ORNL REPORT

ORNL/TM-2013/384
Unlimited Release
Printed September 2013

Hierarchy-Direction Selective Approach for Locally Adaptive Sparse Grids

M. Stoyanov

Prepared by
Oak Ridge National Laboratory
One Bethel Valley Road, Oak Ridge, Tennessee 37831

The Oak Ridge National Laboratory is operated by UT-Battelle, LLC,
for the United States Department of Energy under Contract DE-AC05-00OR22725.
Approved for public release; further dissemination unlimited.

OAK RIDGE NATIONAL LABORATORY
MANAGED BY UT-BATTELLE FOR THE DEPARTMENT OF ENERGY

ORNL/TM-2013/384

Computer Science and Mathematics Division

HIERARCHY-DIRECTION SELECTIVE APPROACH FOR LOCALLY ADAPTIVE SPARSE GRIDS

M. Stoyanov *

Date Published: September 2013

Prepared by
OAK RIDGE NATIONAL LABORATORY
Oak Ridge, Tennessee 37831-6283
managed by
UT-BATTELLE, LLC
for the
U.S. DEPARTMENT OF ENERGY
under contract DE-AC05-00OR22725

*Computer Science and Mathematics Division, Oak Ridge National Laboratory, One Bethel Valley Road, P.O. Box 2008, MS-6367, Oak Ridge, TN 37831-6164 (stoyanovmk@ornl.gov).

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

We consider the problem of multidimensional adaptive hierarchical interpolation. We use sparse grids points and functions that are induced from a one dimensional hierarchical rule via tensor products. The classical locally adaptive sparse grid algorithm uses an isotropic refinement from the coarser to the denser levels of the hierarchy. However, the multidimensional hierarchy provides a more complex structure that allows for various anisotropic and hierarchy selective refinement techniques. We consider the more advanced refinement techniques and apply them to a number of simple test functions chosen to demonstrate the various advantages and disadvantages of each method. While there is no refinement scheme that is optimal for all functions, the fully adaptive family-direction-selective technique is usually more stable and requires fewer samples.

ACKNOWLEDGEMENTS

The ORNL is operated by UT-Battelle, LLC, for the United States Department of Energy under Contract DE-AC05-00OR22725.

1 Introduction

Many real world phenomena are modeled as a set of deterministic algebraic and differential equations, where the solution and the outputs of interest depends on a set of operational parameters. For example, the power output of a combustion engine is a function of the concentration of various chemical species in the cylinder, spark timing, intake and exhaust pressure etc [6]. However, in practice, exact values of the input parameters are seldom available as they may be difficult to measure or they may vary depending on external factors (e.g. external temperature). Therefore, in order to accurately predict the behavior of a physical system, we must consider the full range of possible input parameters.

A common approach is to make every uncertain parameter into an extra dimension of the associated equations and then solve the higher dimensional model. For example, if the physical process is modeled by a partial differential equation (PDE) and the uncertain inputs are parametrization of a noise field via a Karhunen-Lovéle expansion [17], then the PDE can be discretized in both real and stochastic domain using a finite element scheme [2, 3, 5, 10, 15]. This technique is commonly known as stochastic Galerkin method (SGM) and it can accurately approximate the solution to the PDE over the full range of the uncertainty parameters. However, the intrusive nature of this approach requires solving a fully coupled system of equations over both stochastic and real spaces, which significantly increases the computational cost. In addition, the numerical code associated with the the intrusive methods is significantly more complex, and while robust intrusive tools for some types of PDEs exist (e.g. via template programming), such tools are not available for general multi-physics problems (e.g. non-linear coupled PDEs). Furthermore, in many situations, restrictions on code licensing or the need to rely on legacy code, forces scientists and engineers to work only with deterministic solvers, i.e. a software that can only treats the inputs as fixed values.

Sampling methods are an alternative to the intrusive approach. A number of realization of the deterministic problem are computed for a set of values of the input parameters. From that set of samples, the behavior of the solution or an output of interest is inferred over the entire domain of uncertainty. In addition, the samples can often times be computed completely independently leading to natural parallelism. The Monte Carlo (MC) family of sampling methods is the oldest and one of the most popular approaches to estimating the statistics of outputs of interest, see [7, 14, 19] and references therein. The MC samples are taken at random according to a pre-selected probability distribution, and one of the main advantages of the method is that the convergence rate is independent from the regularity of the problem or the number of inputs. However, this independence comes at the expense of very slow convergence of $O(N^{-1/2})$ to $O(N^{-1})$, where N is the number of samples. In many cases, the number of samples needed to achieve an accurate approximation is prohibitively large.

For problems with low to moderate number of inputs as well as smooth dynamics (i.e. multiple bounded derivatives), methods for structured sampling have demonstrated much faster convergence. The most popular approach for structured multidimensional sampling is the sparse grids approach [1, 8, 12, 20–22, 24, 25]. One dimensional interpolation and integration rules (e.g. Clenshaw-Curtis or Gauss-Legendre [4, 23]) are extended to multidimensional rules by taking a select set of tensors. This approach has been demonstrated to converge exponentially for outputs of interest

with globally smooth behavior over the entire range of the uncertain parameters.

A more difficult problem is to approximate locally sharp dynamics, i.e. regions with very large or unbounded derivatives. Approaches have been proposed that are based on a combination of sparse grids and hierarchical basis functions with local support [9, 11, 13, 16, 18]. The idea of one dimensional local adaptivity based on hierarchical surpluses is extended to the multidimensional context. A multidimensional hierarchy is considered and samples from the next level of the hierarchy are taken only in the vicinity of current samples with large multidimensional surplus. However, the multidimensional hierarchy has a more complex structure that allows for a large variety of refinement techniques in both space (i.e. locally isotropic or anisotropic refinement) and hierarchy (i.e. refinement by selecting samples from both higher and lower levels of the hierarchy).

In this paper, we propose several techniques for multidimensional locally adaptive sampling. Unlike the classical approach, we consider both isotropic and anisotropic selection of the refinement as well as simultaneous selection from multiple levels of the hierarchy. We demonstrate that the choice of selection technique can have a significant impact on the stability and convergence of the adaptive approximation.

In Section 2 of the paper, we presents the problem of multidimensional interpolation as well as the various techniques for adaptivity and local refinement. Section 3 gives numerical comparison between the various refinement techniques for a number of functions.

2 Multidimensional Interpolation

In this section we present the problem of multidimensional interpolation using function basis and abscissas induced by tensor products of one dimensional rules. In particular, we are interested in nested hierarchical interpolation local adaptivity.

2.1 One Dimensional Hierarchical Family of Points and Functions

Let $\Gamma \subset \mathbb{R}$ be a bounded interval and let $\mathbb{N}_0$ be the set of non-negative integers. We consider a sequence of nested hierarchical grids (meshes) defined over Γ. The standard approach for labeling the points is to use two indexes corresponding to the level and the index within each level, however, this can significantly and unnecessarily complicate our notation in the multidimensional context. We take a single index approach and order all the points in a sequence $\{x_j\}_{j\in\mathbb{N}_0}$ with an associated index to level map $g(j) : \mathbb{N}_0 \to \mathbb{N}_0$, so that x_j is associated with level $l = g(j)$. We define the hierarchy via a number of index sets. We define the level index sets $D^l \subset \mathbb{N}_0$ as the indexes of points associated with level l

$$D^l = g^{-1}(l) = \{i \in \mathbb{N}_0 : g(i) = l\}.$$

The cumulative level sets V^l are defined as all points at levels less than or equal to l

$$V^0 = D^0, \qquad V^l = V^{l-1} \bigcup D^l \equiv \{i \in \mathbb{N}_0 : g(i) \leq l\}.$$

In addition, we have the hierarchical family structure defined by the index sets P_j and O_j that give the parents and children associated with point x_j

$$\begin{aligned} P_j &= \{i \in \mathbb{N}_0 : x_i \text{ is a parent of } x_j\}, \\ O_j &= \{i \in \mathbb{N}_0 : x_i \text{ is a child (offspring) of } x_j\}. \end{aligned}$$

The points on level $l = 0$ have no parents

$$D^0 \equiv \{i \in \mathbb{N}_0 : P_i = \emptyset\},$$

and in all other cases the parents of a point x_j are on the previous level,

$$P_j \subset D^{g(j)-1}.$$

For all points, the children belong to the next level

$$O_j \subset D^{g(j)+1}.$$

Thus, the hierarchical grid is described via the points $\{x_j\}_{j\in\mathbb{N}_0}$, levels given by $g(j)$ and D^l, and the family relations defined by P_j and O_j. Table 1 gives a glossary of the notation used throughout this section.

In addition to the grid points, we have corresponding set of basis functions $\phi_j(x) : \Gamma \to \mathbb{R}$, so that every funciton is associated with one point and the functions and points form the same hierarchy defined by the same index sets D^l, P_j and O_j.

Example: Uniform Hierarchical Mesh and Piece-Wise Polynomials with Local Support

Consider the example of nested hierarchical points on $\Gamma = [-1, 1]$ given on Figure 1. The points on each level are uniformly spaced starting with the midpoint 0 at level zero, at level 1 we add the two end points -1 and 1, then for every level $l > 1$ we add points that bisect the intervals defined by the points on levels less then l. We index the points top to bottom in terms of level and left to right at each level. Hence the first point on level 0 is $x_0 = 0$, then on level 1 we have $x_1 = -1$ and $x_2 = 1$, then on level 3 we have $x_3 = -0.5$ and $x_4 = 0.5$. The point to level map is given by

$$g(j) = \begin{cases} 0, & j = 0, \\ 1, & j = 1, \\ 1 + \lfloor \log_2(j-1) \rfloor, & j > 1, \end{cases}$$

where $\lfloor x \rfloor = \max\{i \in \mathbb{N}_0 : i < x\}$ is the floor function. The points x_j are given by

$$x_j = \begin{cases} 0, & j = 0, \\ -1, & j = 1, \\ 1, & j = 2, \\ -3 + 2^{-g(j)}\,(4j-2)\,, & j > 2. \end{cases} \tag{2.1}$$

The first few index sets D^l are

$$D^0 = \{0\}, \qquad D^1 = \{1, 2\}, \qquad D^2 = \{3, 4\}, \qquad D^3 = \{5, 6, 7, 8\}, \cdots$$

The family relations are described via the sets

$$\begin{aligned} O_0 &= \{1, 2\}, \qquad Q_1 = \{3\}, \qquad O_2 = \{4\}, \qquad O_j = \{2j, 2j-1\}, \\ P_0 &= \emptyset, \qquad P_1 = P_2 = \{0\}, \qquad P_3 = \{1\}, \qquad P_4 = \{2\}, \qquad P_j = \left\{ \left\lfloor \frac{j+1}{2} \right\rfloor \right\}. \end{aligned}$$

Note that the hierarchy in this particular example has a tree structure, however, this is not a general requirement, e.g. one point can have multiple parents or level zero can have multiple points.

We consider functions $\phi_j(x)$ that are defined as piece-wise polynomials with local support. For illustration purposes, on Figure 1, we show the linear, quadratic and cubic polynomials. The functions $\phi_j(x)$ associated with each x_j are defined as

$$\phi_0^{linear}(x) = 1, \qquad \phi_j^{linear}(x) = \begin{cases} 1 - 2^{1-g(j)}|x - x_j|, & 2^{1-g(j)}|x - x_j| < 1 \\ 0, & otherwise, \end{cases} \tag{2.2}$$

the second order basis is defined as

$$\begin{aligned} \phi_0^{quadratic}(x) &= 1, \qquad \phi_1^{quadratic}(x) = 0.5x(x-1), \qquad \phi_2^{quadratic}(x) = 0.5x(x+1) \\ \phi_j^{quadratic}(x) &= \begin{cases} 1 - 4^{1-g(j)}|x - x_j|^2, & 2^{1-g(j)}|x - x_j| < 1 \\ 0, & otherwise, \end{cases} \end{aligned} \tag{2.3}$$

third order polynomials are defined as

$$\begin{aligned}\phi_0^{cubic}(x) &= 1, \qquad \phi_1^{cubic}(x) = 0.5x(x-1), \qquad \phi_2^{cubic}(x) = 0.5x(x+1),\\ \phi_j^{cubic}(x) &= \begin{cases} \frac{1}{3}\left(3 + 2^{1-g(j)}|x - x_j|\right)\phi_j^{quadratic}(x), & j \text{ is even},\\ \frac{1}{3}\left(3 - 2^{1-g(j)}|x - x_j|\right)\phi_j^{quadratic}(x), & j \text{ is odd}. \end{cases}\end{aligned} \tag{2.4}$$

In order to illustrate our methodology, we restrict our attention to only the linear, quadratic and cubic basis. However, higher order polynomials can also be constructed, see [11] for details.

2.2 One Dimensional Hierarchical Adaptive Interpolation

Let $f(x) \in \mathbb{C}^0(\Gamma)$, i.e. $f(x)$ is a continuous function defined over the interval of interest Γ. We assume that $f(x)$ has complex behavior and evaluating $f(x')$ for a specific $x' \in \Gamma$ is very computationally expensive (e.g. it requires solving one or more differential equations). Our goal is to create an interpolant that approximates $f(x)$ and we want to achieve maximum accuracy with as few evaluations of $f(x)$ as possible.

Given a hierarchical grid $\{x_j\}_{j\in\mathbb{N}_0}$ and basis functions $\{\phi_j(x)\}_{j\in\mathbb{N}_0}$, an interpolant $\mathcal{I}$ is defined by an index set $S \subset \mathbb{N}_0$ as

$$\mathcal{I}_S(f) : \mathbb{C}^0(\Gamma) \to span\{\phi_j(x)\}_{j\in S}, \qquad \mathcal{I}_S(f)(x) = \sum_{j\in S} c_j\phi_j(x), \tag{2.5}$$

where the coefficients c_j are chosen so that the value of the interpolant matches the value of the function at the sample points

$$\sum_{j\in S} c_j\phi_j(x_i) = f(x_i), \text{ for all } i \in S. \tag{2.6}$$

Condition (2.6) gives us an $n \times n$ system of equations, where n is the number of indexes in S. In general, the conditioning of the linear system is of consideration, however, there are well known hierarchical grids and basis that allow us to find c_j in a stable way. For example, the points and functions described in (2.1) and (2.2-2.4) have the property

$$\phi_j(x_i) = \delta_{i,j}, \quad \text{for all} \quad j \in D^l, i \in V^l, \tag{2.7}$$

where $\delta_{i,j}$ is the Kronecker delta. If a hierarchical basis has this property (2.7), then the coefficients c_j can be found recursively via the relation

$$c_j = f(x_j) - \mathcal{I}_{S\cap V^{g(j)-1}}(f)(x_j), \tag{2.8}$$

where, to simplify notation, we assume that $V^{-1} = \emptyset$ and $\mathcal{I}_\emptyset(f)(x) = 0$. Combined with the local properties of the basis functions $\phi_j(x)$, the coefficients c_j are a computable practical local error indicator.

For any finite set of points indexed by S, there is a function $f'(x) \in \mathbb{C}^0(\Gamma)$ so that

$$\|\mathcal{I}_S(f)(x) - f'(x)\|_\infty = 1,$$

therefore, no interpolant can approximate all functions in $\mathbb{C}^0(\Gamma)$ to an arbitrary tolerance. The same is true if the $\mathbb{L}^2$-norm is used. We need to dynamically update the accuracy of the interpolant by identifying regions where $f(x)$ is not well approximated (e.g. local sharp transitions) and then add more points in those regions. Furthermore, we also want to use as few sample points as possible in regions where $f(x)$ is already well approximated.

One way to adapt the interpolant is to consider full levels V^l for $l = 1, 2, 3$ and use the interpolant with smallest level l for which

$$\|\mathcal{I}_{V^l}(f)(x) - \mathcal{I}_{V^{l-1}}(f)(x)\|_\infty < \epsilon, \tag{2.9}$$

where ϵ is the desired error tolerance. Note that if the coefficients c_j are computed via the relation (2.8), then (2.9) is equivalent to the magnitude of the largest coefficient on the level l

$$\|\mathcal{I}_{V^l}(f)(x) - \mathcal{I}_{V^{l-1}}(f)(x)\|_\infty = \left\| \sum_{j \in D^l} c_j \phi_j(x) \right\|_\infty = \max_{j \in D^l} |c_j|.$$

Thus, we can use the coefficients c_j as an indicator of the accuracy of our interpolant.

However, considering only full levels of the hierarchy can add unnecessary number of points in regions where $f(x)$ is very smooth. We need to consider only a subset of points on each level and we can use the coefficients c_j as local error indicators. We include points from the next level of the hierarchy only in the vicinity of points x_j that are associated with coefficients c_j with large magnitude.

Let l be the initial level of the interpolant and let ϵ be the desired tolerance for our error indicator. We can iterate $k = 0, 1, 2, \cdots$ and build a sequence of interpolants

$$\mathcal{I}^{k,l,\epsilon}(f)(x) = \mathcal{I}_{S^{k,l,\epsilon}}(f)(x),$$

where the index sets $S^{k,l,\epsilon}$ are defined recursively. For $k = 0$ the initial index set is

$$S^{0,l,\epsilon} = V^l.$$

The next set $S^{k+1,l,\epsilon}$ is constructed by first considering the points with big coefficients

$$B^{k,l,\epsilon} = \{j \in S^{k,l,\epsilon} : |c_j| > \epsilon\}.$$

The set $B^{k,l,\epsilon}$ gives us the indexes of points that are candidates for refinement. For each candidate point $j \in B^{k,l,\epsilon}$, we define the refinement set $R_j^{k,l,\epsilon}$ as

$$R_j^{k,l,\epsilon} = O_j. \tag{2.10}$$

The interpolant for the $k+1$ iteration is defined by the set

$$S^{k+1,l,\epsilon} = S^{k,l,\epsilon} \bigcup \left(\bigcup_{j \in B^{k,l,\epsilon}} R_j^{k,l,\epsilon} \right).$$

Thus, at each iteration, we update the interpolant by adding the children of the points with large surpluses. We terminate the iteration when

$$S^{k+1,l,\epsilon} = S^{k,l,\epsilon},$$

and we say that the iteration has "converged". We use $\mathcal{I}^{k,l,\epsilon}(f)(x)$ with the last k as our adaptively constructed hierarchical interpolant.

The choice of $R_j^{k,l,\epsilon}$ on equation (2.10) is unique in the context of grid points with tree structure, e.g. the hierarchical piece-wise polynomials on Figure 1. However, should the hierarchical basis fail to form a proper tree, then the choice of $R_j^{k,l,\epsilon}$ is no longer trivial and it can have significant impact on the accuracy, convergence and number of points of the adaptive interpolant. This is particularly true in the multidimensional context, where points have multiple parents and children associated with different direction.

2.3 Multidimensional Hierarchical Family of Functions

We consider a multidimensional hierarchy of points and functions that is induced from a single dimensional rule by tensor products. We use bold letters to identify multi-indexes, sets of multi-indexes and the multidimensional point to level map. Let d be the number of dimensions and $\mathbb{N}_0^d$ be the set of multi-indexes

$$\mathbb{N}_0^d = \mathbb{N}_0 \otimes \mathbb{N}_0 \otimes \cdots \otimes \mathbb{N}_0,$$

with elements

$$\mathbf{j} = (j_1, j_2, \cdots, j_d) \in \mathbb{N}_0^d.$$

The order of a multi-index $|\mathbf{j}|$ is defined as

$$|\mathbf{j}| = \sum_{\alpha=1}^{d} j_\alpha.$$

Note that we use the Greek letters (e.g. α and β) to denote direction. The domain of interest is

$$\Gamma^d = \Gamma \otimes \Gamma \otimes \cdots \otimes \Gamma \subset \mathbb{R}^d,$$

and a point in the multidimensional hierarchy is a vector of points from the one dimensional hierarchy defined by a multi-index $\mathbf{j}$

$$\mathbf{x}_\mathbf{j} = (x_{j_1}, x_{j_2}, \cdots, x_{j_d}) \in \Gamma^d.$$

The point to level map is a multi-index relation

$$\mathbf{g}(\mathbf{j}) : \mathbb{N}_0^d \to \mathbb{N}_0^d, \qquad \mathbf{g}(\mathbf{j}) = (g(j_1), g(j_2), \cdots, g(j_d)) .$$

The level associated with a point $\mathbf{x_j}$ is the order of $|\mathbf{g}(\mathbf{j})|$, e.g.

$$\mathbf{D}^l = \left\{ \mathbf{i} \in \mathbb{N}_0^d : |\mathbf{g}(\mathbf{i})| = \sum_{\alpha=1}^{d} g(j_\alpha) = l \right\} .$$

The multidimensional cumulative level sets $\mathbf{V}^l$ are defined as before

$$\mathbf{V}^0 = \mathbf{D}^0, \qquad \mathbf{V}^l = \mathbf{V}^{l-1} \bigcup \mathbf{D}^l.$$

The multidimensional parent and children sets are associated with each direction, $\mathbf{P}_\mathbf{j}^\alpha$ and $\mathbf{O}_\mathbf{j}^\alpha$ are the parents and children of point $\mathbf{x_j}$ in direction α,

$$\begin{aligned} \mathbf{P}_\mathbf{j}^\alpha &= \left\{ \mathbf{i} \in N_0^d : i_\beta = j_\beta \text{ for } \beta \neq \alpha, \text{ and } i_\alpha \in P_{j_\alpha} \right\}, \\ \mathbf{O}_\mathbf{j}^\alpha &= \left\{ \mathbf{i} \in N_0^d : i_\beta = j_\beta \text{ for } \beta \neq \alpha, \text{ and } i_\alpha \in O_{j_\alpha} \right\}, \end{aligned}$$

in other words,

$$\begin{aligned} &(j_1, j_2, \cdots, j_{\alpha-1}, i, j_{\alpha+1}, \cdots, j_d) \in \mathbf{P_j}^\alpha, \text{ if and only if } i \in P_{j_\alpha}, \\ &(j_1, j_2, \cdots, j_{\alpha-1}, i, j_{\alpha+1}, \cdots, j_d) \in \mathbf{O_j}^\alpha, \text{ if and only if } i \in O_{j_\alpha}. \end{aligned}$$

Note that the multidimensional hierarchy does not form a tree since most points have d parents, i.e. one parent in each direction.

The functions associated with the hierarchy are the tensor product of one dimensional functions

$$\mathbf{\Phi_j}(\mathbf{x}) : \Gamma^d \to \mathbb{R}, \qquad \mathbf{\Phi_j}(\mathbf{x}) = \phi_{j_1}(x_1)\phi_{j_2}(x_2)\cdots\phi_{j_d}(x_d).$$

Thus, a multidimensional hierarchy is described by a set of points $\{\mathbf{x_j}\}_{\mathbf{j}\in\mathbb{N}_0^d}$ with associated basis functions $\{\mathbf{\Phi_j}(\mathbf{x})\}_{\mathbf{j}\in\mathbb{N}_0^d}$, the hierarchical structure described via the point to level map $\mathbf{g}(\mathbf{x})$ and parents and children sets $\mathbf{P}_\mathbf{j}^\alpha$ and $\mathbf{O}_\mathbf{j}^\alpha$.

2.4 Multidimensional Hierarchical Adaptive Interpolation

Given a multidimensional hierarchy, we define an interpolant of a multidimensional function $f(\mathbf{x}) \in \mathbb{C}^0(\Gamma^d)$ via a multi-index set $\mathbf{S} \subset \mathbb{N}_0^d$

$$\mathcal{I} : \mathbb{C}^0(\Gamma^d) \to span\{\mathbf{\Phi_j}(\mathbf{x})\}_{\mathbf{j}\in\mathbf{S}}, \qquad \mathcal{I}(f)(\mathbf{x}) = \sum_{\mathbf{j}\in\mathbf{S}} c_\mathbf{j} \mathbf{\Phi_j}(\mathbf{x}), \tag{2.11}$$

where the coefficients $c_{\mathbf{j}}$ are chosen so that $\mathcal{I}(f)(\mathbf{x}_{\mathbf{j}}) = f(\mathbf{x}_{\mathbf{j}})$ for all $\mathbf{j} \in \mathbf{S}$. The only difference between (2.5) and (2.11) is the multi-index notation. If the one dimensional hierarchical functions have the property (2.7), then so does the multidimensional hierarchy and the coefficients $c_{\mathbf{j}}$ can be computed in the same way

$$c_{\mathbf{j}} = f(\mathbf{x}_{\mathbf{j}}) - \mathcal{I}_{\mathbf{S}\cap\mathbf{V}^{|\mathbf{g}(\mathbf{j})|-1}}(f)(\mathbf{x}_{\mathbf{j}}), \tag{2.12}$$

where the previous conventions hold $\mathbf{V}^{-1} = \emptyset$ and $\mathcal{I}_{\emptyset}(f)(\mathbf{x}) = 0$. Interpolants using full levels $\mathcal{I}_{\mathbf{V}^l}(f)(\mathbf{x})$ corresponds to classical Smolyak sparse grids [11, 22]. Figure 2 gives two examples of full level grids using $\mathbf{V}^3$ and $\mathbf{V}^8$ induced by the uniform hierarchical mesh described in (2.1).

Local iterative refinement follows a very similar procedure as the one dimensional interpolant. Given an initial level l and tolerance ϵ, for $k = 0, 1, 2, 3, \cdots$, we construct the sequence of interpolants

$$\mathcal{I}^{k,l,\epsilon}(f)(\mathbf{x}) = \mathcal{I}_{\mathbf{S}^{k,l,\epsilon}}(f)(\mathbf{x}),$$

where the multi-index sets $\mathbf{S}^{k,l,\epsilon}$ are constructed recursively from $\mathbf{S}^{0,l,\epsilon} = \mathbf{V}^l$ and the set of big coefficients $\mathbf{B}^{k,l,\epsilon}$ as

$$\mathbf{B}^{k,l,\epsilon} = \left\{\mathbf{j} \in \mathbf{S}^{k,l,\epsilon} : |c_{\mathbf{j}}| > \epsilon\right\}, \qquad \mathbf{S}^{k+1,l,\epsilon} = \mathbf{S}^{k,l,\epsilon} \bigcup \left(\bigcup_{\mathbf{j}\in\mathbf{B}^{k,l,\epsilon}} \mathbf{R}_{\mathbf{j}}{}^{k,l,\epsilon} \right),$$

where $\mathbf{R}_{\mathbf{j}}{}^{k,l,\epsilon}$ is the refinement set associated with point $\mathbf{x}_{\mathbf{j}}$. However, unlike the one dimensional case, the choice of $\mathbf{R}_{\mathbf{j}}^{k,l,\epsilon}$ is no longer trivial. It is possible for $\mathbf{x}_{\mathbf{j}}$ at iteration k to be associated with a large coefficient (i.e. $\mathbf{j} \in \mathbf{B}^{k,l,\epsilon}$), while some of the parents of $\mathbf{x}_{\mathbf{j}}$ are not included in the interpolant (i.e. $\mathbf{P}_{\mathbf{j}}{}^{\alpha} \not\subset \mathbf{S}^{k,l,\epsilon}$ for some α). Furthermore, we have the opportunity to either refine isotropically in all directions or anisotropically in only a select set of directions. Thus, we have to answer two important questions

- If some of the parents of $\mathbf{j} \in \mathbf{B}^{k,l,\epsilon}$ are not included in $\mathbf{S}^{k,l,\epsilon}$, do we refine the interpolant by adding the parents or the children?
- Do we refine isotropically or anisotropically, i.e. do we add parents $\mathbf{P}_{\mathbf{j}}^{\alpha}$ and children $\mathbf{O}_{\mathbf{j}}^{\alpha}$ for all directions α or only for a select subset? If we chose to refine anisotropically, then how do we chose the refinement directions?

Depending on how we answer the above question, we have multiple possible choices for $\mathbf{R}_{\mathbf{j}}{}^{k,l,\epsilon}$ and the choice of refinement has a significant impact on the stability and convergence of the interpolant.

Classic Refinement

Isotropic refinement that adds only the children of the points with large coefficients was first proposed by Griebel [11]. The refinement sets are defined as

$$\mathbf{R}_{\mathbf{j}}{}^{k,l,\epsilon} = \bigcup_{\alpha=1}^{d} \mathbf{O}_{\mathbf{j}}{}^{\alpha}. \tag{2.13}$$

This approach has been demonstrated to produce accurate interpolants for many problems. However, in some cases, adding only the children may result in an unstable interpolant, i.e. the algorithm may fail to converge, e.g. see Section 3.1. Furthermore, anisotropic refinement may result in an interpolant with the same overall accuracy, but fewer number of points, e.g. see Section 3.3.

Family Selective Refinement

In order to improve the stability of the refinement algorithm, we propose a family selective strategy. If a point $\mathbf{x}_\mathbf{j}$ with $\mathbf{j} \in \mathbf{S}^{k,l,\epsilon}$ is associated with a large coefficient (i.e. $\mathbf{j} \in \mathbf{B}^{k,l,\epsilon}$), then for every direction α we first consider the parents $\mathbf{P}_\mathbf{j}^\alpha$. If the parents are not already included in the interpolant (i.e. $\mathbf{P}_\mathbf{j}^\alpha \not\subset \mathbf{S}^{k,l,\epsilon}$), then we refine by adding the parents, otherwise we add the children $\mathbf{O}_\mathbf{j}^\alpha$. Define $\Delta_\mathbf{j}^{k,l,\epsilon}$ to be the set of "orphan" directions associated with $\mathbf{j} \in \mathbf{B}^{k,l,\epsilon}$, i.e.

$$\Delta_\mathbf{j}^{k,l,\epsilon} = \left\{\alpha \in \{1, 2, \cdots, d\} : \mathbf{P}_\mathbf{j}^\alpha \not\subset \mathbf{S}^{k,l,\epsilon}\right\}.$$

Then, we define the family selective refinement set

$$\mathbf{R}_\mathbf{j}{}^{k,l,\epsilon} = \left(\bigcup_{\alpha \in \Delta_\mathbf{j}^{k,l,\epsilon}} \mathbf{P}_\mathbf{j}^\alpha\right) \bigcup \left(\bigcup_{\alpha \notin \Delta_\mathbf{j}^{k,l,\epsilon}} \mathbf{O}_\mathbf{j}^\alpha\right). \tag{2.14}$$

Adding the parents first improves the stability of the adaptive interpolant. However, in some cases it may result in an interpolant with more points than the classical refinement (2.13), for example see Section 3.2.

Direction Selective Refinement

Sparse grids, induced by interpolation rules with $O(2^l)$ number of points per level, have strong axis bias. Observe on Figure 2, how the grids introduce a lot of points in straight lines, first on the main axis, then on the boundaries, then on the mid-points and so on. It is natural to seek anisotropic refinement strategy that tends to add fewer points in already clustered directions.

We propose a direction selective strategy that is based on a series of one directional interpolants. For every $\mathbf{j} \in \mathbf{S}^{k,l,\epsilon}$ and every direction α, let $\mathbf{W}_{\mathbf{j},\alpha}^{k,l,\epsilon}$ index the points that belong to the same line as $\mathbf{x}_\mathbf{j}$ in the direction α

$$\mathbf{W}_{\mathbf{j},\alpha}^{k,l,\epsilon} = \left\{\mathbf{i} \in \mathbf{S}^{k,l,\epsilon} : i_\beta = j_\beta, \text{ for all } \beta \neq \alpha\right\}.$$

For $\mathbf{x}_\mathbf{j}$ with $\mathbf{j} \in \mathbf{B}^{k,l,\epsilon}$ consider the one directional interpolants $\mathcal{I}_{\mathbf{W}_{\mathbf{j},\alpha}^{k,l,\epsilon}}(f)(\mathbf{x})$ and the corresponding coefficients $c_\mathbf{j}$. Let $\Lambda_\mathbf{j}^{k,l,\epsilon}$ be the set of directions where $\mathbf{x}_\mathbf{j}$ is associated with a large one directional coefficient

$$\Lambda_\mathbf{j}^{k,l,\epsilon} = \left\{\alpha \in \{1, 2, \cdots, d\} : |c_\mathbf{j}| > \epsilon, \text{ where } c_\mathbf{j} \text{ is associated with } \mathcal{I}_{\mathbf{W}_{\mathbf{j},\alpha}^{k,l,\epsilon}}(f)(\mathbf{x})\right\}.$$

The direction selective refinement set is given by

$$\mathbf{R_j}^{k,l,\epsilon} = \bigcup_{\alpha \in \Lambda_{\mathbf{j}}^{k,l,\epsilon}} \mathbf{O}_{\mathbf{j}}^{\alpha}. \tag{2.15}$$

The direction refinement strategy has the effect of reducing the number of points needed for convergence while still preserving the overall accuracy of the final interpolant. Section 3.3 gives an example of the saving provided by anistoropic refinement. This refinement strategy has the additional computational cost associated with forming each of the one directional interpolants $\mathcal{I}_{\mathbf{W}_{\mathbf{j},\alpha}^{k,l,\epsilon}}(f)(\mathbf{x})$, however, all of those interpolants require a subset of the same samples already computed to build $\mathcal{I}^{k,l,\epsilon}(f)(\mathbf{x})$. In most practical applications, the cost of a single evaluation of $f(x)$ dominates the cost of forming the interpolant by many orders of magnitude, hence, a saving of even a single sample would justify the relatively small additional work needed to from the one directional interpolants. The stability of the direction selective interpolant, however, may be reduced as compared to the classical interpolant.

Family Direction Selective Refinement

We combine the refinement strategies in family hierarchy (2.14) and direction (2.15) to have a fully adaptive scheme. For every points with large surplus, we only refine in the directions of large one directional surplus and we refine by first considering the parents and adding the children only if the parents are already included in the current interpolant. The family-direction-selective refinement sets are

$$\mathbf{R_j}^{k,l,\epsilon} = \left(\bigcup_{\alpha \in \Lambda_{\mathbf{j}}^{k,l,\epsilon} \cap \Delta_{\mathbf{j}}^{k,l,\epsilon}} \mathbf{P_j}^{\alpha} \right) \bigcup \left(\bigcup_{\alpha \in \Lambda_{\mathbf{j}}^{k,l,\epsilon} \setminus \Delta_{\mathbf{j}}^{k,l,\epsilon}} \mathbf{O_j}^{\alpha} \right). \tag{2.16}$$

The fully adaptive scheme couples advantages of both selective refinements, while mitigating some of the disadvantages. It is possible for the family direction selective scheme (2.16) to produce an interpolant with more points and less stability than the classical refinement (2.13), for an example see Section 3.4, however, our experience shows that functions associated with such pathological cases are not common in practical applications.

One Dimensional Context	
$\Gamma \subset \mathbb{R}$	interval of interest
$\mathbb{N}_0$	the set of non-negative integers used to index the points and functions
$i, j \in \mathbb{N}_0$	indexes associated with points and functions
$x_j \in \Gamma$	the point on the mesh with index j
$\phi_j(x) : \Gamma \to \mathbb{R}$	the basis function associated with x_j
$g(j)$	the point to level map, i.e. $g(j)$ is the level associated with point x_j
$l \in \mathbb{N}_0$	denotes the level of a point, a function or an interpolant
$D^l \subset \mathbb{N}_0$	the set of indexes associated with level l
$V^l \subset \mathbb{N}_0$	cumulative index set $V^0 = D^0$, $V^l = V^{l-1} \cup D^l$
$P_j \subset D^{g(j)-1}$	the index set of the parents associated with point x_j
$O_j \subset D^{g(j)+1}$	the index set of the children (offspring) associated with point x_j
$S \subset \mathbb{N}_0$	an arbitrary set of indexes
$\mathcal{I}_S(f)(x)$	the interpolant of $f(x)$ using points indexed by S, $\mathcal{I}_S(f)(x) = \sum_{j \in S} c_j \phi_j(x)$
c_j	the coefficient associated with function $\phi_j(x)$
k	denotes the iteration of the adaptive interpolant construction algorithm
ϵ	error tolerance used to compare $\lvert c_j \rvert$
$\mathcal{I}^{k,l,\epsilon}(f)(x)$	the adaptive interpolant at iteration k with initial level l using tolerance ϵ
$S^{k,l,\epsilon}$	the index set of points used by the adaptive interpolant for iteration k, starting at initial level l using tolerance ϵ i.e.$\mathcal{I}^{k,l,\epsilon}(f)(x) = \mathcal{I}_{S^{k,l,\epsilon}}(f)(x)$
$B^{k,l,\epsilon}$	this is the set of indexes $j \in S^{k,l,\epsilon}$ for which the coefficients c_j have magnitude bigger than ϵ i.e. this is the set of points that are candidates for refinement
$R_j^{k,l,\epsilon}$	for every $j \in B^{k,l,\epsilon}$, the indexes of the points that should be added to $S^{k+1,l,\epsilon}$ to improve the interpolant in the vicinity of x_j
Multidimensional Context	
d	denotes the number of dimensions in the multidimensional context
α, β	denote directions in the multidimensional context
$\Gamma^d = \otimes_{\alpha=1}^d \Gamma$	is the multidimensional domain of interest
$\mathbb{N}_0^d = \otimes_{\alpha=1}^d \mathbb{N}_0$	the set of all multi-indexes
$\mathbf{i}, \mathbf{j} \in \mathbb{N}_0^d$	multi-indexes associated with points and functions
$\lvert \mathbf{g(j)} \rvert$	the level of a point $\mathbf{x_j}$ in multidimensional context
$\mathbf{x_j}, \mathbf{\Phi_j(x)}$	points and functions in the multidimensional hierarchy
$\mathbf{D}^l, \mathbf{V}^l, \mathbf{S}$	capital bold letters indicate corresponding sets of multi-indexes
$\mathbf{P_j}^\alpha$, $\mathbf{O_j}^\alpha$	parents and children associated with point $\mathbf{x_j}$ in the direction α
$\mathbf{W}_{\mathbf{j},\alpha}^{k,l,\epsilon}$	the set of multi-indexes of all the points $\mathbf{i} \in \mathbf{S}^{k,l,j}$ that belong to the same line as $\mathbf{x_j}$ in direction α
$\Lambda_{\mathbf{j}}^{k,l,\epsilon}$	the set of directions α, where the one directional interpolant $\mathcal{I}_{\mathbf{W}_{\mathbf{j},\alpha}^{k,l,\epsilon}}(f)(\mathbf{x})$ has a coefficient $c_\mathbf{j}$ with large magnitude (i.e. $\lvert c_\mathbf{j} \rvert > \epsilon$)
$\Delta_{\mathbf{j}}^{k,l,\epsilon}$	the set of directions α, for which $\mathbf{x_j}$ is orphan (i.e. $\mathbf{P_j}^\alpha \not\subset \mathbf{S}^{k,l,\epsilon}$)

Table 1: Glossary of notation.

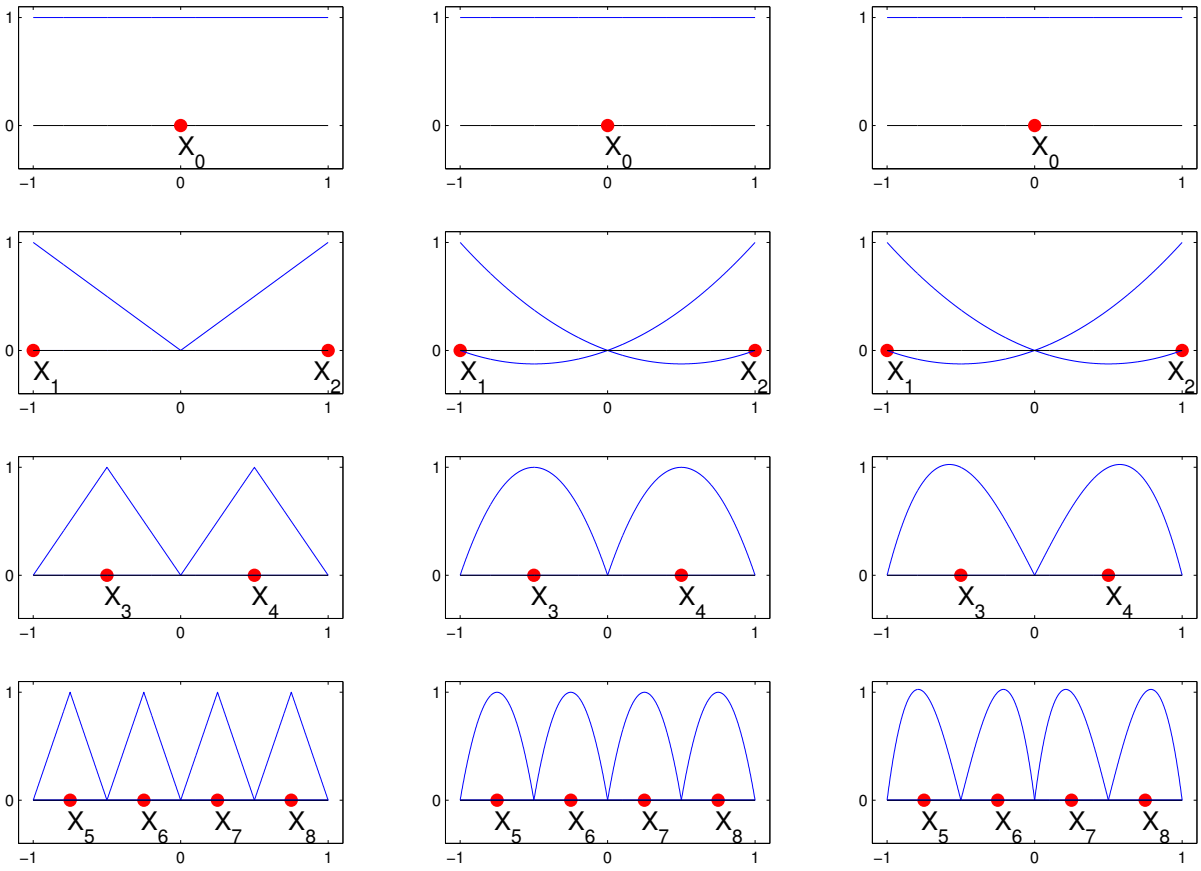

Figure 1: One dimensional hierarchical points and basis functions. On top, we have level 0 with only one point $x_0 = 0$. Level 1 has two points $x_1 = -1$ and $x_2 = 1$. On level 2 we add the two mid-points $x_3 = -0.5$ and $x_4 = 0.5$. On level 3 we have the next four points $x_5 = -0.75$, $x_6 = -0.25$, $x_7 = 0.25$ and $x_8 = 0.75$. The three type of funciton basis are piece-wise local polynomials: linear (left), quadratic (center) and cubic (right).

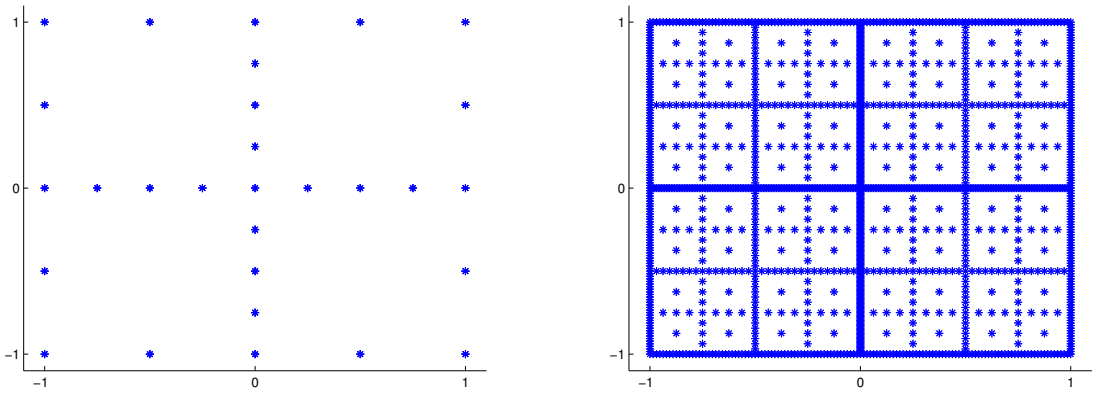

Figure 2: Examples of sparse grids induced by the uniform hierarhical mesh. The two grids correspond to levels 3 (left) and 8 (right). Note how the sparse grids isotropic algorithm clusters points on straight lines, which often times leads to samples with negligible contribution to the overall accuracy of the interpolant.

3 Numerical Examples

In this section we present numerical examples of several test functions. We select a number of simple test functions that are artificially constructed to demonstrate the advantages and disadvantages of the various refinement techniques. Application to problems governed by differential equations is beyond the scope of this work.

All numerical experiments are done using the Sparse Grids module of the Toolkit for Adaptive Stochastic Modeling and Non-Intrusive Approximation (TASMANIAN). The software is available at http://tasmanian.ornl.gov

3.1 Instability Due to Missing Parents

We want to test the adaptive hierarchical interpolation algorithms applied to a function with sharp behavior. Consider the function

$$f(x,y) = \frac{1}{1 + exp(16 - 40\sqrt{x^2 + y^2})}, \tag{3.1}$$

which is plotted on Figure 3. The walls of the sinkhole present a challenge for any interpolation technique.

We use the one dimensional hierarchy described (2.1) with cubic functions basis (2.4) and we take the initial level to be $l = 7$; we use tolerance $\epsilon = 10^{-4}$. The initial grid has 705 points, which is way too few to capture the sharp behavior. We first use the classical refinement given by (2.13). After 13 iterations, the method stagnates. For $k > 13$, every iteration requires exactly 160 new points taken in the vicinity of 4 bad regions around $(\pm 0.265625, \pm 0.265625)$. Figure 4 gives a close up plot of the points in $S^{16,7,10^{-4}}$ around $(0.265625, 0.265625)$.

We run the adaptive algorithm with the same parameters, however, we use the family selective refinement (2.14). In that case, the algorithm converges after 9 iterations with total number of points $9,937$. If we look at the same region that causes problems for the classic refinement, we see that the family selective algorithm has added the point $(0.265625, 0.265625)$ which corresponds to $\mathbf{j} = (104, 104)$. The function $\mathbf{\Phi}_{(104,104)}(\mathbf{x})$ associated with this point has a significant contribution to the local behavior of the sinkhole function, however, the classical refinement technique fails to add that point. Instead, the classical refinement attempts to resolve the behavior with the children, which is unfeasible and hence the method fails to converge.

We could start the initial iteration at a higher level l, however, $\mathbf{g}(104, 104) = (7, 7)$ and hence we need to use minimum level of $l = 14$. At that level, the initial interpolant requires $147,457$ samples, which is unreasonably high when compared to the $9,973$ samples of the family selective iteration.

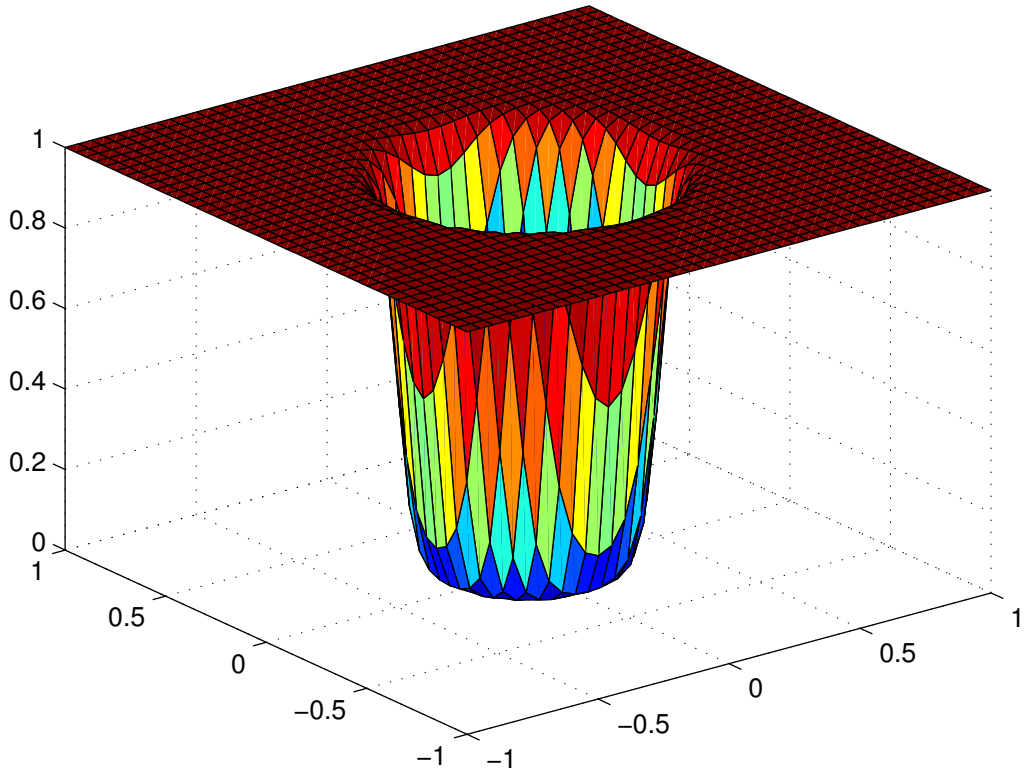

Figure 3: Surface plot of funciton (3.1) using dense grid on the entire domain $[-1, 1]^2$. The sharp behavior of the funciton makes it difficult to interpolate and causes the classical refinement technique (2.13) to become unstable.

3.2 Unnecessary Inclusion of Parents

As a counter example to (3.1), we present a function for which the family selective refinement technique would unnecessarily require more points than the classical refinement. Consider the function

$$f(x, y) = \sin(x) \sin(y) \tag{3.2}$$

which is plotted on Figure 5. Note that each point of the hierarchy has at least one ancestor that belongs to the main axis and the function (3.2) is constant along $x = 0$ and $y = 0$. Samples chosen on the main axis, do not contribute to the accuracy of the interpolant.

We use the one dimensional hierarchy described (2.1) with linear functions basis (2.2) and we use $l = 4$ and $\epsilon = 10^{-4}$. We compare the results from applying the hierarchical adaptive interpolation algorithms with classical (2.13) and family selective (2.14) refinement. Both algorithms converge, however, the classical algorithm requires only $1,049$ points, while the family selective one takes $1,165$. Figure 6 gives the final grids. We note that the family selective algorithm adds a lot of points on the main axis.

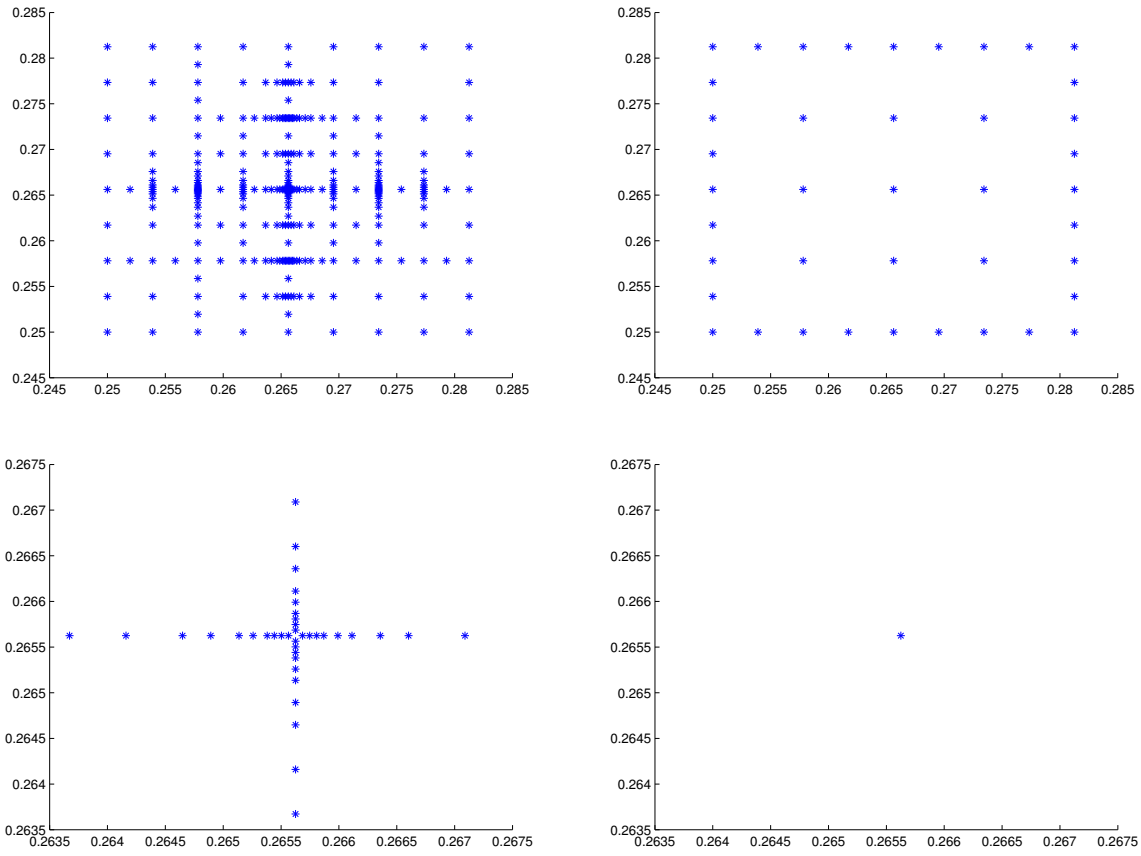

Figure 4: Close view of the points associated with the classical and family selective interpolants applied to function (3.1). The plots focus around the points $(0.265625, 0.265625)$, which is included in the family selective interpolant, but missing from the classical one. The two plots on the left correspond to the classical refinement and they show the clustering of points near the missing parent. The plots on the right correspond to the family selective refinement and show the much coarser grid that is sufficient for convergence.

The previous section gave an example of instability due to missing parents. The children of a function cannot resolve its parents, hence, a missing parent can result in a failure to converge. However, parents do not always have a more significant contribution to the accuracy of the interpolant, hence, the second example shows that a family selective refinement technique may result in more points than the classic one. The function (3.2) was specifically chosen to be zero for many of the sample points, however, we expect that functions associated with real world application would not have this behavior or at least the behavior would not be common.

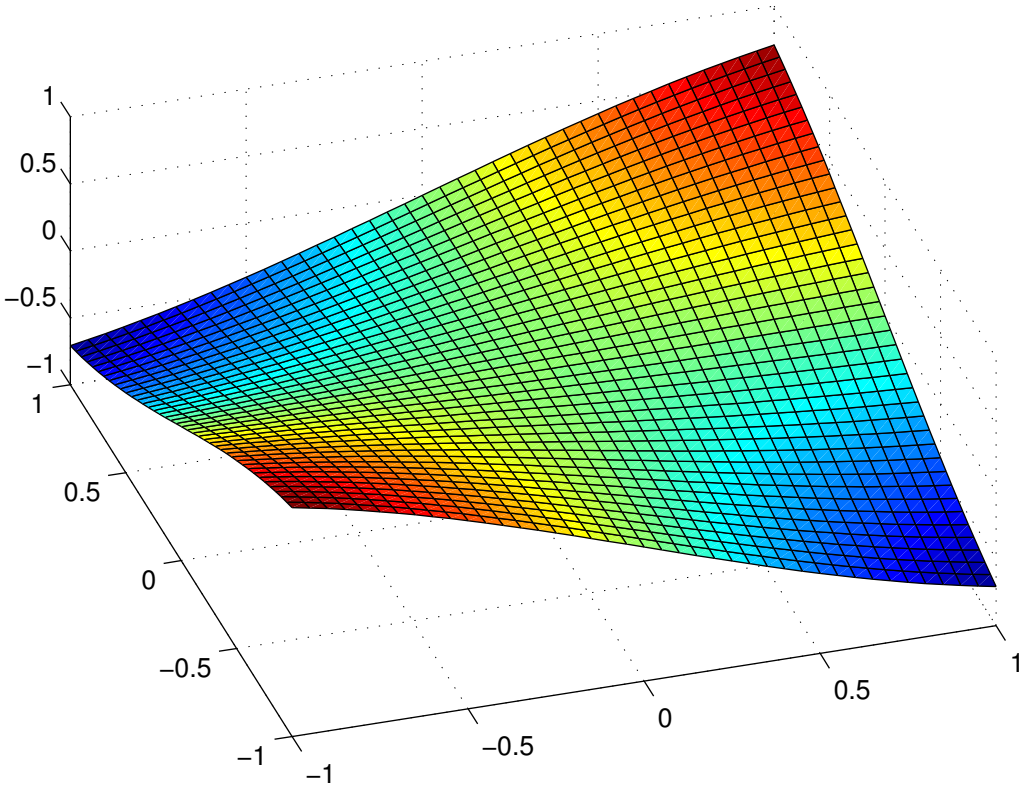

Figure 5: Surface plot of funciton (3.2) using dense grid on the entire domain $[-1,1]^2$. The function is zero along both axis and hence the associated points do not contributite to the accuracy of the interpolant. However, the family selective algorithm (2.14) will include more parents and many of those coinside with the axis.

3.3 Direction Selective Improvement

In order to test the direction selective refinement, we consider the function

$$f(x,y) = exp(-x^2 - y^2) \tag{3.3}$$

which is plotted on Figure 7. Despite the fact that the function is globally isotropic (i.e. it is globally invariant under rotation), (3.3) is locally anisotropic. Furthermore, sparse grids add a lot of points along straight lines and even for a fully isotropic function the direction selective refinement (2.15) may result in improvement over the classical refinement (2.13).

We use the one dimensional hierarchy described (2.1) with linear functions basis (2.2). We use $l = 3$ and $\epsilon = 10^{-3}$ and we compare the results from applying the hierarchical adaptive interpolation algorithms with classical (2.13) and direction selective (2.15) refinement. The classical refinement converges to a grid with 421 points, while the direction refinement requires only 397. Figure 8 shows the two final grids. We observe the clustering of points in both cases, however,

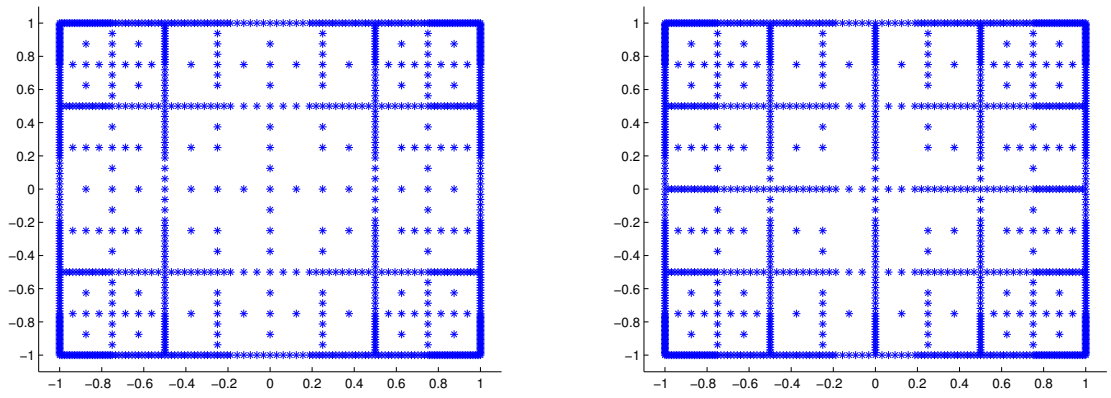

Figure 6: The resulting adaptive grids form applyting classical (2.13) (left) and family selective (2.14) (right) algorithms to the function (3.2). The classical grid contain $1,049$ points, while the family selective grid has $1,165$. We observe that the family selective algorithm adds a large number of points on the main axis, however, those points do not contribute to the accuracy of the interpolant.

the grid corresponding to the direction selective refinement reduces the cluster in several regions along the boundary. The amount of points that is saved depends on the function, order of the basis, initial level and refinement tolerance. However, 5% saving is significant and many other examples exhibit an bigger savings.

3.4 Family Direction Selective Higher Dimensions

We want to test our method for a function of dimensions higher than 2. We expect that for a higher number of dimensions, the anisotropic refinement would result in even more savings. Consider the function

$$f(\mathbf{x}) = \frac{1}{1 + exp(-0.1\|\mathbf{x}\|)}, \tag{3.4}$$

where $\mathbf{x} \in \mathbb{R}^d$ and $\|\mathbf{x}\|$ indicates the Euclidean norm.

We use the one dimensional hierarchy described (2.1) with linear functions basis (2.4). We use $l = 5$ and $\epsilon = 10^{-3}$. We test the different refinement techniques for two values of $d = 4$ and $d = 5$. We compare the classical refinement (2.13) to the fully adaptive (2.16). For $d = 4$, the fully adaptive scheme converges with only $3,401$ samples as opposed to $9,249$, which is a 63% improvement. When we increase the dimension to $d = 5$, the fully converges scheme converges with $19,145$ as opposed to $53,425$ samples, which is an bigger savings of 64%. This demonstrates the massive advantage that the fully adaptive scheme can have for some problems.

A summary of all of the numerical tests is shown on Table 2. We note that while there is no scheme that converges with fewer samples in all cases, the fully adaptive scheme (2.16) requires fewer samples in almost all cases.

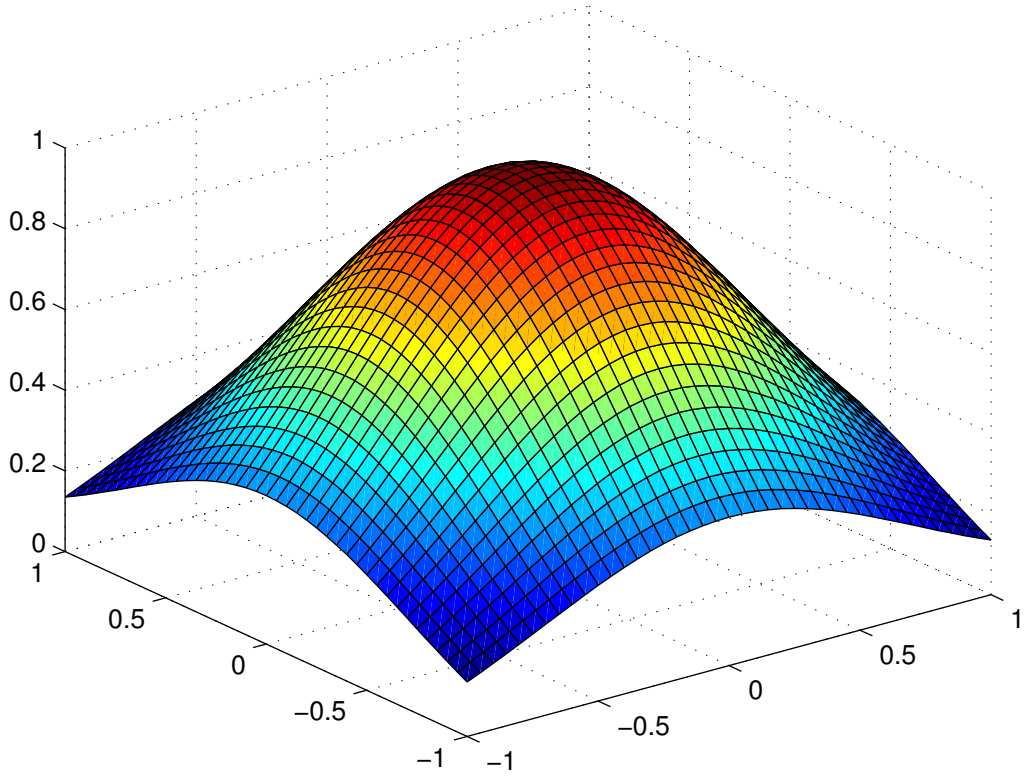

Figure 7: Surface plot of funciton (3.3) using dense grid on the entire domain $[-1,1]^2$. Despite the fact that the funciton is globally istorpic (i.e. it is globlly invariant under rotation), it is locally anisotropic.

Function	l	ϵ	Basis	Number of Points			
				Classic	Family	Direction	FDS
(3.1)	8	10^{-4}	Cubic	∞	9937	∞	8157
(3.2)	4	10^{-4}	Linear	1033	1165	889	1021
(3.2)	4	10^{-6}	Linear	12745	14245	11769	13269
(3.3)	3	10^{-3}	Linear	421	421	397	397
(3.3)	3	10^{-4}	Linear	1657	1657	1433	1433
(3.3)	3	10^{-4}	Quadratic	561	561	545	545
(3.3)	3	10^{-4}	Cubic	329	329	313	313
(3.4) with $d=4$	5	10^{-3}	Cubic	9249	9249	3401	3401
(3.4) with $d=5$	5	10^{-3}	Cubic	53425	53425	19145	19145

Table 2: Summary of the number of tests performed on all the functions. While there is no scheme that is always better, the fully adaptive family-direction-selective scheme (FDS) most often requires the least number of samples.

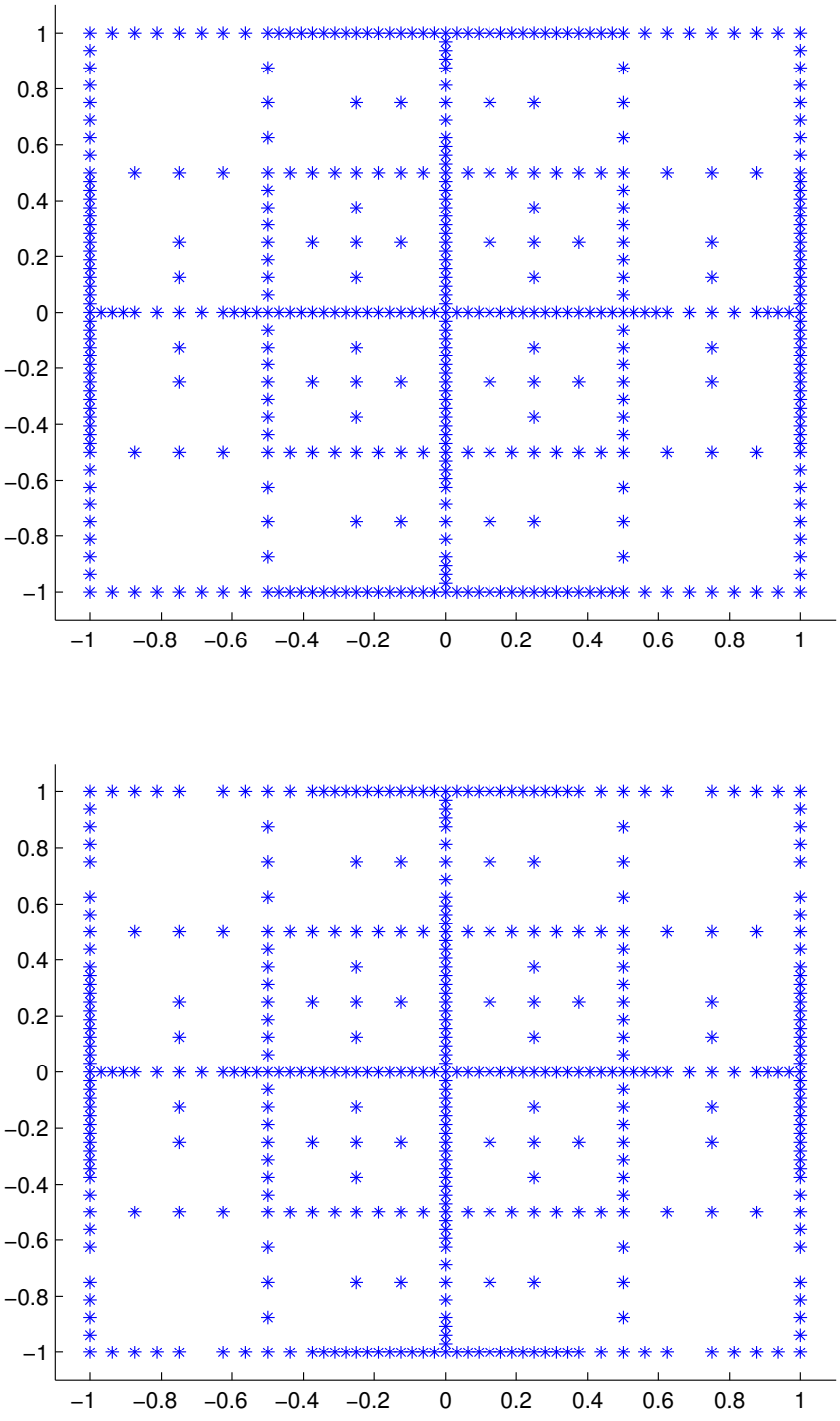

Figure 8: The resulting adaptive grids form applyting classical (2.13) (left) and direction selective (2.15) (right) algorithms to function (3.3). The classical grid contain 421 points, while the direction selective grid has 397. We observe that the direction selective scheme does not add as many point to the already clusteed lines, mainly along the boundary.

4 Conclusions

We consider the problem of multidimensional adaptive hierarchical interpolation. We present the construction of a multidimensional interpolation rule that is induced by a one dimensional rule via sparse selection of the tensor products. We observe that the multidimensional hierarchy has more structure than the original one dimensional rule, which allows for a wide variety of refinement techniques beyond the classical isotropic selection of children. We present refinement schemes that refine anisotropically as well as adding samples from multiple levels of the hierarchy.

We present a number of numerical examples of simple functions that are designed to demonstrate the different advantages and disadvantages of each refinement scheme. There is no scheme that will always converge to an interpolant with smallest number of samples, however, the fully adaptive scheme wins on all but a small number of carefully crafted examples. In practice, such functions would be at best rare, hence, the fully adaptive family-direction-selective scheme is the most competitive method.

REFERENCES

[1] S. ACHARJEE AND N. ZABARAS, *A non-intrusive stochastic galerkin approach for modeling uncertainty propagation in deformation processes*, Computers & structures, 85 (2007), pp. 244–254. 2

[2] I. BABUSKA, R. TEMPONE, AND G. E. ZOURARIS, *Galerkin finite element approximations of stochastic elliptic partial differential equations*, SIAM Journal on Numerical Analysis, 42 (2004), pp. 800–825. 2

[3] F. E. BENTH AND J. GJERDE, *Convergence rates for finite elementapproximations of stochastic partial differential equations*, Stochastics: An International Journal of Probability and Stochastic Processes, 63 (1998), pp. 313–326. 2

[4] P. J. DAVIS AND P. RABINOWITZ, *Methods of numerical integration*, Courier Dover Publications, 2007. 2

[5] M. K. DEB, I. M. BABUŠKA, AND J. T. ODEN, *Solution of stochastic partial differential equations using galerkin finite element techniques*, Computer Methods in Applied Mechanics and Engineering, 190 (2001), pp. 6359–6372. 2

[6] C. FINNEY, M. STOYANOV, S. PANNALA, C. DAW, R. WAGNER, K. EDWARDS, G. WEBSTER, AND J. GREEN, *Application of high-performance computing for simulating the unstable dynamics of dilute, spark-ignited combustion*, in Proceedings of the 2012 International Conference on Theory and Applications of Nonlinear Dynamics, 2012. 2

[7] G. S. FISHMAN, *Monte Carlo: concepts, algorithms, and applications*, vol. 1196, Springer New York, 1996. 2

[8] T. GERSTNER AND M. GRIEBEL, *Numerical integration using sparse grids*, Numerical algorithms, 18 (1998), pp. 209–232. 2

[9] ——, *Dimension–adaptive tensor–product quadrature*, Computing, 71 (2003), pp. 65–87. 3

[10] R. GHANEM AND P. D. SPANOS, *Stochastic finite elements: a spectral approach*, Dover-Publications. com, 2003. 2

[11] M. GRIEBEL, *Adaptive sparse grid multilevel methods for elliptic pdes based on finite differences*, Computing, 61 (1998), pp. 151–179. 3, 6, 10

[12] M. GUNZBURGER, C. TRENCHEA, AND C. WEBSTER, *A generalized stochastic collocation approach to constrained optimization for random data identification problems*, Tech. Rep. ORNL/TM-2012/185, Oak Ridge National Laboratory, 2012. 2

[13] M. GUNZBURGER, C. WEBSTER, AND G. ZHANG, *An adaptive wavelet stochastic collocation method for irregular solutions of partial differential equations with random input data*, Tech. Rep. ORNL/TM-2012/186, Oak Ridge National Laboratory, 2012. 3

[14] J. C. HELTON AND F. J. DAVIS, *Latin hypercube sampling and the propagation of uncertainty in analyses of complex systems*, Reliability Engineering & System Safety, 81 (2003), pp. 23–69. 2

[15] M. KLEIBER AND D. H. TRAN, *The stochastic finite element method: basic perturbation technique and computer implementation*, Wiley New York, 1992. 2

[16] A. KLIMKE AND B. WOHLMUTH, *Algorithm 847: Spinterp: piecewise multilinear hierarchical sparse grid interpolation in matlab*, ACM Transactions on Mathematical Software (TOMS), 31 (2005), pp. 561–579. 3

[17] M. LOEVE, *Probability theory, vol. ii*, Graduate texts in mathematics, 46 (1978), pp. 0–387. 2

[18] X. MA AND N. ZABARAS, *An adaptive hierarchical sparse grid collocation algorithm for the solution of stochastic differential equations*, Journal of Computational Physics, 228 (2009), pp. 3084–3113. 3

[19] H. NIEDERREITER, *Quasi-Monte Carlo Methods*, Wiley Online Library, 1992. 2

[20] F. NOBILE, R. TEMPONE, AND C. G. WEBSTER, *An anisotropic sparse grid stochastic collocation method for partial differential equations with random input data*, SIAM Journal on Numerical Analysis, 46 (2008), pp. 2411–2442. 2

[21] F. NOBILE, R. TEMPONE, AND C. G. WEBSTER, *A sparse grid stochastic collocation method for partial differential equations with random input data*, SIAM Journal on Numerical Analysis, 46 (2008), pp. 2309–2345. 2

[22] S. A. SMOLYAK, *Quadrature and interpolation formulas for tensor products of certain classes of functions*, Dokl. Akad. Nauk SSSR, 4 (1963), pp. 240–243 (English translation). 2, 10

[23] A. H. STROUD AND D. SECREST, *Gaussian quadrature formulas*, vol. 374, Prentice-Hall Englewood Cliffs, NJ, 1966. 2

[24] G. ZHANG AND M. GUNZBURGER, *Error analysis of a stochastic collocation method for parabolic partial differential equations with random input data*, SIAM Journal on Numerical Analysis, 50 (2012), pp. 1922–1940. 2

[25] G. ZHANG, M. GUNZBURGER, AND W. ZHAO, *A sparse grid method for multi-dimensional backward stochastic differential equaitons*, Journal of Computational Mathematics, 31 (2013), pp. 221–248. 2


OAK RIDGE NATIONAL LABORATORY
MANAGED BY UT-BATTELLE FOR THE DEPARTMENT OF ENERGY