mathbf{j}} = f(Y^{i_1}_{j_1}, \ldots, Y^{i_N}_{j_N}) - \mathcal{A}_{|\mathbf{i}|-1,N}(f)(Y^{i_1}_{j_1}, \ldots, Y^{i_N}_{j_N}) \tag{48}$$

as the hierarchical surplus, which is just the difference between the function value at a point in the current level of interpolation and the corresponding value at the previous interpolation level [24,26]. Eq. (47) actually provides a hierarchical subspace splitting of V_Γ

$$V_\Gamma := \underset{|\mathbf{i}|\leqslant q}{\oplus} W^{\mathbf{i}}. \tag{49}$$

Thus, we can work either in the nodal basis functional space or the hierarchical basis space. For smooth functions, the hierarchical surpluses tend to zero as the interpolation level tends to infinity as shown in Fig. 2. On the other hand, for non-smooth functions, steep gradients/finite discontinuities are indicated by the magnitude of the hierarchical surplus. The bigger the magnitude is, the stronger the underlying discontinuity is. Therefore, the hierarchical surplus is a natural candidate for error control and implementation of adaptivity.

3.4. Interpolation error

As a matter of notation, the interpolation function used will be denoted $A_{N+k,N}$, where k is called the level of the Smolyak interpolation. This is because we always start the construction from the N-dimensional multi-index $\mathbf{i} = (1, \ldots, 1)$. We consider the interpolation error in the space

$$F_N := \{f : [0,1]^N \to \mathbb{R}, D^{|\mathbf{m}|}f \text{ continues}, m_i \leqslant 2, \forall i\}, \tag{50}$$

where $\mathbf{m} \in \mathbb{N}^N_0$ and $D^{|\mathbf{m}|}$ is the usual N-variate partial derivative of order $|\mathbf{m}|$:

$$D^{|\mathbf{m}|} = \frac{\partial^{|\mathbf{m}|}}{\partial Y_1^{m_1} \cdots Y_N^{m_N}}. \tag{51}$$

Then the interpolation error in the maximum norm is given by [23–25]

$$\|f - A_{q,N}(f)\|_\infty = \mathcal{O}(M^{-2}|\log_2 M|^{3(N-1)}), \tag{52}$$

where $M = dim(\mathcal{H}(q,N))$ is the number of interpolation points.

3.5. From hierarchical interpolation to hierarchical integration

Any function $u \in \Gamma$ can now be approximated by the following reduced form from Eq. (47):

$$u(\boldsymbol{x}, \mathbf{Y}) = \sum_{|\mathbf{i}|\leqslant q} \sum_{\mathbf{j}\in B_{\mathbf{i}}} w^{\mathbf{i}}_{\mathbf{j}}(\boldsymbol{x}) \cdot a^{\mathbf{i}}_{\mathbf{j}}(\mathbf{Y}). \tag{53}$$

This expression can be considered as an approximate solution of the problem in Eqs. (9) and (10). It is just a simple weighted sum of the value of the basis functions for all collocation points in the sparse grid. Therefore, we can easily extract the useful statistics of the solution from it. For example, we can sample independently N times from the uniform distribution [0, 1] to obtain one random vector $\mathbf{Y}$, then we can place this vector into the above expression to obtain one realization of the solution. In this way, it is easy to plot realizations of the solution as well as its PDF. On the other hand, if the Smolyak algorithm Eq. (14) is used based on the cubature rule [29], although it is easy to calculate the mean and variance, it is difficult to extract the value of the solution at a particular point in the random space. This is one of the advantages of applying the stochastic collocation method based on the present interpolation rule, which allows us to obtain a visualization of the solution dependence on the random variables. After obtaining the expression in Eq. (53), it is also easy to extract the mean and variance analytically, leaving only the interpolation error.

The mean of the random solution can be evaluated as follows:

$$\mathbb{E}[u(\boldsymbol{x})] = \sum_{|\mathbf{i}|\leqslant q} \sum_{\mathbf{j}\in B_{\mathbf{i}}} w^{\mathbf{i}}_{\mathbf{j}}(\boldsymbol{x}) \cdot \int_\Gamma a^{\mathbf{i}}_{\mathbf{j}}(\mathbf{Y}) d\mathbf{Y}, \tag{54}$$

where the probability density function $\rho(\mathbf{Y})$ is 1 since the stochastic space is a unit hypercube $[0, 1]^N$. The 1D integral can be evaluated analytically:

$$\int_0^1 a_j^i(Y)dY = \begin{cases} 1, & \text{if } i = 1, \\ \frac{1}{4}, & \text{if } i = 2, \\ 2^{1-i}, & \text{otherwise}. \end{cases} \tag{55}$$

This is independent of the location of the interpolation point and only depends on the interpolation level in each stochastic dimension due to the translation and dilation of the basis functions. Since the random variables are assumed independent of each other, the value of the multi-dimensional integral is simply the product of the 1D integrals. Denoting $\int_\Gamma a_{\mathbf{j}}^{\mathbf{i}}(\mathbf{Y})d\mathbf{Y} = I_{\mathbf{j}}^{\mathbf{i}}$, we can re-write Eq. (54) as

$$\mathbb{E}[u(\boldsymbol{x})] = \sum_{|\mathbf{i}|\leqslant q} \sum_{\mathbf{j}\in B_{\mathbf{i}}} w_{\mathbf{j}}^{\mathbf{i}}(\boldsymbol{x}) \cdot I_{\mathbf{j}}^{\mathbf{i}}. \tag{56}$$

Thus, the mean is just an arithmetic sum of the product of the hierarchical surpluses and the integral weights at each interpolation point.

To obtain the variance of the solution, we need to first obtain an approximate expression for u^2, i.e.,

$$u^2(\boldsymbol{x}, \mathbf{Y}) = \sum_{|\mathbf{i}|\leqslant q} \sum_{\mathbf{j}\in B_{\mathbf{i}}} v_{\mathbf{j}}^{\mathbf{i}}(\boldsymbol{x}) \cdot a_{\mathbf{j}}^{\mathbf{i}}(\mathbf{Y}). \tag{57}$$

Then the variance of the solution can be computed as

$$\mathrm{Var}[u(\boldsymbol{x})] = \mathbb{E}[u^2(\boldsymbol{x})] - (\mathbb{E}[u(\boldsymbol{x})])^2 = \sum_{|\mathbf{i}|\leqslant q} \sum_{\mathbf{j}\in B_{\mathbf{i}}} v_{\mathbf{j}}^{\mathbf{i}}(\boldsymbol{x}) \cdot I_{\mathbf{j}}^{\mathbf{i}} - \left(\sum_{|\mathbf{i}|\leqslant q} \sum_{\mathbf{j}\in B_{\mathbf{i}}} w_{\mathbf{j}}^{\mathbf{i}}(\boldsymbol{x}) \cdot I_{\mathbf{j}}^{\mathbf{i}} \right)^2. \tag{58}$$

3.6. Adaptive sparse grid interpolation

As discussed in Section 3.3, the magnitudes of the hierarchical surpluses decay to zero quickly as the level of interpolation increases assuming a smooth function in the stochastic space. If the smoothness condition is not fulfilled, an adaptive sparse grid is preferred, which for example may place more points around the discontinuity region and less point in the region of smooth variation. One way to perform adaptation and refinement is on the level of the hierarchical subspaces $W^{\mathbf{i}}$ in Eq. (42). This leads to the so called dimension-adaptive (anisotropic) sparse grids [36–38]. This approach detects important dimensions and places all the collocation points from the hierarchical subspace $W^{\mathbf{i}+\mathbf{e}_j}$ along the important dimension j. Thus, this method is not suitable if we want to look at the local behavior of the stochastic function. Alternatively, the adaptation and refinement process can be performed on the level of the single hierarchical basis functions $a_{\mathbf{j}}^{\mathbf{i}}$ from Eq. (41). We then obtain a method which, besides the detection of important dimensions, identifies and resolves singularities and local non-smooth variations in the stochastic function [26,27,39]. In this section, we focus on the latter method and develop an adaptive sparse grid stochastic collocation algorithm based on the error control of the hierarchical surpluses.

Before discussing the algorithm, let us first introduce some notation. The 1D equidistant points of the sparse grid in Eq. (23) can be considered as a tree-like data structure as shown in Fig. 3. It is noted that special treatment is needed here going from level 2 to level 3. For the nodes 0 and 1 in level 2, we only add one point along the dimension (there is only one son here instead of two sons as is the case for all other subsequent levels of interpolation). Then, we can consider the interpolation level of a grid point Y as the depth of the tree $D(Y)$. For example, the level of point 0.25 is 3. Denote the father of a grid point as $F(Y)$, where the father of the root 0.5 is itself, i.e., $F(0.5) = 0.5$.

Thus, the conventional sparse grid in the N-dimensional random space can be reconsidered as

$$\mathcal{H}_{q,N} = \left\{ \mathbf{Y} = \{Y_1, \ldots, Y_N\} | \sum_{i=1}^{N} D(Y_i) \leqslant q \right\}. \tag{59}$$

We denote the sons of a grid point $\mathbf{Y} = (Y_1, \ldots, Y_N)$ by

$$\mathrm{Sons}(\mathbf{Y}) = \{\mathbf{S} = (S_1, S_2, \ldots, S_N) | (F(S_1), S_2, \ldots, S_N) = \mathbf{Y}, \quad \text{or} \quad (S_1, F(S_2), \ldots, S_N) = \mathbf{Y}, \ldots, \ \text{or} \quad (S_1, S_2, \ldots, F(S_N)) = \mathbf{Y}\}. \tag{60}$$

From this definition, it is noted that, in general, for each grid point there are two sons in each dimension, therefore, for a grid point in a N-dimensional stochastic space, there are $2N$ sons. It is also noted that, the sons are also the *neighbor points* of the

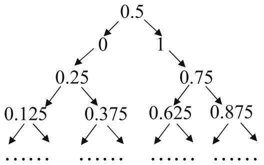


Fig. 3. 1D tree-like structure of the sparse grid.

father. Recall from the definition of grid points from Eq. (23) and the definition of hierarchical basis from Eq. (41) that the neighbor points are just the support nodes of the hierarchical basis functions in the next interpolation level. By adding the neighbor points, we actually add the support nodes from the next interpolation level, i.e., we perform interpolation from level $|\mathbf{i}|$ to level $|\mathbf{i}|+1$. Therefore, in this way, we refine the grid locally while not violating the developments of the Smolyak algorithm Eq. (47).

The basic idea here is to use hierarchical surpluses as an error indicator to detect the smoothness of the solution and refine the hierarchical basis functions $a^{\mathbf{i}}_{\mathbf{j}}$ whose magnitude of the hierarchical surplus satisfies $|w^{\mathbf{i}}_{\mathbf{j}}| \geqslant \varepsilon$. If this criterion is satisfied, we simply add the $2N$ neighbor points of the current point from Eq. (60) to the sparse grid. An example of a case with two random variables is shown in Fig. 4. It is noted that the growth of the points scales linearly with increasing dimensionality rather than the $O(2^N)$ tree-like scaling of the standard h-type adaptive refinement as in a random element-based framework, e.g. in ME-gPC.

In the Smolyak construction, we always perform the interpolation level by level. For each level, we first calculate the hierarchical surplus for each point, then we check whether the adaptive criterion $|w^{\mathbf{i}}_{\mathbf{j}}| \geqslant \varepsilon$ is satisfied. If so, we generate the $2N$ neighboring points. There is a possibility that the neighbors have already been generated by other points. Therefore, it is critical to keep track of the uniqueness of the newly generated neighboring points. We refer to these newly generated neighboring points as active points. To this end, we use the data structure ⟨set⟩ from the standard template library in C++ to store all the active points and we refer to this as the active index set. ⟨set⟩ is a kind of sorted associative container that stores unique elements (keys). When inserting a new element, this data structure will check if the new element already exists. If so, it will not insert the element. If not, the element is inserted according to the ordering of the elements in the ⟨set⟩. Due to the sorted nature of the ⟨set⟩, the searching and inserting is always very efficient. Another advantage of using this data structure is that it is easy for a parallel code implementation. Since we store all of the new points from the next level in the ⟨set⟩, we can evaluate the surplus for each point in parallel, which increases the performance significantly.

In addition, when the discontinuity is very strong, the hierarchical surpluses may decrease very slowly and the algorithm may not stop until a sufficiently high interpolation level. However, from Eq. (55), it is seen that the weights $I^{\mathbf{i}}_{\mathbf{j}}$ decrease very quickly as the level of interpolation increases. The same is true with the hierarchical surpluses. The contribution of this term to the mean and the variance may be neglected in comparison to a certain desired accuracy level of the statistics. Therefore, a maximum interpolation level is always specified as another stopping criterion. It is noted here that the definition of the level of the Smolyak interpolation for the ASGC method is the same as that of the conventional sparse grid even if not all points are included. The first hierarchical surplus is always the function value at the point $(0.5,\ldots,0.5)$. There is a possibility that the function value may be zero and thus the refinement terminates immediately. In order to avoid the early stop for the refinement process, we always refine the first level and keep a provision on the first few hierarchical surpluses [26]. Therefore, let $\varepsilon > 0$ be the parameter for the adaptive refinement threshold. We propose the following iterative refinement algorithm beginning with a coarsest adaptive sparse grid $\mathcal{G}_{N,N}$, i.e., we begin with the N-dimensional multi-index $\mathbf{i} = (1,\ldots,1)$, which is just a point $(0.5,\ldots,0.5)$.

(1) Set level of Smolyak construction $k = 0$.

(2) Construct the first level adaptive sparse grid $\mathcal{G}_{N,N}$.
- Calculate the function value at the point $(0.5,\ldots,0.5)$;
- Generate the $2N$ neighbor points and add them to the active index set;
- Set $k = k + 1$.

(3) While $k \leqslant k_{max}$ and the active index set is not empty:
- Copy the points in the active index set to an old index set and clear the active index set.
- Calculate in parallel the hierarchical surplus of each point in the old index set according to

$$w^{\mathbf{i}}_{\mathbf{j}} = f(Y^{i_1}_{j_1},\ldots,Y^{i_N}_{j_N}) - \mathcal{G}_{N+k-1,N}(f)(Y^{i_1}_{j_1},\ldots,Y^{i_N}_{j_N}). \quad (61)$$

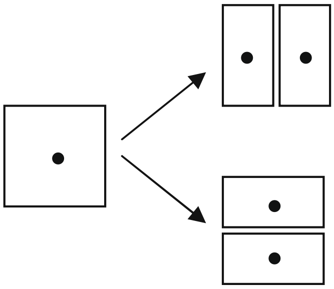


Fig. 4. An example of nodes and supports of a locally refined sparse grid in 2D random domain.

Here, we use all of the existing collocation points in the current adaptive sparse grid $\mathcal{G}_{N+k-1,N}$. This allows us to evaluate the surplus for each point from the old index set in parallel.

- For each point in the old index set, if $|w_{\mathbf{j}}^{\mathbf{i}}| \geqslant \varepsilon$
  - Generate $2N$ neighbor points of the current active point according to Eq. (60);
  - Add them to the active index set.

- Add the points in the old index set to the existing adaptive sparse grid $\mathcal{G}_{N+k-1,N}$. Now the adaptive sparse grid becomes $\mathcal{G}_{N+k,N}$.
- $k = k + 1$.

(4) Calculate the mean and the variance, the PDF and if needed realizations of the solution (see Section 3.5).

Remark 1. In practice, instead of using $w_{\mathbf{j}}^{\mathbf{i}}$, it is sometimes preferable to use $v_{\mathbf{j}}^{\mathbf{i}}$ from Eq. (57) as the error indicator. This is because the hierarchical surplus $v_{\mathbf{j}}^{\mathbf{i}}$ is related to the calculation of the variance. In principle, we can consider it like a local variance. Thus, it is more sensitive to the local variation of the stochastic function than $w_{\mathbf{j}}^{\mathbf{i}}$. In all but the first example in Section 4, the set threshold ϵ refers to the surpluses $v_{\mathbf{j}}^{\mathbf{i}}$. Recall that for accurate calculation of the variance in Eq. (58), both the function and its square are interpolated independently, see Eqs. (56) and (57), respectively.

Remark 2. The algorithm developed here is different from the original algorithm in [26]. In [26], the sparse grid introduced is based on the so called maximum-norm-based sparse grid [24,25]. It assumes that the function value vanishes on the boundary and the hierarchical surplus is calculated through a N-dimensional stencil which gives the coefficients for a linear combination of function values at the collocation points. Generally, this kind of sparse grid is not very suitable for high-dimensional stochastic spaces [24,25].

Remark 3. It is also noted that, in the adaptive sparse grid $\mathcal{G}$, we also keep some points whose surpluses are smaller than the threshold when they are generated from their fathers whose surpluses are larger than the threshold. In other words, we want to keep the adaptive sparse grid balanced, the so called balanced adaptivity, see [41]. This is different from the algorithm in [26], where all of the points whose hierarchical surplus is less than the threshold are omitted.

3.6.1. Convergence and accuracy of the adaptive collocation method

For a threshold ε and a fixed level $q = N + k$ of the sparse grid interpolation, the approximation $u_{\mathcal{H}}^{q}(\boldsymbol{x}, \mathbf{Y})$ from Eq. (53) of the conventional sparse grid method can be rewritten as a sum of two terms $u_{\mathcal{G}}^{q}$, representing the interpolation on the adaptive sparse grid $\mathcal{G}$ and $u_{<}^{q}$ that accounts for all of the missing points (see also Remark 3) whose hierarchical surpluses are below the threshold ε. Since for any piecewise N-linear basis function $a_{\mathbf{j}}^{\mathbf{i}}(\mathbf{Y})$, $\|a_{\mathbf{j}}^{\mathbf{i}}\|_{\infty} = 1$ [27], we can show that the error between the adaptive sparse grid interpolation and that of using conventional sparse grid is

$$\|u_{\mathcal{H}}^{q} - u_{\mathcal{G}}^{q}\|_{\infty} = \|u_{<}^{q}\|_{\infty} = \sum_{|\mathbf{i}| \leqslant q} \sum_{\substack{\mathbf{j} \in B_{\mathbf{i}} \\ |w_{\mathbf{j}}^{\mathbf{i}}| < \varepsilon}} |w_{\mathbf{j}}^{\mathbf{i}}(\boldsymbol{x})| \cdot \|a_{\mathbf{j}}^{\mathbf{i}}(\mathbf{Y})\|_{\infty} \leqslant \varepsilon M_2, \tag{62}$$

where M_2 is the number of all missing points. When decreasing the threshold ε, the number of missing terms M_2 also decreases (as the tolerance is reduced, more points are locally refined). Therefore, we can see that indeed the approximation of the adaptive sparse grid interpolation converges to the conventional interpolation case when decreasing the threshold ε. Accordingly, the interpolation error when using the adaptive spare grid collocation method can be approximated by

$$\|u - u_{\mathcal{G}}^{q}\|_{\infty} = \|u - u_{\mathcal{H}}^{q} + u_{\mathcal{H}}^{q} - u_{\mathcal{G}}^{q}\|_{\infty} \leqslant \|u - u_{\mathcal{H}}^{q}\|_{\infty} + \|u_{\mathcal{H}}^{q} - u_{\mathcal{G}}^{q}\|_{\infty}. \tag{63}$$

The first term in the equation above is the interpolation error of the conventional sparse grid collocation method (see Eq. (52)). The second term is the error between the conventional and adaptive sparse grid collocation methods that was shown to be of the order of $\mathcal{O}(\varepsilon)$. Numerical investigation of these errors are provided in Section 4.1.

Hereafter, for convenience, we use CSGC to denote the conventional sparse grid collocation method from Eq. (47) using multi-linear basis functions and ASGC to denote the adaptive sparse grid collocation method from the algorithm introduced in this section with the same basis functions.

4. Numerical examples

This section consists of four examples. The first example is used to demonstrate the failure of the dimension-adaptive method when the singularity is not aligned along the grid. On the second example, we compare our method with MC and the multi-element based method on a benchmark problem involving stochastic discontinuity. In the third example, we assess the ability of ASGC to detect the important dimension in a high-dimensional stochastic elliptic problem. In the last example, Rayleigh–Bénard instability is considered to showcase the use of the method in physical problems.

4.1. Approximation of function with regularized line singularity

In this section, we demonstrate the ability of the ASGC method in interpolating given functions. The computed results are compared with the CSGC method. We consider the function on $[0, 1]^2$:

$$f(x,y) = \frac{1}{|0.3 - x^2 - y^2| + \delta}, \tag{64}$$

where $\delta = 10^{-1}$. We first construct the interpolant $\mathcal{A}_{q,2}(f)$, then we randomly generate 1000 points in $[0, 1]^2$ and finally compute the error as follows:

$$e = \max_{i=1,\dots,1000} |f(\boldsymbol{x}_i) - \mathcal{A}_{q,2}(f)(\boldsymbol{x}_i)|. \tag{65}$$

The function of interest has a line singularity that is not along the grid lines, see Fig. 5. From the convergence plot with respect to ε on the left of Fig. 6, it is seen that the error converges nearly exponentially fast with respect to ε. On the right of Fig. 6, the convergence rate is shown with respect to the needed number of points for different thresholds. For example, it is noted that for threshold $\varepsilon = 10^{-3}$ more points are needed than when using the other two thresholds shown but a higher level of accuracy is obtained. Also note that much less points are needed in the ASGC than in the CSGC to achieve the same accuracy. The highest accuracy achieved for ASGC is 6.09×10^{-3}, where the interpolation level is 19 and the number of points is 16,659 as opposed to 6,029,313 points using the same level of CSGC. The evolution of the adaptive grid for threshold $\varepsilon = 10^{-3}$ is shown in Fig. 7. The line of discontinuity is automatically detected by the ASGC method.

Since the line singularity is not along any dimension, it is expected that the dimension-adaptive (anisotropic) sparse grid method [36,38] fails in this case. The results are shown in Fig. 8, where the algorithm is implemented using the MatLab Sparse Grid Interpolation Toolbox developed by Klimke [42]. From the convergence plot, it is interesting to note that the convergence rate is nearly the same as that of the CSGC method in Fig. 6. This is because the line singularity results in the same importance of both dimensions and the anisotropic method thus puts points in all dimensions. This is seen from the sparse grid in Fig. 8, where the grid is nearly the same as the full-tensor product case. Therefore, this example verifies that if the singularity is not exactly along the dimensions, the dimension-adaptive sparse grid method is not applicable and identifies the need to develop a different adaptive strategy that is working directly on the hierarchical basis as the one presented in this paper.

4.2. Kraichnan–Orszag (K–O) problem

The transformed Kraichnan–Orszag three-mode problem can be expressed as [17]

$$\begin{aligned} \frac{dy_1}{dt} &= y_1 y_3, \\ \frac{dy_2}{dt} &= -y_2 y_3, \\ \frac{dy_3}{dt} &= -y_1^2 + y_2^2 \end{aligned} \tag{66}$$

subject to initial conditions

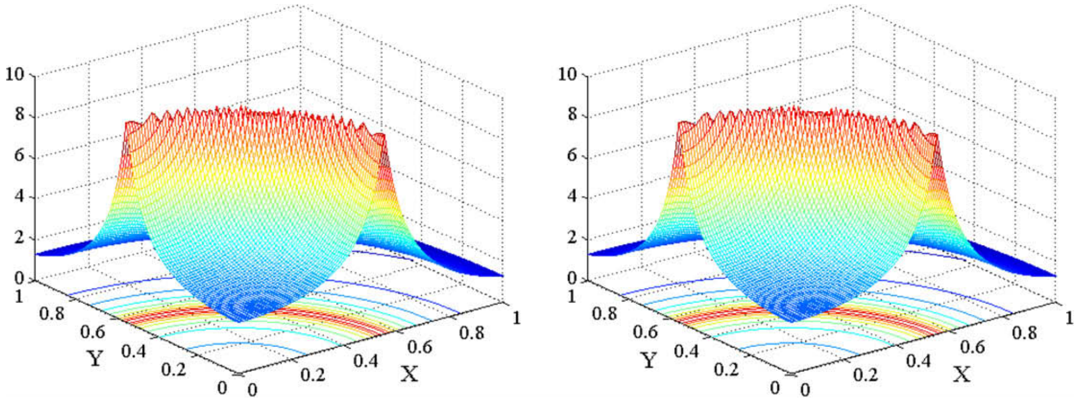


Fig. 5. Line singularity: Comparison of the exact (left) and interpolant (right) functions using the ASGC method with threshold $\varepsilon = 10^{-3}$.

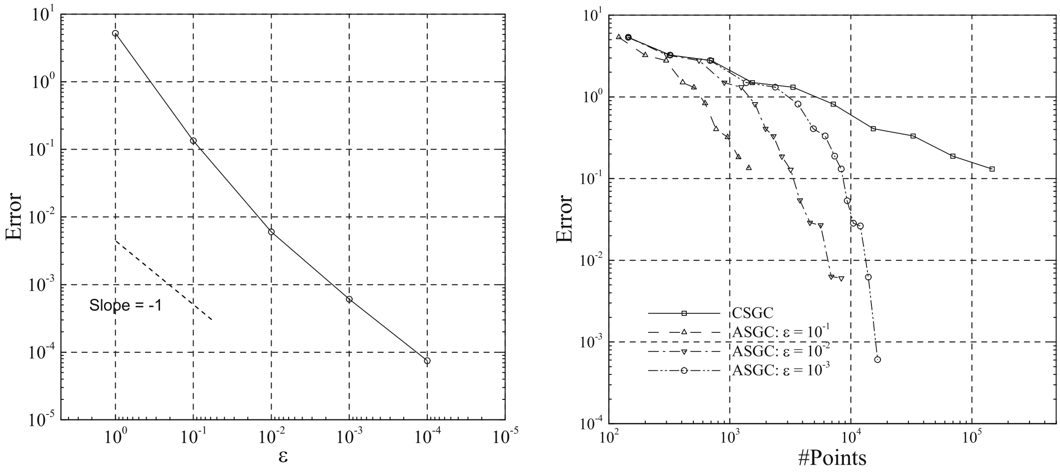
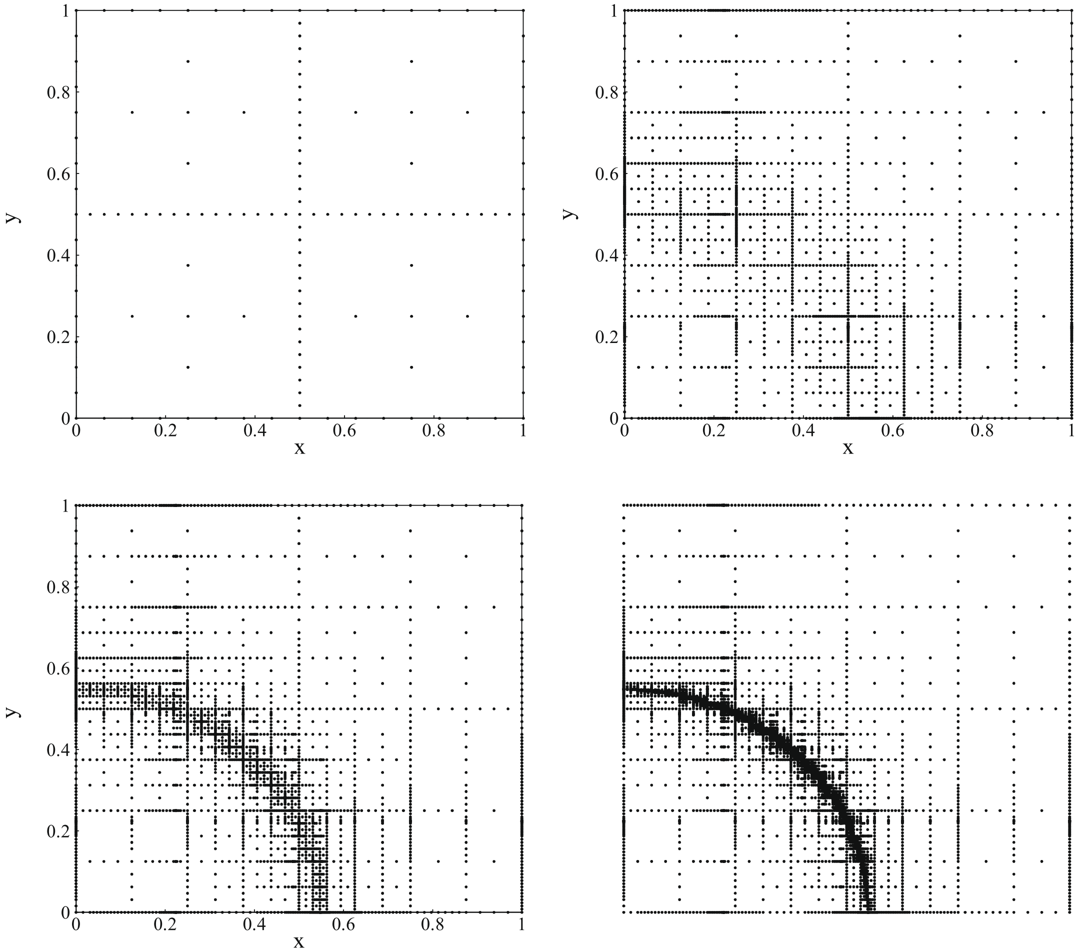


Fig. 6. Line singularity: (left) Convergence of the ASGC method with respect to the threshold ε; (right) Comparison of the interpolation error for conventional and adaptive sparse grid interpolation using different threshold ε values.



$$y_1(0) = Y_1(0;\omega), \quad y_2(0) = Y_2(0;\omega), \quad y_3(0) = Y_3(0;\omega). \tag{67}$$

This problem shows a bifurcation on the parameter $y_1(0)$ and $y_2(0)$. The deterministic solutions of the problem are periodic, and the period goes to infinity if the initial conditions are located at the planes $y_1 = 0$ and $y_2 = 0$, i.e., discontinuity occurs

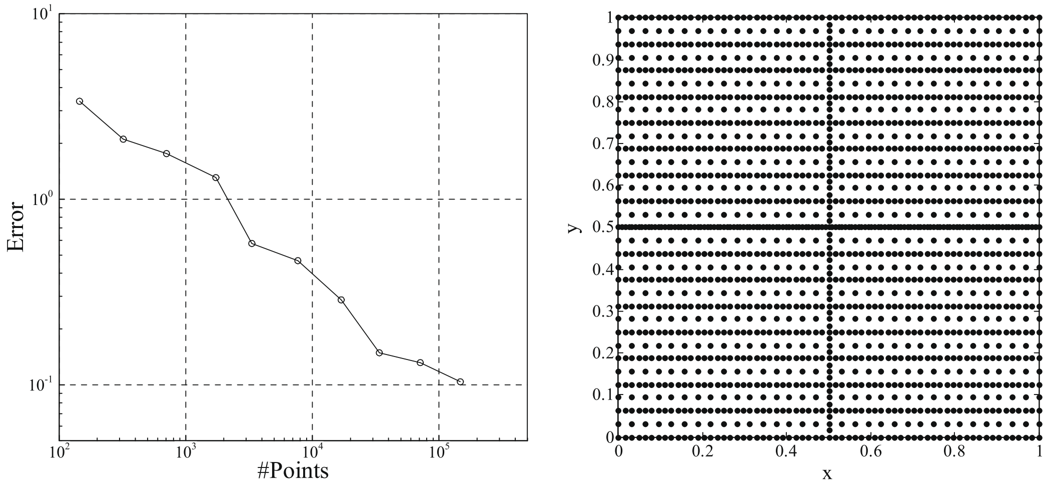


Fig. 8. Line singularity: (left) Convergence of the dimension-adaptive method; (right) Dimension-adaptive sparse grid.

when the initial conditions cross these two planes [17]. Here, we choose the random initial conditions subject to the uniform distribution $Y \sim U(-1, 1)$. In this formulation, the initial conditions cross the discontinuity plane and thus as expected the gPC method fails in computing the solution to this problem. This problem was originally solved using ME-gPC and ME-PCM in [17,18,31]. Here, we are addressing this problem using the ASGC method. The time integration of Eq. (66) is performed using a fourth-order Runge–Kutta scheme. In all computations described in this section, a time step $\Delta t = 0.01$ was used. Error convergence and comparison of computation cost with both ME-gPC and ME-PCM are conducted.

4.2.1. One-dimensional random input

At first, we study the following random initial conditions:

$$y_1(0) = 1.0, \quad y_2(0) = 0.1Y(0;\omega), \quad y_3(0) = 0. \tag{68}$$

In Fig. 9, we show the evolution of the variance within the time interval [0, 30] (short time behavior). For comparison, the results of gPC are also included. The 'exact' solution is obtained using a quasi-random Sobol (MC-SOBOL) sequence with 10^6 iterations. Due to the discontinuity, the result from MC-SOBOL is much more accurate than the standard MC simulation directly sampling from the uniform distribution. It is seen that the gPC begins to fail at time $t \approx 8$, while the ASGC method converges even with a large threshold $\varepsilon = 0.1$. From the adaptive sparse grid in Fig. 9, it is noted that even though most of the points are refined as a result of the small threshold ε, most of the refinement after level *8* occurs around the discontinuity point $Y = 0.0$. The refinement stops at level 16, which corresponds to 425 number of points, while the conventional sparse grid requires 65,537 points.

The maximum error of the variance of y_1, y_2 and y_3 at $t = 30$ is tabulated in Table 1. The maximum error of the variance is defined as $\max_{i=1,2,3}|\mathrm{Var}(y_i) - \mathrm{Var}(y_{i,MC})|$ at $t = 30$. The 'exact' solution is taken as the results given by MC-SOBOL 10^6 iterations. For each threshold ε, we show the level when the refinement stops, the corresponding number of collocation points and the error. It is seen that, with decreasing threshold, the stopping interpolation levels and the number of collocation points increase. At the same time, the accuracy becomes better and we can approximately obtain an error of the order of 0.01ε. We have also tabulated the error using MC-SOBOL and MC with the same number of samples. Although the MC-SOBOL is a quasi-MC method with better convergence than the standard MC method, from the table it can be seen that the ASGC procedure approximately leads to 2 and 1 orders of magnitude reduction in the error, as compared to MC and MC-SOBOL, respectively. We then compare the computational cost between the ASGC and multi-element based methods (Table 2). The error level is achieved by decreasing the error threshold ε in AGSC and θ_1 in both ME-gPC and ME-PCM. It is noted that we conduct the ME-gPC using a third-order expansion while we use linear basis functions in ASGC. In ME-PCM, a level 3 Clenshaw–Curtis sparse grid is used in each element. From the results, we note that for comparable accuracy, both the ASGC method and ME-PCM are much faster than the *h*-adaptive ME-gPC. Although using higher-order gPC expansion can reduce the number of random elements, the increase of the expansion terms results in more computation time. On the other hand, the computational time is nearly the same for ASGC and ME-PCM. However, it is more meaningful to compare the number of function calls since both methods are based on a collocation algorithm. Many more points are needed for the ME-PCM than the ASGC to achieve the same accuracy.

The long-term behavior of the solution is presented in Fig. 10 within the time interval [0, 100]. The corresponding realizations at different times as a function of random variable Y are given in Fig. 11. These realizations are reconstructed using hierarchial surpluses according to Eq. (53). It is seen that at earlier time, the discontinuity has not yet been developed, which explains the reason the gPC is accurate at earlier times. With increasing time, the discontinuity is growing stronger and the solution is very oscillatory. Thus, many more interpolation points are needed to successfully resolve the discontinuity.

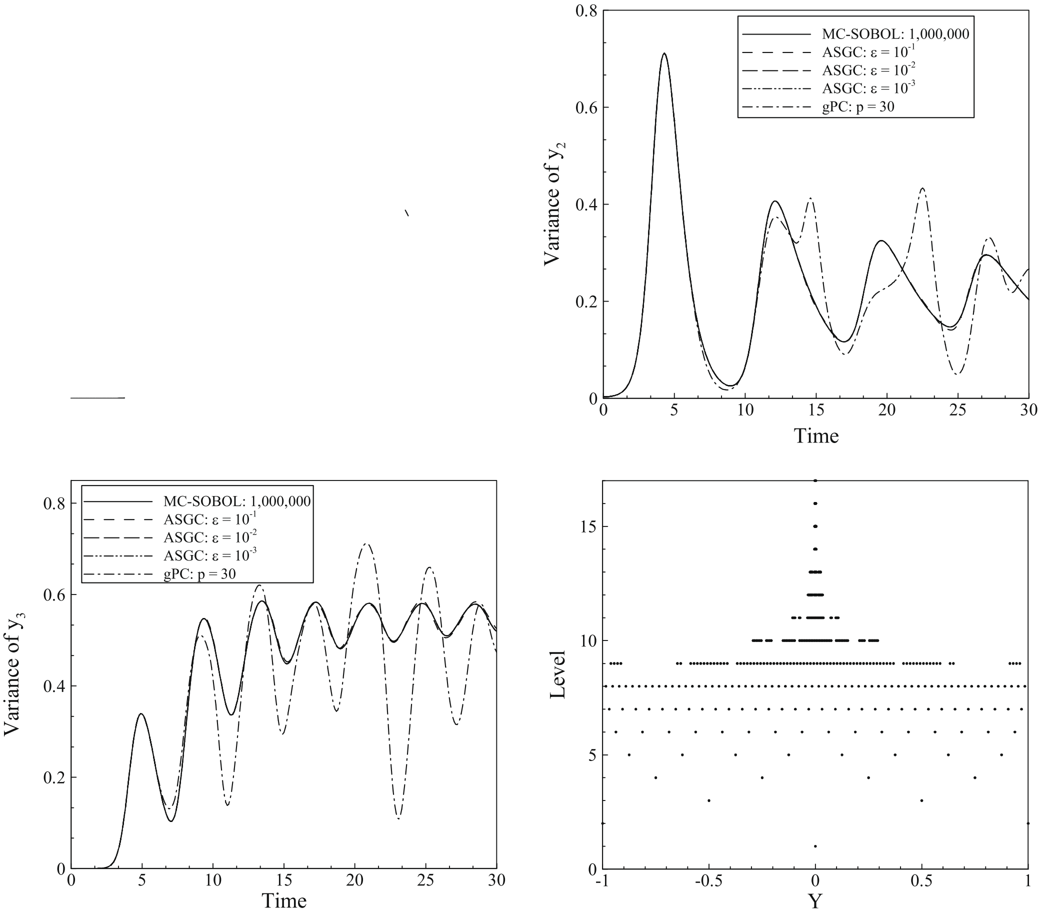


Table 2
Comparison of computational costs for the 1D K–O problem.

Error level	ASGC		*h*-Adaptive ME-gPC		*h*-Adaptive ME-PCM	
	Time (s)	#Points	Time (s)	#Elements	Time (s)	#Points
10^{-2}	0.03	21	13.75	16	0.38	126
10^{-3}	0.25	117	51.11	58	0.87	288
10^{-4}	1.70	425	82.02	92	7.12	2304
10^{-5}	13.65	1381	97.42	110	14.26	4608

4.2.2. Two-dimensional random input

In this section, we study the K–O problem with 2D random input

$$y_1(0) = 1.0, \quad y_2(0) = 0.1Y_1(0;\omega), \quad y_3(0) = Y_2(0;\omega). \tag{69}$$

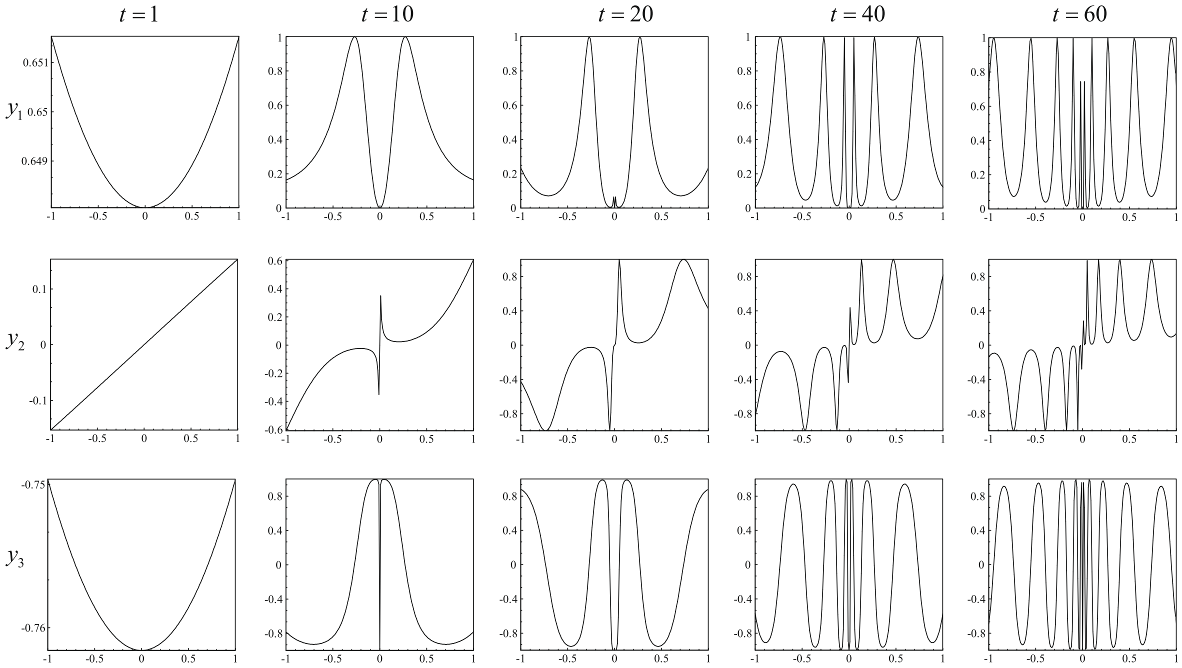


Fig. 11. Realizations of the solution (y_1, y_2, y_3) for the 1D K–O problem as a function of the random variable Y at different times.

verified from comparison with the result obtained by MC-SOBOL that unlike the previous results, here 2×10^6 iterations are needed to correctly resolve the discontinuity. Due to the symmetry of y_1 and y_2 in Eq. (66) and the corresponding random input, the variances of y_1 and y_2 are the same. Therefore, in Fig. 13 we only show the results for y_1 and y_3. The maximum interpolation level is set at 15. Finally, in order to show the implementations of h-adaptive multi-element based methods are correct, we provide the results from both methods in Fig. 14. The computation results for the ASGC, ME-gPC and ME-PCM methods are shown in Tables 5–7, respectively, where we fix $\theta_1 = 10^{-4}$ and $\theta_2 = 10^{-3}$. The error is defined as the maximum of the absolute error of the variance of y_2 and y_3 at time $t = 10$ from the 'exact' solution given by MC-SOBOL with 2×10^6 iterations. It is seen that although the ASGC method has larger error compared with MC-SOBOL and the convergence rate is not optimal, it has better accuracy than the standard MC method. In addition, it is still much faster than the ME-gPC for a comparable accuracy of the order of 10^{-4}. Due to the strong discontinuity in this problem, it took much longer time for both the ASGC and ME-PCM to arrive at the same accuracy than in the 2D problem. It is interesting to note that ME-PCM is the fastest method in this case. However, as before, to achieve the same accuracy as the ASGC, many more points are required for the ME-PCM. The advantage of ME-PCM is its p-type convergence such that the error quickly drops to the order of 10^{-6} when interpolation level increases to 8 in each element. However, the number of function evaluations is 10 times more than that of the reference solution. Therefore, the efficiency of all the methods over the MC method is not as obvious as in the previous two examples. In this extreme case, the MC-SOBOL is more favorable.

4.3. Stochastic elliptic problem

In this section, we compare the convergence rate of the CSGC and ASGC methods through a stochastic elliptic problem in two spatial dimensions. As shown in the previous examples, the adaptive sparse grid collocation method can accurately capture the non-smooth region of the stochastic space. Therefore, when the non-smooth region is along a particular dimension (i.e., one dimension is more important than others), the ASGC method is expected to identify and resolve it. In this example, we demonstrate this ability of the ASGC method to detect important dimensions when each dimension weighs unequally. This is similar to the dimension-adaptive method, especially in high stochastic dimension problems.

Here, we adopt the model problem from [30]:

$$\begin{aligned} &-\nabla \cdot (a_N(\omega,\cdot)\nabla u(\omega,\cdot)) = f_N(\omega,\cdot), \quad \text{in } D \times \Gamma \\ &u(\omega,\cdot) = 0, \quad \text{on } \partial D \times \Gamma, \end{aligned} \tag{71}$$

with the physical domain $D = \{\boldsymbol{x} = (x, y) \in [0, 1]^2\}$. To avoid introducing large errors from physical discretization, we take a deterministic smooth load $f_N(\omega, x, y) = \cos(x)\sin(y)$ with homogeneous boundary conditions. Therefore, we assume that there are no substantial errors from the physical discretization. The deterministic problem is solved using the finite element method with 900 bilinear quadrilateral elements. Furthermore, as in [30], in order to eliminate the errors associated with a

numerical Karhunen–Loève expansion solver and to keep the random diffusivity strictly positive, we construct the random diffusion coefficient $a_N(\omega, x)$ with 1D spatial dependence as

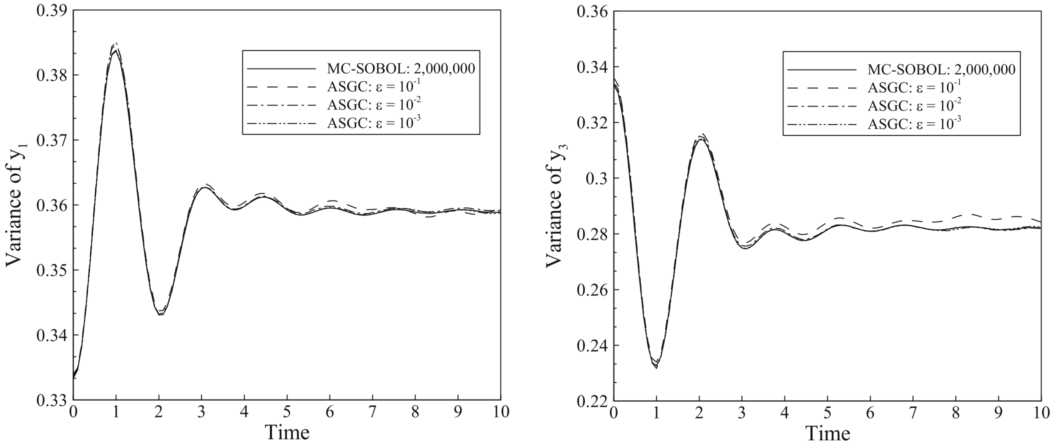


Fig. 13. Evolution of the variance of $y_1 = y_2$ (left) and y_3 (right) for the 3D K–O problem using ASGC.

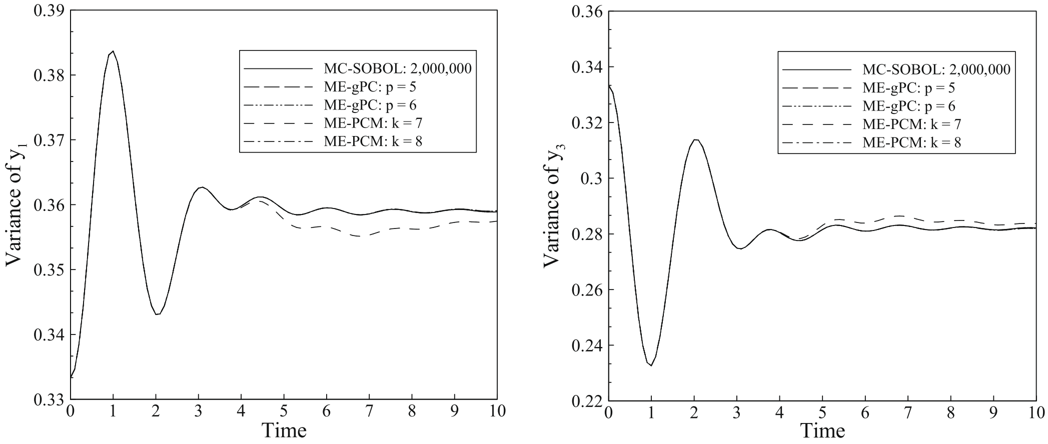


Fig. 14. Evolution of the variance of $y_1 = y_2$ (left) and y_3 (right) for the 3D K–O problem using both ME-gPC and ME-PCM. Both results are obtained with $\theta_1 = 10^{-4}$ and $\theta_2 = 10^{-3}$.

Table 5
Computational results for 3D K–O problem using ASGC.

ASGC				MC	MC-SOBOL
Threshold	#Points	Error	Time (h)	Error	Error
10^{-1}	46,953	1.95×10^{-3}	0.09	2.72×10^{-3}	6.23×10^{-5}
10^{-2}	210,177	2.95×10^{-4}	0.85	8.99×10^{-4}	6.61×10^{-5}
10^{-3}	498,025	1.58×10^{-4}	3	5.47×10^{-4}	2.74×10^{-5}

Table 6
Computational results for 3D K–O problem using *h*-adaptive ME-gPC.

order *p*	*h*-adaptive ME-gPC		
	#Elements	Error	Time (h)
3	5584	1.10×10^{-3}	10
4	3256	5.85×10^{-4}	22.5
5	2336	3.29×10^{-4}	39
6	1624	2.98×10^{-4}	108

Table 7
Computational results for 3D K–O problem using *h*-adaptive ME-PCM.

Level *k*	*h*-Adaptive ME-PCM		
	#Points	Error	Time (h)
6	590,364	4.15×10^{-3}	0.05
7	14,305,746	1.53×10^{-3}	0.48
8	15,788,608	6.78×10^{-6}	0.55

$$\log(a_N(\omega,x) - 0.5) = 1 + Y_1(\omega)\left(\frac{\sqrt{\pi}L}{2}\right)^{1/2} + \sum_{n=2}^{N} \xi_n \phi_n(x) Y_n(\omega), \tag{72}$$

where

$$\xi_n := (\sqrt{\pi}L)^{1/2} \exp\left(\frac{-(\lfloor\frac{n}{2}\rfloor\pi L)^2}{8}\right), \quad \text{if } n > 1, \tag{73}$$

and

$$\phi_n(x) := \begin{cases} \sin\left(\frac{\lfloor\frac{n}{2}\rfloor\pi x}{L_p}\right), & \text{if } n \text{ even}, \\ \cos\left(\frac{\lfloor\frac{n}{2}\rfloor\pi x}{L_p}\right), & \text{if } n \text{ odd}. \end{cases} \tag{74}$$

$Y_n(\omega)$, $n = 1,\ldots,N$, are independent uniformly distributed random variables in the interval $[-\sqrt{3}, \sqrt{3}]$. The parameter L_p in Eq. (74) can be taken as $L_p = \max\{1, 2L_c\}$ and the parameter L in Eqs. (72) and (73) is $L = L_c/L_p$. This expansion is similar to a Karhunen–Loève expansion of a 1D random field with stationary covariance

$$\mathrm{cov}[\log(a_N - 0.5)](x_1, x_2) = \exp\left(\frac{-(x_1 - x_2)^2}{L_c^2}\right). \tag{75}$$

Small values of the correlation length L_c correspond to slow decay in Eq. (72), i.e., each stochastic dimension weighs almost equally. On the other hand, large values of L_c result in fast decay rates, i.e., the first several stochastic dimensions corresponding to large eigenvalues weigh most importantly. By using the expansion Eq. (72), it is assumed that we are given an analytic stochastic input. Thus there is no truncation error. It is different from the discretization of a random field using the K–L expansion, where for different correlation lengths we keep different terms accordingly. In this example, we fix N and change L_c to adjust the importance of each stochastic dimension. In this way, we want to investigate the effect of correlation length L_c on the ability of the ASGC method to detect important dimensions.

To study the convergence of the algorithm, we consider a problem where the interpolation level increases linearly. We estimate the $L^2(D)$ approximation error for the mean and the variance. Specifically, to estimate the computation error in the qth level, we fix the dimension N and compare the results at two consecutive levels, e.g. the error for the mean is $E[\mathcal{A}_{q,N}(u_N) - \mathcal{A}_{q+1,N}(u_N)]$. Similar error is defined for the variance. The results are shown in Fig. 15 for different correlation lengths at $N = 11$. Each symbol denotes one interpolation level. To compare the convergence rate between the CSGC and ASGC methods, we choose the same maximum interpolation level for both methods. Then we decrease the threshold ε until the ASGC method arrives approximately at the same accuracy as the CSGC method. In [30], the authors proved for the stochastic elliptic problem that the convergence rate for the CSGC is nearly exponential. Since linear basis can be consider as polynomial order one, it is seen in Fig. 15 that the error for CSGC indeed decreases nearly exponentially which verifies the result in [30]. For small correlation lengths, the effects on the convergence rate for both CSGC and ASGC are nearly the same. On the other hand, for large correlation length, if we adopt the ASGC method, much less number of collocation points is required to achieve the same accuracy as the CSGC method. This is because more points are placed along the important dimensions which are associated with large eigenvalues. Therefore, the larger correlation lengths have positive effects on the rate of convergence, while decreasing L_c leads to a deterioration of the rate of convergence of the ASGC method due to equal weighting of all directions. It is also noted that, for small correlation length, we need a small threshold to achieve the desired accuracy. Smaller correlation length indicates a smoother stochastic space. The surplus also decreases very fast. Therefore, for a larger threshold, the refinement stops earlier.

Next, in Fig. 16 we study some higher-dimensional case. Due to the rapid increase in the number of collocation points, we focus on a moderate correlation length $L_c = 0.6$ so that the ASGC is effective. From this figure, it is seen that for $N \geqslant 25$, the ASGC method successfully detects the important dimensions and terminates the refinement automatically. The expansion in Eq. (72) is dominated by the first several terms which depend on the chosen correlation length. Since we choose a moderate large value of L_c, the important expansion terms associated with large eigenvalues are almost the same for the four cases considered in Fig. 16. Thus, the error level that can be achieved is nearly identical for the four cases. However, many more points are needed for increasing dimensions. Therefore, it is rather difficult to solve this problem using the CSGC method. For

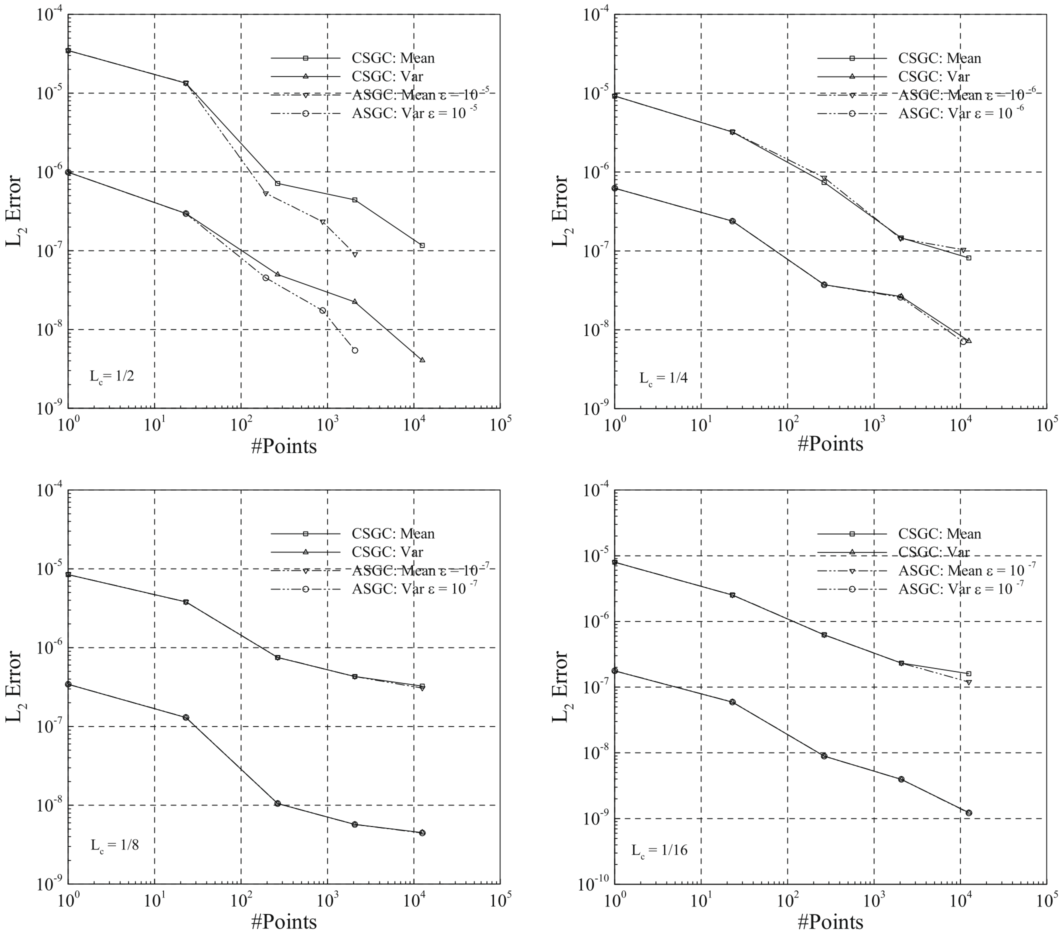


Fig. 15. The convergence of the stochastic elliptic problem in $N = 11$ dimensions for different correlation lengths $L_c =$ 1/2, 1/4, 1/8, 1/16, using both the CSGC and ASGC methods.

example, when $N = 75$ and $\varepsilon = 10^{-6}$, the refinement using the ASGC method stops at level 8 and the corresponding number of points is 276,913. On the other hand, the number of points required for the CSGC method with the same interpolation level is 3.5991×10^{11}.

In order to further verify our results, we compare the mean and the variance when $N = 75$ using the AGSC method with $\varepsilon = 10^{-6}$ with the 'exact' solution given by MC simulation with 10^6 samples. The comparison is shown in Fig. 17 over the entire physical domain. The relative error is defined as $\frac{|E(\mathcal{A}_{q,N}(u_N)) - E(u_{MC})|}{|E(u_{MC})|}$. The maximum relative error for the mean is 9.89×10^{-4}, while the maximum relative error for the variance is 5.29×10^{-3}. Therefore, the ASGC method is indeed a very accurate method comparable to the MC method even in high stochastic dimensions. The computational time is about 0.5 h, which is much less than the time needed by the MC method which took about 2 h on 10 processors.

We also compare the convergence rate with the standard MC method. $L^2(D)$ error is computed by comparing the solution with a reference solution computed from 10^6 MC samples. The results are shown in Fig. 18. As expected, when the correlation length is large, the ASGC method is most effective. With the same number of points, the error of the ASGC is nearly one-order lower than that of the MC method. When the correlation length decreases, the effect of the ASGC is nearly the same as that of the CSGC. Thus, the convergence rate becomes slower as seen in the right plot of Fig. 18, where the ASGC error is nearly one-order larger than the MC error. This is due to the performance of the CSGC method suffering from increasing number of dimensions (unlike MC method) as a result of the weak dependence on the dimensionality in the logarithmic term of the error bound as indicated in Eq. (52). To achieve a desired accuracy, we have to increase the interpolation level. However, the number of collocation points will increase excessively fast as shown in Fig. 19, and therefore the problem becomes prohibitively expensive. We also implemented the ME-PCM version of this problem that however was not appropriate for such high-dimensional problem. When we start solving this problem with only one element, due to the large local variance, the method tends to split the element in every dimension, which results in 2^{25} new elements thus exceeding computer memory. Therefore, the multi-element based domain decomposition method depends much more on the dimensionality than the ASGC does.

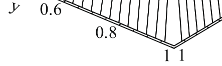
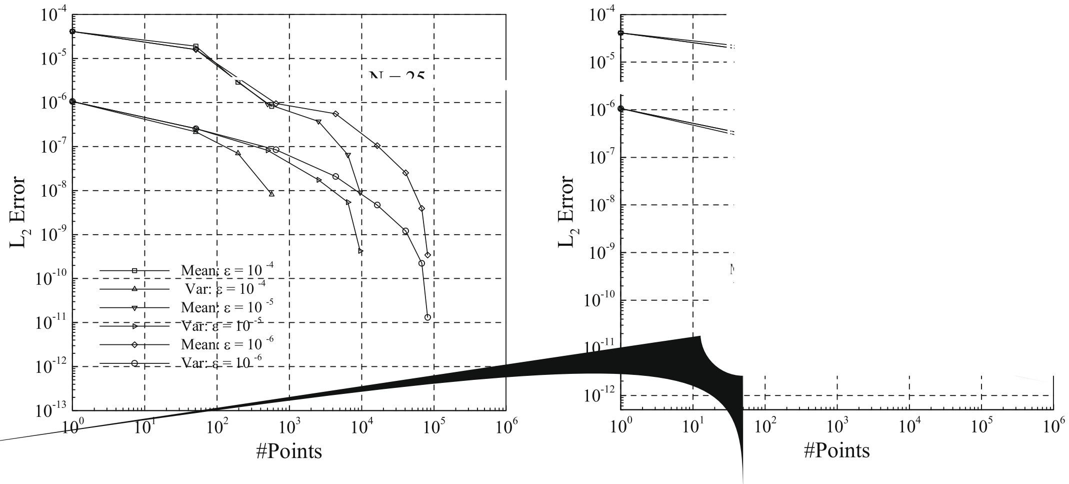


Fig. 17. Relative error of the mean (left) and varian
by MC with 10^6 iterations.



In summary, the results show that, besides indicating local variance in the solution, the ASGC method is also an efficient numerical approach to detect the important dimensions in the stochastic space. This plays the same role as in the dimensional adaptive method, in particular in problems with high stochastic dimension. However, the performance of the method depends on the smoothness of the stochastic space. If the stochastic space is rather smooth (i.e., all stochastic dimensions are equally important), the MC method is still the best choice for problems with a high number of stochastic dimensions. Finally,

where $F(\boldsymbol{u}, \theta)$ is the forcing function in the Navier–Stokes equations and Pr is the Prandtl number of the fluid. In the problems considered later, $F(\boldsymbol{u}, \theta)$ is the Bousinnessq approximated buoyant force term $-RaPr\theta\boldsymbol{g}$, where Ra is the thermal Rayleigh number and $\boldsymbol{g}$ is the gravity vector.

Here, we want to study the stochastic formulation of this problem. The physical domain is taken to be a closed cavity $[0, 1]^2$ filled with air ($Pr = 0.7$). No-slip conditions are imposed on the boundary. The vertical walls are assumed to be adiabatic. The top wall is maintained under a deterministic cold temperature $\theta_c = -0.5$. The bottom wall temperature is assumed to be a random hot temperature θ_h. The statistics of θ_h are assumed to be such that both stable and unstable modes occur with finite probability. We set $Ra = 2500$, which is larger than the critical Rayleigh number, so that convection can be initialized by varying the hot wall temperature.

Under these conditions the problem is to find stochastic functions that describe the velocity $\boldsymbol{u} \equiv \boldsymbol{u}(\boldsymbol{x}, t, \omega) : D \times [0, T] \times \Gamma \to \mathbb{R}^2$, pressure $p \equiv p(\boldsymbol{x}, t, \omega) : D \times [0, T] \times \Gamma \to \mathbb{R}$ and temperature $\theta \equiv \theta(\boldsymbol{x}, t, \omega) : D \times [0, T] \times \Gamma \to \mathbb{R}$, such that the following equations are satisfied:

$$\nabla \cdot \boldsymbol{u}(\cdot, \omega) = 0, \quad \text{in } D \times [0, T] \times \Gamma, \tag{79}$$

$$\frac{\partial \boldsymbol{u}(\cdot, \omega)}{\partial t} + \boldsymbol{u}(\cdot, \omega) \cdot \nabla \boldsymbol{u}(\cdot, \omega) = -\nabla p(\cdot, \omega) + Pr \nabla^2 \boldsymbol{u}(\cdot, \omega) + F(\boldsymbol{u}(\cdot, \omega), \theta(\cdot, \omega)), \quad \text{in } D \times [0, T] \times \Gamma, \tag{80}$$

$$\frac{\partial \theta(\cdot, \omega)}{\partial t} + \boldsymbol{u}(\cdot, \omega) \cdot \nabla \theta(\cdot, \omega) = \nabla^2 \theta(\cdot, \omega), \quad \text{in } D \times [0, T] \times \Gamma. \tag{81}$$

The deterministic governing Eqs. (76)–(78) are solved using the second-order stabilized projection finite element method developed in [43]. The spatial domain is discretized using 40 × 40 bilinear quadrilateral finite elements. Prior to stochastic simulation, several deterministic computations were performed in order to find out the range where the critical point lies in. These simulations were conducted by perturbing the hot wall temperature from the purely conductive solution. We monitor the time evolution of the average kinetic energy in the field as illustrated in Fig. 20. It is seen that after a short time, the kinetic energy exhibits growth (heat convection mode) or decay (heat conduction mode) until steady-state is arrived, depending on the hot wall temperature. Therefore, the critical temperature lies in the range [0.5, 0.55]. In addition, we also monitor the steady-state Nusselt number which denotes the rate of heat transfer:

$$Nu \equiv \frac{1}{\theta_h - \theta_c} \int_0^1 \frac{\partial \theta}{\partial y}\Big|_{y=0} dx. \tag{82}$$

Clearly, in the conductive (stable) regime, $Nu = 1$. For temperature larger than the critical value, convection is initialized and heat transfer enhancement occurs so that $Nu(\theta_h) > 1$. The results are shown in Table 8. Obviously, when the hot wall temperature is larger than 0.55, heat convection begins. This again verifies the critical hot wall temperature lies in the range [0.5, 0.55]. However, the exact critical value is not known to us. So, we now try to capture this unstable equilibrium using the ASGC method.

4.4.2. Adaptive sparse grid collocation scheme

In this section, we assume the following stochastic boundary condition for the hot wall temperature:

$$\theta_h = 0.4 + 0.3Y, \tag{83}$$

where Y is a uniform random variable in the interval [0,1]. Following the discussion above, both a stable and an unstable flow occur for this range of θ_h.

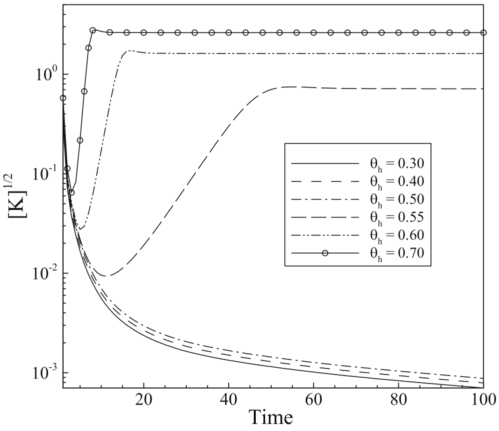


Fig. 20. Evolution of the average kinetic energy for different hot wall temperatures.

Table 8
Steady-state Nusselt number for different hot wall temperatures.

θ_h	0.30	0.40	0.50	0.55	0.60	0.70
Nu	1.00000	1.00000	1.00000	1.01496	1.07357	1.17744

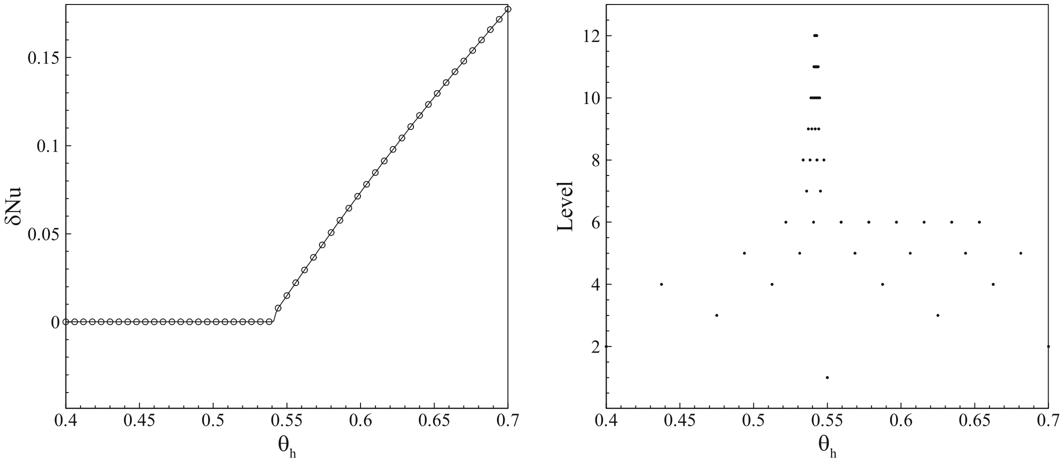


Fig. 21. Steady-state δNu (left) versus hot wall temperature using ASGC and the corresponding adaptive sparse grid with threshold $\varepsilon = 0.01$ (right).

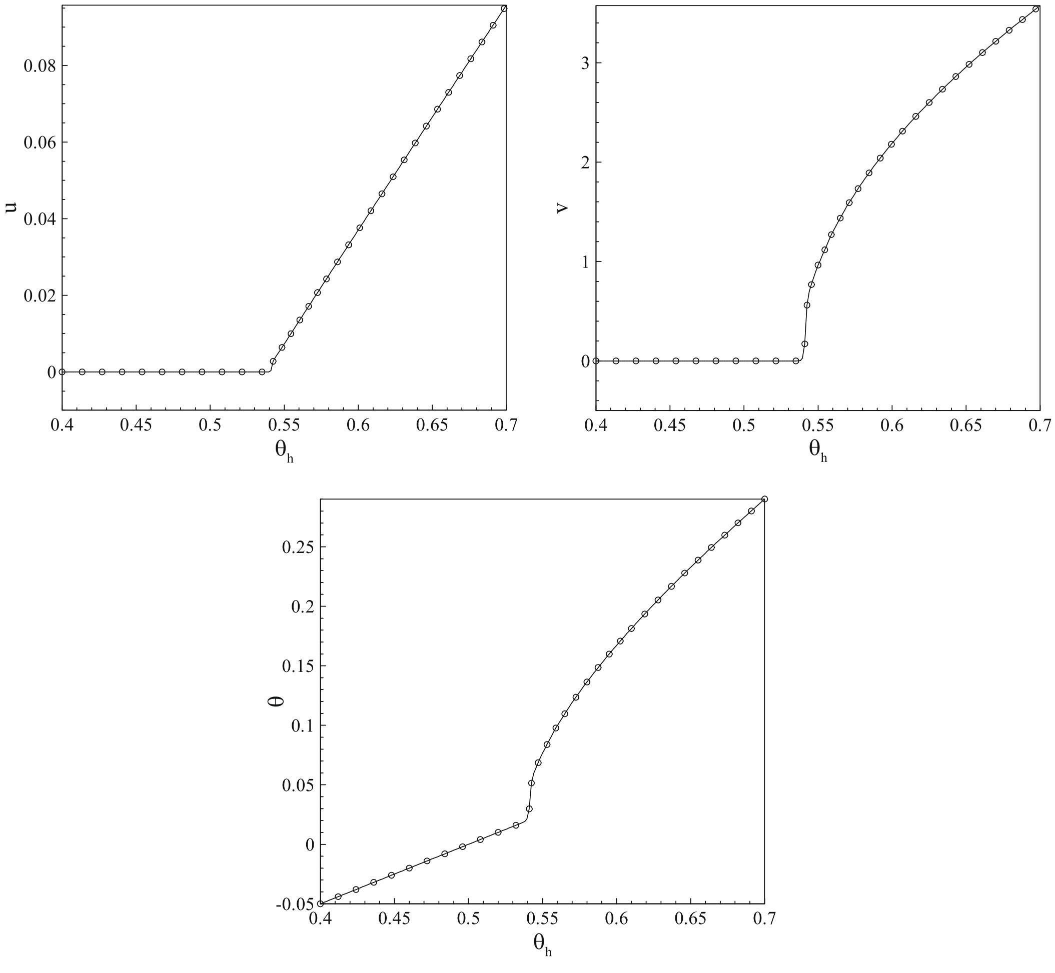


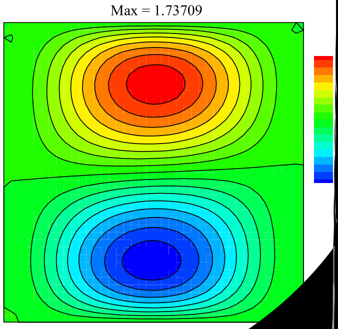
Fig. 22. Solution of the state variables versus hot wall temperature at point (0.1, 0.5). Top left: *u* velocity, Top right: *v* velocity, Bottom: temperature.

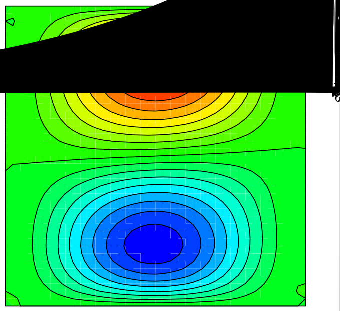
hot wall temperature below the critic

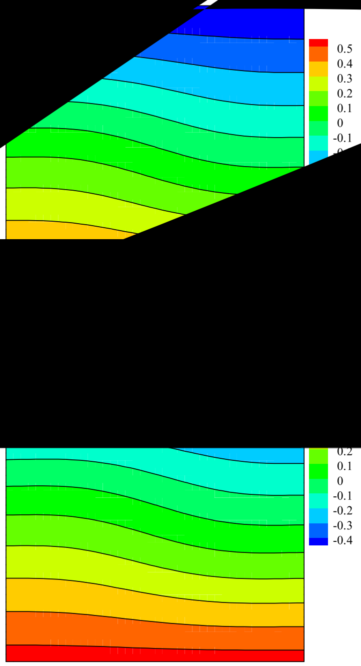
temperature grows non-linearly beyon

To further verify the results, we san

and reconstruct the solution from the







same realization of the random variable. The comparison of the results is shown in Fig. 23. The velocities are zero and thus they are not shown. We can see that the contour distribution of the temperature is characterized by parallel horizontal lines which is a typical distribution for heat conduction. The prediction from the collocation method is the same as the deterministic solution. We repeat this process by sampling a hot wall temperature from the convection regime. The results are shown in Fig. 24. Again, the results compare very well and correctly predict the convective behavior. It is interesting to note that the difference of the maximum value of the velocity between the predicted and deterministic solutions is within the order $O(10^{-3})$, which is consistent with the error threshold 10^{-2}.

Finally, we provide the mean and variance in Figs. 25 and 26, respectively. We also include for comparison the results obtained with the MC-SOBOL sequence with 10^4 iterations. The results compare very well, which again verifies the accuracy of the present method. The number of collocation points, i.e., the number of runs of the deterministic simulator needed for the adaptive sparse grid is 49, while the computation time is about 1.5 h. However, the computation time for MC-SOBOL is 15 h. Therefore, the time needed by the MC method is much more than that for the ASGC.

In summary, the ASGC method can successfully capture the unstable equilibrium in natural convection. In addition, it can also predict quite accurately the critical point.

5. Conclusions

In this paper, we developed an adaptive hierarchical sparse grid collocation method based on the error control of local hierarchical surpluses. By utilizing multi-linear hierarchical basis functions of local support, this method can resolve successfully discontinuities in the stochastic space. Through numerical examples, we demonstrated that in the presence of discontinuity in the stochastic space, this approach leads to significant reduction in the number of points required to achieve the same level of accuracy as the CSGC method. Unlike the dimension-adaptive collocation method developed earlier, the current method refines the sparse grids locally by working directly in the hierarchical basis. Thus, besides the detection of important dimensions as dimension-adaptive methods can do, additional singularity and local behavior of the solution can also be revealed. By applying this method to the Rayleigh–Bénard instability, it is shown that the adaptive sparse grid can accurately predict the critical point.

We provided extensive comparisons with the ME-gPC and ME-PCM methods (note that the ME-PCM work [31] was submitted for publication to this journal long after the initial submission of this work and an adaptive version of it was implemented here during revision). Due to the decoupled nature of the stochastic collocation method, it is shown that the computational cost of the ASGC and ME-PCM methods is much less than that of the ME-gPC method at least for the low-dimensional problems considered. The ME-PCM is even faster than the ASGC in terms of computational time for low-dimensional problems. However, the ASGC requires much less number of collocation points than the ME-PCM to achieve the same accuracy. Furthermore, the multi-element based method is not suitable for high-dimensional problems due to its $O(2^N)$ tree-like scaling of the standard *h*-type adaptive refinement. An interesting topic is to combine the ASGC and multi-element method, i.e., solving stochastic problems using the ASGC in each random element. The accuracy and efficiency of this method is worth further investigation.

Solutions are also compared with MC results. For the 3D K–O problem, due to the rather strong discontinuity, the convergence rate of the AGSC is not optimal compared with MC. However, in general, the ASGC can achieve a desired accuracy at a cost much lower than that of the MC method provided that the ASGC is effective in detecting the regularity in the stochastic space. On the other hand, as shown in the stochastic elliptic problem, if each dimension weighs almost equally for a high-dimensional problem, the ASGC is not the best choice. This is because although the Smolyak algorithm depends less on dimensionality than the gPC method, it still suffers with increasing number of dimensions due to the weak dependence on the dimensions in the logarithmic term of the error bound. Therefore, for problems with high stochastic dimensions, the Monte Carlo method, which is independent of the number of random dimensions, is more favorable than any other method discussed in this paper.

It is also worth noting that the ASGC method not only gives us the solution statistics the same as using MC, but also calculates an approximate functional representation of the solution in the stochastic space. Therefore, in the context of function approximation, MC is not applicable to solve stochastic problems. Currently, the ME-PCM is based on quadrature rule and only gives us mean and variance. Although it can provide response surface of the solution by projecting the collocation solution onto the gPC basis, it is not as straightforward as the ASGC since it needs to search in the multi-element grid.

To increase the convergence rate of this method, we are currently implementing the use of higher-order polynomials of local support as introduced in [23,27]. In addition, since the local refinement still treats each dimension equally, we plan to couple the local adaptive nature of the current methodology with the dimension-adaptive method that can detect locally the important dimension(s).

Acknowledgments

This research was supported by the Computational Mathematics program of AFOSR (Grant F49620-00-1-0373) and by the Computational Mathematics program of the NSF (Award DMS-0809062). The computational work was supported in part by the National Science Foundation through the TeraGrid resources provided by NCSA.

References

[1] R.L. Iman, W.J. Conover, Small sample sensitivity analysis techniques for computer models, with an application to risk assessment, Commun. Statist. A9 (1980) 1749–1842.

[2] R. Ghanem, P.D. Spanos, Stochastic Finite Elements: A Spectral Approach, Springer-Verlag, New York, 1991.

[3] N. Wiener, The homogeneous chaos, Am. J. Math. 60 (1938) 897–936.

[4] R. Ghanem, Probabilistic characterization of transport in heterogeneous media, Comput. Meth. Appl. Mech. Eng. 158 (1998) 199–220.

[5] R. Ghanem, A. Sarkar, Mid-frequency structural dynamics with parameter uncertainty, Comput. Meth. Appl. Mech. Eng. 191 (2002) 5499–5513.

[6] R. Ghanem, Higher order sensitivity of heat conduction problems to random data using the spectral stochastic finite element method, ASME J. Heat Transfer 121 (1999) 290–299.

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

[8] D. Xiu, G.E. Karniadakis, Modeling uncertainty in steady state diffusion problems via generalized polynomial chaos, Comput. Meth. Appl. Mech. Eng. 191 (2002) 4927–4948.

[9] D. Xiu, G.E. Karniadakis, Modeling uncertainty in flow simulations via generalized polynomial chaos, J. Comput. Phys. 187 (2003) 137–167.

[10] D. Xiu, G.E. Karniadakis, A new stochastic approach to transient heat conduction modeling with uncertainty, Int. J. Heat Mass Transfer 46 (2003) 4681–4693.

[11] I. Babuska, R. Tempone, G.E. Zouraris, Galerkin finite elements approximation of stochastic finite elements, SIAM J. Numer. Anal. 42 (2004) 800–825.

[12] M.K. Deb, I.K. Babuska, J.T. Oden, Solution of stochastic partial differential equations using the Galerkin finite element techniques, Comput. Meth. Appl. Mech. Eng. 190 (2001) 6359–6372.

[13] I. Babuska, R. Tempone, G.E. Zouraris, Solving elliptic boundary value problems with uncertain coefficients by the finite element method: the stochastic formulation, Comput. Meth. Appl. Mech. Eng. 194 (2005) 1251–1294.

[14] B. Velamur Asokan, N. Zabaras, Using stochastic analysis to capture unstable equilibrium in natural convection, J. Comput. Phys. 208 (2005) 134–153.

[15] O.P. Le Maître, H.N. Najm, R.G. Ghanem, O.M. Knio, Uncertainty propagation using Wiener–Haar expansions, J. Comput. Phys. 197 (2004) 28–57.

[16] O.P. Le Maître, H.N. Najm, R.G. Ghanem, O.M. Knio, Multi-resolution analysis of Wiener-type uncertainty propagation schemes, J. Comput. Phys. 197 (2004) 502–531.

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

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

[19] X. Wan, G.E. Karniadakis, Long-term behavior of polynomial chaos in stochastic flow simulations, Comput. Meth. Appl. Math. Eng. 195 (2006) 41–43.

[20] I. Babuska, F. Nobile, R. Tempone, A stochastic collocation method for elliptic partial differential equations with random input data, ISIAM J. Numer. Anal. 45 (2007) 1005–1034.

[21] S. Smolyak, Quadrature and interpolation formulas for tensor product of certain classes of functions, Soviet Math. Dokl. 4 (1963) 240–243.

[22] T. Gerstner, M. Griebel, Numerical integration using sparse grids, Numer. Algorithms 18 (1998) 209–232.

[23] V. Barthelmann, E. Novak, K. Ritter, High dimensional polynomial interpolation on sparse grids, Adv. Comput. Math. 12 (2000) 273–288.

[24] A. Klimke, B. Wohlmuth, Algorithm 847: spinterp: piecewise multilinear hierarchical sparse grid interpolation in MATLAB, ACM Trans. Math. Software 31 (2005).

[25] A. Klimke, Uncertainty Modeling using Fuzzy Arithmetic and Sparse Grids, Ph.D. Thesis, Universität Stuttgart, Shaker Verlag, Aachen, 2006.

[26] M. Griebel, Adaptive sparse grid multilevel methods for elliptic PDEs based on finite differences, Computing 61 (1998) 151–179.

[27] H.J. Bungartz, M. Griebel, Sparse grids, Acta Numerica 13 (2004) 147–269.

[28] D. Xiu, J.S. Hesthaven, High order collocation methods for the differential equation with random inputs, SIAM J. Sci. Comput. 27 (2005) 1118–1139.

[29] D. Xiu, Efficient collocational approach for parametric uncertainty analysis, Commun. Comput. Phys. 2 (2007) 293–309.

[30] F. Nobile, R. Tempone, C. Webster, A sparse grid collocation method for elliptic partial differential equations with random input data, SIAM J. Numer. Anal. 45 (2008) 2309–2345.

[31] 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.

[32] D.L. Donoho, Interpolating wavelet transforms, preprint, Stanford University, 1992.

[33] S. Bertoluzza, G. Naldi, A wavelet collocation method for the numerical solution of partial differential equations, Appl. Comput. Harmon. Anal. 3 (1996) 1–9.

[34] O.V. Vasilyev, S. Paolucci, A dynamically adaptive multilevel wavelet collocation method for solving differential equations in a finite domain, J. Comput. Phys. 125 (1996) 498–512.

[35] O.V. Vasilyev, S. Paolucci, A fast adaptive wavelet collocation algorithm for multidimensional PDEs, J. Comput. Phys. 138 (1997) 16–56.

[36] T. Grestner, M. Griebel, Dimension adaptive tensor product quadrature, Computing 71 (2003) 65–87.

[37] B. Ganapathysubramanian, N. Zabaras, Sparse grid collocation schemes for stochastic natural convection problems, J. Comput. Phys. 225 (2007) 652–685.

[38] F. Nobile, R. Tempone, C. Webster, An anisotropic sparse grid collocation method for elliptic partial differential equations with random input data, SIAM J. Numer. Anal. 46 (2008) 2411–2442.

[39] M. Griebel, Sparse grids and related approximation schemes for high dimensional problems, in: Proceedings of the Conference on Foundations of Computational Mathematics, Santander, Spain, 2005.

[40] B. Oksendal, Stochastic Differential Equations: An introduction with applications, Springer-Verlag, New York, 1998.

[41] H.-J. Bungartz, S. Dirnstorfer, Multivariate quadrature on adaptive sparse grids, Computing 71 (2003) 89–114.

[42] A. Klimke, Sparse Grid Interpolation Toolbox – User's Guide, IANS report 2006/001, University of Stuttgart, 2006.

[43] X. Ma, N. Zabaras, A stabilized stochastic finite element second-order projection method for modeling natural convection in random porous media, J. Comput. Phys. 227 (2008) 8448–8471.