Algorithm 847: `spinterp`: Piecewise Multilinear Hierarchical Sparse Grid Interpolation in MATLAB

ANDREAS KLIMKE and BARBARA WOHLMUTH
University Stuttgart

To recover or approximate smooth multivariate functions, sparse grids are superior to full grids due to a significant reduction of the required support nodes. The order of the convergence rate in the maximum norm is preserved up to a logarithmic factor. We describe three possible piecewise multilinear hierarchical interpolation schemes in detail and conduct a numerical comparison. Furthermore, we document the features of our sparse grid interpolation software package `spinterp` for MATLAB.

Categories and Subject Descriptors: G.1.1 [**Numerical Analysis**]: Interpolation; G.1.2 [**Numerical Analysis**]: Approximation; G.4 [**Mathematical Software**]: *Documentation*

General Terms: Algorithms, Documentation

Additional Key Words and Phrases: Multivariate interpolation, sparse grids, Smolyak algorithm

1. INTRODUCTION

The interpolation problem considered with sparse grid interpolation is an optimal recovery problem (i.e., the selection of points such that a smooth function can be approximated with a suitable interpolation formula). Depending on the characteristics of the function to interpolate (degree of smoothness, periodicity), various interpolation techniques based on sparse grids exist (e.g., Barthelmann et al. [2000]; Schreiber [2000]; Sprengel [1997, 1998]). All of them employ Smolyak's construction [Smolyak 1963], which forms the basis of all sparse grid methods. With Smolyak's famous method, well-known univariate interpolation formulas are extended to the multivariate case by using tensor products in a special way. As a result, one obtains a powerful interpolation method that requires significantly fewer support nodes than conventional interpolation on a full grid. The points comprising the multidimensional sparse grid are selected in a predefined fashion. The difference in the number of required points can

Authors' address: Institute of Applied Analysis and Numerical Simulation, University Stuttgart, 70569 Stuttgart, Germany; email: `{klimke,wohlmuth}@ians.uni-stuttgart.de`.

be several orders of magnitude with increasing problem dimension. The most important property of the method is the fact that the asymptotic error decay of full grid interpolation with increasing grid resolution is preserved up to a logarithmic factor. An additional benefit of the method is its hierarchical structure, which can be used to obtain an estimate of the current approximation error. Thus, one can easily develop an interpolation algorithm that terminates automatically when a desired accuracy is reached.

In the following, we present an implementation of piecewise multilinear hierarchical sparse grid interpolation. We compare three different sparse grid types with respect to their grid structure and matching piecewise multilinear basis functions. Suitable agorithms are given and analyzed. We then perform numerical tests with respect to the error decay for a set of test functions. The last section gives a brief discussion of the user interface of our sparse grid interpolation software package `spinterp` for MATLAB.

Interpolation is not the only application area of sparse grids. Bungartz and Griebel [2004] provided an excellent overview of the foundations as well as current developments in this research field.

2. MULTILINEAR SPARSE GRID INTERPOLATION

2.1 Smolyak's Algorithm

This brief description of Smolyak's construction for multivariate interpolation adheres to the notation given in Barthelmann et al. [2000]. We would like to approximate smooth functions $f : [0,1]^d \to \mathbb{R}$ using a finite number of support nodes. In the one-dimensional case, an interpolation formula is given by

$$U^i(f) = \sum_{x^i \in X^i} a_{x^i} \cdot f(x^i),$$

with the set of support nodes $X^i = \{x^i \mid x^i \in [0,1]\}$, $\#X^i = m_i$, and the basis functions $a_{x^i} \in C([0,1])$, $a_{x^i}(x^i) = 1$, $a_{x^i}(y^i) = 0 \ \forall\ y^i \in X^i, x^i \neq y^i$, $i \in \mathbb{N}$. To obtain an interpolation formula for the multivariate case, one can use the tensor product formula

$$(U^{i_1} \otimes \cdots \otimes U^{i_d})(f) = \sum_{x^{i_1} \in X^{i_1}} \cdots \sum_{x^{i_d} \in X^{i_d}} (a_{x^{i_1}} \otimes \cdots \otimes a_{x^{i_d}}) \cdot f(x^{i_1}, \ldots, x^{i_d}). \qquad (1)$$

However, the above formula requires a very high number of $m_{i_1} \cdots m_{i_d}$ support nodes, which are sampled on the full grid. To reduce the number of support nodes while maintaining the approximation quality of the interpolation formula up to a logarithmic factor, Smolyak's construction is used. With $U^0 = 0$, $\Delta^i = U^i - U^{i-1}$, $|\mathbf{i}| = \mathbf{i_1} + \cdots + \mathbf{i_d}$ for $\mathbf{i} \in \mathbb{N}^d$, and $q \geq d, q \in \mathbb{N}$, the Smolyak algorithm is given by

$$A_{q,d}(f) = \sum_{|\mathbf{i}| \leq q} (\Delta^{i_1} \otimes \cdots \otimes \Delta^{i_d})(f) = A_{q-1,d}(f) + \underbrace{\sum_{|\mathbf{i}| = q} (\Delta^{i_1} \otimes \cdots \otimes \Delta^{i_d})(f)}_{\Delta A_{q,d}(f)}, \qquad (2)$$

with $A_{d-1,d} = 0$. From (2), we observe that we can increase the accuracy of the interpolation based on Smolyak's construction without having to discard previous results. Most importantly, to compute $A_{q,d}(f)$, only the function values at the sparse grid

$$H_{q,d} = \bigcup_{q-d+1 \le |\mathbf{i}| \le q} (X^{i_1} \times \cdots \times X^{i_d}) \tag{3}$$

are needed. One should select the sets X^i in a nested fashion such that $X^i \subset X^{i+1}$ to obtain many recurring points with increasing q. With $X^0 = \emptyset$, $X^i_\Delta = X^i \backslash X^{i-1}$, one can rewrite (3) to

$$H_{q,d} = \bigcup_{|\mathbf{i}| \le q} \left(X^{i_1}_\Delta \times \cdots \times X^{i_d}_\Delta\right) = H_{q-1,d} \cup \Delta H_{q,d}, \tag{4}$$

with

$$\Delta H_{q,d} = \bigcup_{|\mathbf{i}| = q} \left(X^{i_1}_\Delta \times \cdots \times X^{i_d}_\Delta\right), \tag{5}$$

$H_{d-1,d} = \emptyset$, which is more convenient for a successive refinement of the grid with increasing parameter q. For later use, we denote the cardinality of a subset $(X^{i_1}_\Delta \times \cdots \times X^{i_d}_\Delta)$ by $n(\mathbf{i})$.

2.2 Selecting a Suitable Sparse Grid

For multilinear interpolation, it is natural to select sparse grids with sets X^i, $i \in \mathbb{N}$ of equidistant nodes. We have examined three possibilities to construct the sparse grid, namely:

(1) The "classical" maximum- or L_2-norm-based sparse grid H^{M}, including the boundary, as thoroughly discussed in Bungartz [1992, 1998]. The points x^i_j comprising the set of support nodes X^i are defined as

$$m_i = 2^i + 1,$$
$$x^i_j = (j-1)/(m_i - 1) \text{ for } j = 1, \ldots, m_i, \text{ and } i \ge 1.$$

(2) The maximum-norm-based sparse grid, but excluding the points on the boundary, denoted by H^{NB}. Now the x^i_j are defined as

$$m_i = 2^i - 1,$$
$$x^i_j = j/(m_i + 1) \text{ for } j = 1, \ldots, m_i.$$

(3) The Clenshaw-Curtis-type sparse grid H^{CC} with equidistant nodes, as described in Novak and Ritter [1996] and Schreiber [2000]. Although the original article [Clenshaw and Curtis 1960] used Chebyshev-distributed nodes, we adhere to this grid name due to its association in the literature with the grid structure (resulting from the sequence $m_i = 2^{i-1} + 1$). Here, the x^i_j are

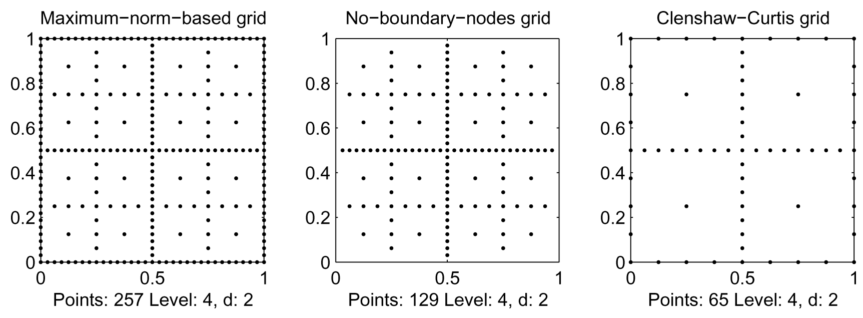


Fig. 1. Sparse grids $H^{\mathrm{M}}_{6,2}$ (left), $H^{\mathrm{NB}}_{6,2}$ (middle), and $H^{\mathrm{CC}}_{6,2}$ (right).

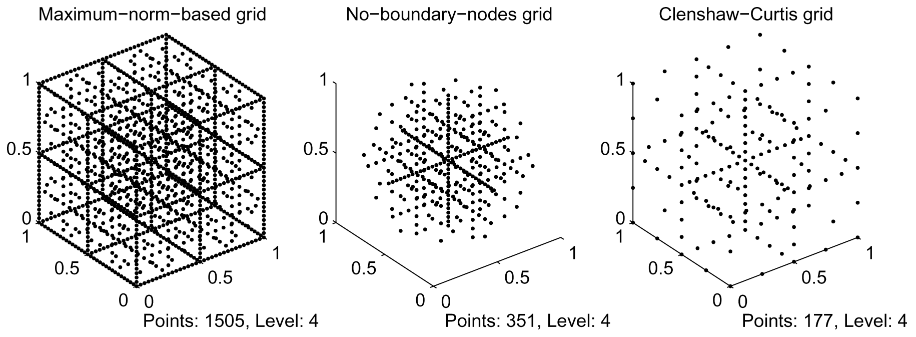


Fig. 2. Sparse grids $H^{\mathrm{M}}_{7,3}$ (left), $H^{\mathrm{NB}}_{7,3}$ (middle), and $H^{\mathrm{CC}}_{7,3}$ (right).

defined as

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

$$x^i_j = \begin{cases} (j-1)/(m_i - 1) & \text{for } j = 1, \dots, m_i \text{ if } m_i > 1, \\ 0.5 & \text{for } j = 1 \text{ if } m_i = 1. \end{cases}$$

All three resulting sets of points fulfill the important property $X^i \subset X^{i+1}$, and therefore also $H_{q,d} \subset H_{q+1,d}$. Figure 1 illustrates the grids $H^{\mathrm{M}}_{6,2}$, $H^{\mathrm{NB}}_{6,2}$, and $H^{\mathrm{CC}}_{6,2}$ for $d = 2$. Figure 2 illustrates the grids $H^{\mathrm{M}}_{7,3}$, $H^{\mathrm{NB}}_{7,3}$, and $H^{\mathrm{CC}}_{7,3}$ for $d = 3$. The number of grid points grows much faster with increasing q, d for H^{M}. The number of points of the Clenshaw-Curtis grid H^{CC} increases the slowest. To further illustrate the growth of the number of nodes with q, d depending on the chosen grid type, we have included Table I. Note that the grid H^{M} is not suited for higher-dimensional problems, since at least 3^d support nodes are needed.

Our numerical tests in Section 4 will show that the Clenshaw-Curtis-type grid outperforms the other grid types in most cases (although the asymptotic error decay remains the same). However, depending on the objective function characteristics, another grid type may also perform better occasionally.

Table I. Comparison: Number of Grid Points for Level $n = q - d$

	$d = 2$			$d = 4$			$d = 8$		
n	M	NB	CC	M	NB	CC	M	NB	CC
0	9	1	1	81	1	1	6561	1	1
1	21	5	5	297	9	9	41553	17	17
2	49	17	13	945	49	41	1.9e5	161	145
3	113	49	29	2769	209	137	7.7e5	1121	849
4	257	129	65	7681	769	401	2.8e6	6401	3937
5	577	321	145	20481	2561	1105	9.3e6	31745	15713
6	1281	769	321	52993	7937	2929	3.0e7	141569	56737
7	2817	1793	705	1.3e5	23297	7537	9.1e7	5.8e5	1.9e5

2.3 The Univariate Nodal Basis Functions

To compute $A_{q,d}(f)$, we need to specify the basis functions a of the interpolation formulas $U^i(f)$. We provide the piecewise linear basis functions for each of the three grid types below.

(1) For the maximum-norm-based grid, we define

$$a_{x_j^i}(x) = \begin{cases} 1 - (m_i - 1)\left|x - x_j^i\right|, & \text{if } \left|x - x_j^i\right| < 1/(m_i - 1), \\ 0, & \text{otherwise,} \end{cases}$$

for $j = 1, \dots, m_i$.

(2) For the maximum-norm-based grid without boundary nodes, we define

$a_{x_1^1}(x) = 1 \quad \text{for } i = 1$, and

$$\text{if } j = \begin{cases} 1, & a_{x_1^i}(x) = \begin{cases} 2 - (m_i + 1)x, & \text{if } x < \frac{2}{m_i+1} \\ 0, & \text{otherwise,} \end{cases} \\ m_i, & a_{x_{m_i}^i}(x) = \begin{cases} (m_i + 1)x - m_i + 1, & \text{if } x > \frac{m_i-1}{m_i+1} \\ 0, & \text{otherwise,} \end{cases} \\ \text{otherwise}, & a_{x_j^i}(x) = \begin{cases} 1 - (m_i + 1)\left|x - x_j^i\right|, & \text{if } \left|x - x_j^i\right| < \frac{1}{m_i+1}, \\ 0, & \text{otherwise,} \end{cases} \end{cases}$$

for $i > 1$ and $j = 1, \dots, m_i$.

(3) For the Clenshaw-Curtis grid, we get

$$a_{x_1^1}(x) = 1 \quad \text{for } i = 1,$$

and

$$a_{x_j^i}(x) = \begin{cases} 1 - (m_i - 1) \cdot \left|x - x_j^i\right|, & \text{if } \left|x - x_j^i\right| < 1/(m_i - 1), \\ 0, & \text{otherwise,} \end{cases}$$

for $i > 1$ and $j = 1, \dots, m_i$.

2.4 Accuracy of Piecewise Multilinear Interpolation

We now take a brief look at the approximation quality. An a priori error estimate can be obtained for a d-variate function f if continuous mixed derivatives

$$D^\alpha f = \frac{\partial^{|\alpha|} f}{\partial x_1^{\alpha_1} \cdots x_d^{\alpha_d}},$$

with $\alpha \in \mathbb{N}_0^d$, $|\alpha| = \sum_{i=1}^{d} \alpha_i$, and $\alpha_1, \ldots, \alpha_d \leq 2$, exist, that is, $f \in F$ with

$$F := \{f : [0,1]^d \to \mathbb{R},\ D^\alpha f \in C^0([0,1]^d), \alpha_1, \ldots, \alpha_d \leq 2)\}.$$

According to Bungartz [1998] or Barthelmann et al. [2000], Remark 9, the order of the interpolation error in the maximum norm is then given by

$$\left\| f - A_{q,d}(f) \right\|_\infty = \mathcal{O}\big(N^{-2} \cdot |\log_2 N|^{3\cdot(d-1)}\big), \tag{6}$$

with N denoting the number of grid points of $H_{q,d}^{\mathrm{CC}}$ or $H_{q,d}^{\mathrm{M}}$. Piecewise multilinear approximation on a full grid with $\hat{N}$ grid points is much less efficient, that is, $\mathcal{O}(\hat{N}^{-\frac{2}{d}})$.

2.5 From Univariate Nodal to Multivariate Hierarchical

In this section, we will show how one can obtain a multivariate hierarchical formulation of Smolyak's algorithm of Section 2.1 in explicit form. In this context, the univariate nodal basis functions are transformed into multivariate hierarchical bases. To do this, we use the fact that $U^i(f)$ can exactly represent $U^{i-1}(f)$. This is the case for the piecewise linear basis functions chosen for the Clenshaw-Curtis grid H^{CC} and for the maximum-norm-based grid H^{M} including the boundary.

Remark. The nodal basis functions of the no-boundary-nodes grid H^{NB} violate this assumption close to the boundary. The following construction of the hierarchical bases is therefore not applicable in this particular case. Despite this, the no-boundary-nodes grid performed well in the numerical tests, which led to the decision to include this grid type in our implementation.

By taking advantage of the subset property $X^i \subset X^{i+1}$, we start with the transformation of the univariate nodal basis into the hierarchical one. By definition, we have

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

With $U^i(f) = \sum_{x^i \in X^i} a_{x^i} \cdot f(x^i)$ and $U^{i-1}(f) = U^i(U^{i-1}(f))$, we obtain

$$\begin{aligned}\Delta^i(f) &= \sum_{x^i \in X^i} a_{x^i} \cdot f(x^i) - \sum_{x^i \in X^i} a_{x^i} \cdot U^{i-1}(f)(x^i) \\ &= \sum_{x^i \in X^i} a_{x^i} \cdot (f(x^i) - U^{i-1}(f)(x^i)),\end{aligned}$$

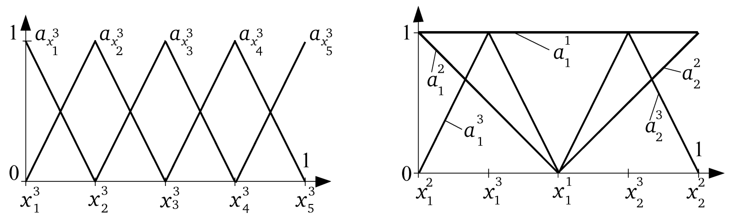


Fig. 3. Nodal basis functions $a_{x_j^3}$, $x_j^3 \in X^3$ (left) and hierarchical basis functions a_j^i with the support nodes $x_j^i \in X_\Delta^i$, $i = 1, \ldots, 3$ (right) for the Clenshaw-Curtis grid.

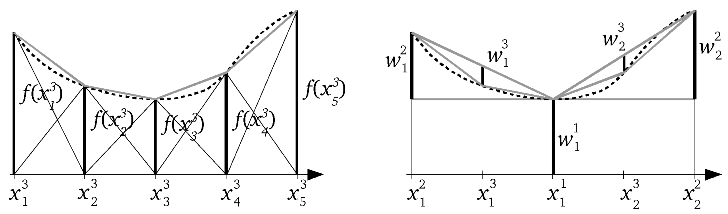


Fig. 4. Nodal versus hierarchical interpolation in one dimension.

and, since $f(x^i) - U^{i-1}(f)(x^i) = 0 \ \ \forall \, x^i \in X^{i-1}$,

$$\Delta^i(f) = \sum_{x^i \in X_\Delta^i} a_{x^i} \cdot (f(x^i) - U^{i-1}(f)(x^i)), \tag{7}$$

recalling $X_\Delta^i = X^i \setminus X^{i-1}$. Clearly, X_Δ^i has $m_i^\Delta = m_i - m_{i-1}$ elements, since $X^{i-1} \subset X^i$. By consecutively numbering the elements in X_Δ^i, and denoting the jth element of X_Δ^i as x_j^i, we can rewrite (7) as

$$\Delta^i(f) = \sum_{j=1}^{m_i^\Delta} a_j^i \cdot \underbrace{\left(f\left(x_j^i\right) - U^{i-1}(f)\left(x_j^i\right)\right)}_{w_j^i}. \tag{8}$$

To simplify the notation, we have replaced $a_{x_j^i}$ by a_j^i. Note that, for all $\Delta^i(f)$, $i > 1$, we only have to consider the contributions of the basis functions belonging to the grid points that have not jet occurred in a previous set X^{i-k}, $1 \leq k \leq i-1$. For a comparison of the nodal and the hierarchical basis functions, see Figure 3. For the construction of the interpolant of a univariate objective function f using nodal basis functions and function values versus using hierarchical basis functions and hierarchical surpluses, see Figure 4.

The tensor product formula (1) and the Smolyak algorithm (2) can now be applied to the Δ^i from Equation (8) to obtain the sparse grid interpolation formula for the multivariate case in hierarchical form. Recalling Equation (2), we have

$$A_{q,d}(f) = A_{q-1,d}(f) + \Delta A_{q,d}(f), \tag{9}$$

with

$$\Delta A_{q,d}(f) = \sum_{|\mathbf{i}|=q} (\Delta^{i_1} \otimes \cdots \otimes \Delta^{i_d})(f), \tag{10}$$

with $A_{d-1,d} = 0$. We can express U^{i-1} in Equation (8) by the following sum by using the telescopic property $\Delta^i = U^i - U^{i-1}$:

$$U^{i-1}(f) = \sum_{i_1=1}^{i-1} \Delta^{i_1}(f).$$

In case of the full grid, we would thus obtain a multivariate extension of an interpolant U^{i-1} with the tensor product formula (1) to

$$(U^{i-1} \otimes \cdots \otimes U^{i-1})(f) = \sum_{i_1=1}^{i-1} \cdots \sum_{i_d=1}^{i-1} (\Delta^{i_1} \otimes \cdots \otimes \Delta^{i_d})(f).$$

However, the Smolyak formula $A_{q-1,d}$ only uses the support nodes and basis functions of the sparse grid with $|\mathbf{i}| \leq q-1$ (between the index i of the univariate interpolation formula $U^i(f)$ and the parameter q of the Smolyak formula $A_{q,d}(f)$ constructed from $U^1(f), \ldots, U^i(f)$, we have the relation $q = i + d - 1$). Therefore, we use the reduced sum

$$A_{q-1,d}(f) = \sum_{|\mathbf{i}| \leq q-1} (\Delta^{i_1} \otimes \cdots \otimes \Delta^{i_d})(f)$$

instead of $(U^{i-1} \otimes \cdots \otimes U^{i-1})(f)$ when constructing $\Delta A_{q,d}(f)$ in (10) from (8). Finally, we obtain the Smolyak algorithm in entirely hierarchical form with (9) and

$$\Delta A_{q,d}(f) = \sum_{|\mathbf{i}|=q} \sum_{\mathbf{j}} \underbrace{\left(a_{j_1}^{i_1} \otimes \cdots \otimes a_{j_d}^{i_d}\right)}_{a_{\mathbf{j}}^{\mathbf{i}}} \cdot \underbrace{\left(f\left(x_{j_1}^{i_1}, \ldots, x_{j_d}^{i_d}\right) - A_{q-1,d}(f)\left(x_{j_1}^{i_1}, \ldots, x_{j_d}^{i_d}\right)\right)}_{w_{\mathbf{j}}^{\mathbf{i}}}, \tag{11}$$

with $\mathbf{j}$ denoting the multi-index $(j_1, \ldots, j_d)$, $j_l = 1, \ldots, m_{i_l}^{\Delta}$, and $l = 1, \ldots, d$. Note that all the required support nodes $\mathbf{x}_{\mathbf{j}}^{\mathbf{i}} = (x_{j_1}^{i_1}, \ldots, x_{j_d}^{i_d})$ of a single level k are included in the set $\Delta H_{k+d,d}$, as defined in Equation (4). Furthermore, it is useful to define

$$w_{\mathbf{j}}^{k,\mathbf{i}} = f\left(\mathbf{x}_{\mathbf{j}}^{\mathbf{i}}\right) - A_{k+d-1,d}\left(\mathbf{x}_{\mathbf{j}}^{\mathbf{i}}\right)$$

as the *hierarchical surpluses* [Bungartz 1998] at level k with $k = |\mathbf{i}| - d$, since $A_{k+d,d}$ corrects $A_{k+d-1,d}$ at the points $\mathbf{x}_{\mathbf{j}}^{\mathbf{i}}$ to the actual value of $f(\mathbf{x}_{\mathbf{j}}^{\mathbf{i}})$. For continuous functions, the hierarchical surpluses tend to zero as the level k tends to infinity (see Figure 4 (right). This makes the hierarchical surpluses a natural candidate for error estimation and control. A two-dimensional illustration of the hierarchical construction scheme is shown in Figure 5.

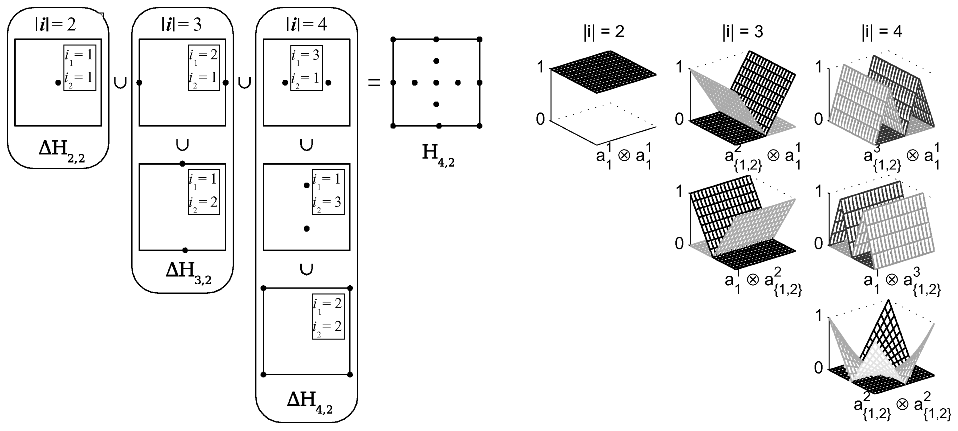


Fig. 5. Two-dimensional example: sparse grid $H^{\mathrm{CC}}_{4,2}$ (left) and corresponding bilinear basis functions of $A_{4,2}$ (right).

3. ALGORITHMS

In this section, we present detailed algorithms for performing sparse grid interpolation. We have abstracted the data structures to ordered sets, since depending on the programming environment, different data types may be available that are suitable to hold the complex data, such as the sets of multiindices, the multivariate sparse grid coordinates, and the hierarchical surpluses. In Section 5, we provide information on the organization of the data in our MATLAB implementation.

3.1 General Remarks

We first state some general remarks on the notation. The # operator denotes the cardinality of a set. The algorithms produce nested sets, that is, a set will often contain subsets. Note that, in this case, the cardinality refers to the *number of subsets only*. To extract subsets or elements from a set, we use the notation $s = (S)_k$, meaning that s is the kth element (or subset) of S. In Algorithms 1, 2, 4, we use the $+ = (- =)$ operator on sets. In this case, we mean that each element *at the lowest level* (i.e., the values) is incremented (decreased) by the right-hand side value at the same position in the set (the sets on the left- and on the right-hand sides are of equal size). Similarly, the min and max operations in Algorithm 1 run over each element at the lowest level.

3.2 Description of the Algorithms

Algorithm 1, spvals, summarizes the procedure of obtaining an increasingly accurate sparse grid interpolant via hierarchical construction. The hierarchical surpluses w^k are used to estimate the current interpolation error, and to formulate the break condition. One obtains the interpolation depth n and an ordered set of hierarchical surpluses $Z_{n,d}$ as the output of the algorithm (actually, $Z_{n,d}$ contains subsets of subsets of hierarchical surpluses), which can then be used

ALGORITHM 1. `spvals(`f`,`d`)`: COMPUTE SET OF HIERARCHICAL SURPLUSES $Z_{n,d}$

```
In:  $d \in \mathbb{N}$.
     $f : [0,1]^d \to \mathbb{R}$: Objective function.
     $\delta_{\text{rel}}$, $\delta_{\text{abs}}$: Relative and absolute error tolerance.
     $n_{\max} \in \mathbb{N}$: Maximum interpolation depth.
Out: $n$: Interpolation depth.
     $Z_{n,d}$: Ordered set of hierarchical surpluses.

Let $w_{\max}^{-1} = \infty,\ y_{\min} = \infty,\ y_{\max} = -\infty, k = 0$.
While $w_{\max}^{k-1} \geq \max[\delta_{\text{rel}} \cdot (y_{\max} - y_{\min}),\ \delta_{\text{abs}}]$ And $k \leq k_{\max}$ Do
    $[\Delta H_{k+d,d}, \Delta S_{k,d}]$ = spgrid($k$,$d$).
    $Y_k = f(\Delta H_{k+d,d})$.        {Evaluate $y = f(\mathbf{x})$ at all points $\mathbf{x} \in \Delta H_{k+d,d}$}
    Let $Z_k = Y_k$.
    If $k > 0$ Then
        For $m = 0$ To $k - 1$
            $Z_k -=$ spvalstep($d$, $Z_m$, $\Delta H_{k+d,d}$, $\Delta S_{k,d}$, $\Delta S_{m,d}$).
        End
    End
    $y_{\max} = \max\big[y_{\max}, y\ \forall\ y \in Y_k\big]$. $y_{\min} = \min\big[y_{\min}, y\ \forall\ y \in Y_k\big]$.
    $w_{\max}^k = \max[w^k\ \forall\ w^k \in Z_k]$.
    $k = k + 1$.
End
Let $n = k - 1$.
Let $Z_{n,d} = \{Z_0, \ldots, Z_n\}$.
```

ALGORITHM 2. `spvalstep(`d`,`Z^{old}`,`ΔH^{new}`,`ΔS^{new}`,`ΔS^{old}`)`: SUBROUTINE OF `spvals`

```
In:  $d \in \mathbb{N}$.
     $Z^{\text{old}}$: Ordered set of hierarchical surpluses.
     $\Delta H^{\text{new}}$: Ordered set of sparse grid points.
     $\Delta S^{\text{new}}$, $\Delta S^{\text{old}}$: Ordered sets of multi-indices.
Out: $\Delta Y$: Ordered set of interpolated values.

Let $m_{\text{old}} = \#\Delta S^{\text{old}}$, $m_{\text{new}} = \#\Delta S^{\text{new}}$.
Let $\Delta Y = \{(\Delta Y)_1, \ldots, (\Delta Y)_{m_{\text{new}}}\}$,        {initialize $\Delta Y$ to contain all zeros}
    with $(\Delta Y)_j = \{0, \ldots, 0\}$, $\#(\Delta Y)_j = \#(\Delta H^{\text{new}})_j$, $j = 1, \ldots, m_{\text{new}}$.
For $l_{\text{old}} = 1$ To $m_{\text{old}}$ Do
    Let $\mathbf{i}^{\text{old}} = (\Delta S^{\text{old}})_{l_{\text{old}}}$.
    For $l_{\text{new}} = 1$ To $m_{\text{new}}$ Do
        Let $\mathbf{i}^{\text{new}} = (\Delta S^{\text{new}})_{l_{\text{new}}}$.
        If $i_j^{\text{old}} \leq i_j^{\text{new}}\ \forall\ j \in \{1, \ldots, d\}$ Then
            $(\Delta Y)_{l_{\text{new}}} +=$ spinterpstep($d$,$\{(Z^{\text{old}})_{l_{\text{old}}}\}$,$\{(\Delta H^{\text{new}})_{l_{\text{new}}}\}$,$\{\mathbf{i}^{\text{old}}\}$).
        End
    End
End
```

by `spinterp` to evaluate the corresponding interpolant $A_{n+d,d}(f)$. Note that the sparse grid coordinates need not be stored.

Algorithm 2, `spvalstep`, is called multiple times by `spvals` to compute the hierarchical surpluses by subtracting the interpolated values $A_{k-1+d,d}(f)(\mathbf{x}_{\mathbf{j}}^{\mathbf{i}})$

ALGORITHM 3. spgrid(n,d): COMPUTE THE SET OF SPARSE GRID POINTS $\Delta H_{n+d,d}$

```
In:  $n \in \mathbb{N}^0$, $d \in \mathbb{N}$.
Out: $\Delta H_{n+d,d}$: Ordered set of points.
     $\Delta S_{n,d}$: Ordered set of according indices.

Let $\Delta S_{n,d} = \{\mathbf{i} \in \mathbb{N}^d \mid |\mathbf{i}| = n + d, |\mathbf{i}| = i_1 + \cdots + i_d\}$.
Let $(\Delta S_{n,d})_k$ denote the $k$th element of $\Delta S_{n,d}$.
Let $m = \#\Delta S_{n,d}$.
For $k = 1$ To $m$ Do
    Let $\mathbf{i} = (\Delta S_{n,d})_k$.
    Let $\mathbf{X}_k = X_\Delta^{i_1} \times \cdots \times X_\Delta^{i_d}$.
End
Let $\Delta H_{n+d,d} = \{\mathbf{X}_1, \ldots, \mathbf{X}_m\}$.
```

ALGORITHM 4. spinterp$(d, n, Z_{n,d}, P)$: EVALUATE THE INTERPOLANT $A_{n+d,d}(f)$

```
In:  $d \in \mathbb{N}$. $n \in \mathbb{N}^0$: Interpolation depth.
     $Z_{n,d}$: Ordered set of hierarchical surpluses.
     $P$: Set of points to interpolate.
Out: $Y$: Ordered set of interpolated values $Y = A_{d+n,d}(f)(P)$.

Let $Y = \{0, \ldots, 0\}$ with $\#Y = \#P$.
For $k = 0$ To $n$ Do
    Let $\Delta S_{k,d} = \{\mathbf{i} \in \mathbb{N}^d \mid |\mathbf{i}| = k + d, |\mathbf{i}| = i_1 + \cdots + i_d\}$.
    $Y += $ spinterpstep$(d, (Z_{n,d})_k, P, \Delta S_{k,d})$.
End
```

at the new grid points $\mathbf{x}_\mathbf{j}^\mathbf{i} \in \Delta H_{k+d,d}$ from the function values $f(\mathbf{x}_\mathbf{j}^\mathbf{i})$, according to Equation (11). The test in the if-then clause assures that subdomains where the interpolated values at the points $\mathbf{x}_\mathbf{j}^\mathbf{i}$ are known to be zero are omitted.

Algorithm 3, spgrid, explicitly constructs the set of sparse grid points $\Delta H_{n+d,d}$ according to Equation (5), which is required by spvals to evaluate the objective function at these points. This function also returns the corresponding set of multi-indices.

Algorithm 4, spinterp, evaluates the interpolant $A_{n+d,d}(f)$ using the set of hierarchical surpluses $Z_{n,d}$ computed by spvals. The subroutine spinterpstep performs the actual work. Note that we recompute the sets of indices $\Delta S_{k,d}$ here instead of retaining the index sets in memory after the construction of the hierarchical surpluses using spvals, since this is not significantly affecting the computational cost.

Finally, Algorithm 5, spinterpstep, computes interpolated values for the given (sub)sets of hierarchical surpluses Z_k with corresponding index sets $\Delta S_{k,d}$. In the presented form of spinterp, the subfunction spinterpstep computes the interpolated values for all $\mathbf{i}$ with $|\mathbf{i}| = k + d$ at once. This proved to be more efficient, especially in an interpreted environment such as MATLAB, since the number of function calls is not too high. This is due to the fact that, with increasing d, the cardinality of $\Delta S_{k,d}$ can become quite large. The total number

ALGORITHM 5. spinterpstep(d, Z, P, ΔS): SUBROUTINE OF spinterp

```
In:  $d \in \mathbb{N}$.
     $Z$: Ordered set of hierarchical surpluses.
     $P$: Set of points to interpolate.
     $\Delta S$: Ordered set of indices.
Out: $\Delta Y$: Ordered set of interpolated values.

Let $\Delta Y = \{0, \dots, 0\}$ with $\#\Delta Y = \#P$.
Let $m = \#\Delta S$.
For $l = 1$ To $m$ Do
    Let $\mathbf{i} = (\Delta S)_l$.
    For Each $p_r \in P, r = 1, \dots, \#P$ Do
        For Each $w_{\mathbf{j}}^{\mathbf{i}} \in (Z)_l$ with $p_r \in \mathrm{supp}(a_{\mathbf{j}}^{\mathbf{i}})$ Do
            $(\Delta Y)_r += a_{\mathbf{j}}^{\mathbf{i}}(p_r) \cdot w_{\mathbf{j}}^{\mathbf{i}}$.
        End
    End
End
```

of index sets of an interpolant $A_{n+d,d}(f)$ can be easily computed by the formula

$$S_{n,d} = \sum_{k=0}^{n} \#\Delta S_{k,d} = \binom{n+d}{d}. \qquad (12)$$

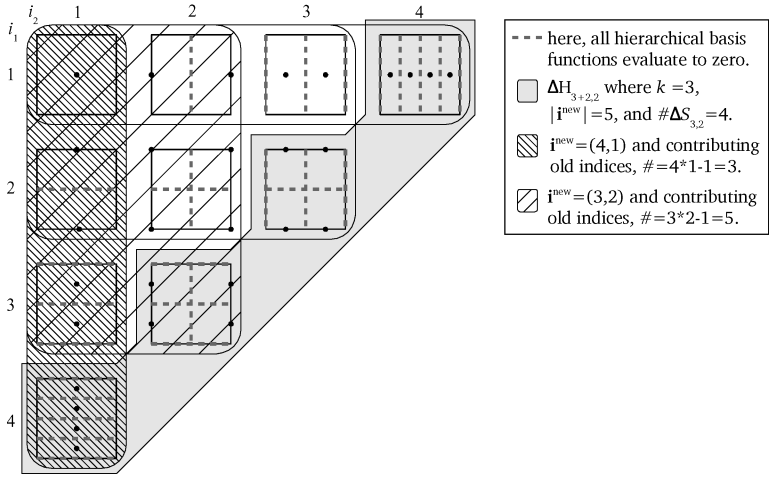
3.3 Computational Cost

In the following, we briefly address the computational complexities of computing the hierarchical surpluses with spvals, as well as evaluating the sparse grid interpolant with spinterp.

3.3.1 *Computational Complexity of spinterpstep.* Both spvals and spinterp require a sequence of calls to spinterpstep. Therefore, we assess the complexity of this function first. The innermost loop can be performed in $\mathcal{O}(d)$ operations in case of piecewise linear basis functions, since evaluation of a d-variate basis function takes $\mathcal{O}(d)$ operations, and there is only one basis function $a_{\mathbf{j}}^{\mathbf{i}}$ with $p_r \in \mathrm{supp}(a_{\mathbf{j}}^{\mathbf{i}})$ for all $w_{\mathbf{j}}^{\mathbf{i}} \in (Z_k)_l$. This holds for all $\mathbf{i}$ for the grids H^{CC}, H^{NB}, and in case of H^{M}, for all $\mathbf{i}$ with $i_1, \dots, i_d > 1$. Thus, considering the outer loops, the computational complexity is given by

$$\mathrm{cost}(\texttt{spinterpstep}, k, d) = \mathcal{O}(\#P \cdot \#\Delta S_{k,d} \cdot d). \qquad (13)$$

3.3.2 *Computational Complexity of spvals.* Determining the *hierarchical surpluses* $w_{\mathbf{j}}^{\mathbf{i}}$ at the sparse grid points is computationally more involved than applying the Smolyak formula directly to the nodal representation from Barthelmann et al. [2000], Equation (3), which requires the *function values* at the sparse grid points only. However, the hierarchical surpluses allow the reliable assessment of the current interpolation error, which is crucial for the automatic termination of the spvals algorithm provided that the requested error tolerance is satisfied. Furthermore, evaluation of the interpolant can be performed rapidly.



i_2
i_1
1
2
3
4
here, all hierarchical basis functions evaluate to zero.
ΔH$_{3+2,2}$ where k =3, |$\mathbf{i}^{\text{new}}$|=5, and #Δ$S_{3,2}$=4.
$\mathbf{i}^{\text{new}}$=(4,1) and contributing old indices, #=4*1-1=3.
$\mathbf{i}^{\text{new}}$=(3,2) and contributing old indices, #=3*2-1=5.

Fig. 6. Multi-indices and sparse grid points of $H^{\text{CC}}_{5,2}$.

Let us start with analyzing `spvalstep` which is called multiple times by `spvals`. The outer loop runs over all old multi-indices $\mathbf{i}^{\text{old}}$ at level k contained in the set $\Delta S_{k,d}$. The inner loop runs over all new multi-indices $\mathbf{i}^{\text{new}}$ at level n from $\Delta S_{n,d}$, but only for old indices with all components smaller than the components of the new indices, the method `spinterpstep` must be called, since, due to the hierarchical scheme, all basis functions $a^{\mathbf{i}^{\text{old}}}_{\mathbf{j}}$ evaluate to zero at the grid points associated with the index set $\mathbf{i}^{\text{new}}$. If one considers the loop calling `spvalstep`, one finds that the number of calls $\Delta C_{k,d}$ to `spinterpstep` during a single step k of the while-loop of `spvals` is given by

$$\Delta C_{k,d} = \sum_{j=1}^{\#\Delta S_{k,d}} \left(\prod_{l=1}^{d} i_l^{\text{new}} - 1 \right), \quad \text{with } \mathbf{i}^{\text{new}} = (\Delta S_{k,d})_j .$$

Consider Figure 6 for an illustration. The total number of calls $\Delta C_{3,2}$ is given by $\sum_{j=1}^{4} (\prod_{l=1}^{2} i_l^{\text{new}} - 1) = (4 \cdot 1 - 1) + (3 \cdot 2 - 1) + (2 \cdot 3 - 1) + (1 \cdot 4 - 1) = 16$.

Finally, to determine the total cost of the calls to `spvals`, we must consider the number of points that `spinterpstep` must interpolate in each call, as well as the problem dimension d according to Equation (13). Since the number of points depends on the index set $\mathbf{i}^{\text{new}}$, we denote it by $n(\mathbf{i}^{new})$ (see Equation (5)). Furthermore, we consider the outermost while loop in `spvals`, which runs over the indices $k = 0, \ldots, n$, and obtain

$$\text{cost}(\texttt{spvals}, n, d) = \mathcal{O}\left(d \cdot \sum_{k=0}^{n} \sum_{j=1}^{\#\Delta S_{k,d}} n(\mathbf{i}^{\text{new}}) \cdot \left(\prod_{l=1}^{d} i_l^{\text{new}} - 1 \right) \right), \tag{14}$$

with $\mathbf{i}^{\text{new}} = (\Delta S_{k,d})_j$. Note that this cost does not include the cost for the required $\#H_{n+d,d}$ evaluations of the objective function f.

3.3.3 *Computational Complexity of* `spinterp`. We immediately obtain the cost of `spinterp`

$$\text{cost}(\texttt{spinterp}, n, d) = \mathcal{O}\left(\sum_{k=0}^{n} p \cdot \#\Delta S_{k,d} \cdot d\right) = \mathcal{O}\left(p \cdot d \cdot \binom{n+d}{d}\right), \tag{15}$$

where $p = \#P$ is the number of points to interpolate, by simply considering the number of calls to `spinterpstep`.

4. NUMERICAL RESULTS

We have conducted a comparison of the piecewise multilinear interpolation using the L_∞-based grids, with (H^{M}) and without (H^{NB}) boundary nodes and the Clenshaw-Curtis grid H^{CC}, using the testing package of Genz [1987], designed for evaluating the performance of numerical integration algorithms. In Barthelmann et al. [2000], it was used to test the performance of sparse grid interpolation. The testing package consists of six families of functions defined on $[0, 1]^d$ with the following characteristics:

$$\begin{aligned}
\text{oscillatory:}\quad & f_1(x) = \cos\left(2\pi w_1 + \sum_{i=1}^{d} c_i x_i\right),\\
\text{product peak:}\quad & f_2(x) = \prod_{i=1}^{d} \left(c_i^{-2} + (x_i - w_i)^2\right)^{-1},\\
\text{corner peak:}\quad & f_3(x) = \left(1 + \sum_{i=1}^{d} c_i x_i\right)^{-(d+1)},\\
\text{Gaussian:}\quad & f_4(x) = \exp\left(-\sum_{i=1}^{d} c_i^2 \cdot (x_i - w_i)^2\right),\\
\text{continuous:}\quad & f_5(x) = \exp\left(-\sum_{i=1}^{d} c_i \cdot |x_i - w_i|\right),\\
\text{discontinuous:}\quad & f_6(x) = \begin{cases} 0, & \text{if } x_1 > w_1 \text{ or } x_2 > w_2,\\ \exp\left(\sum_{i=1}^{d} c_i x_i\right), & \text{otherwise.}\end{cases}
\end{aligned}$$

Varying test functions can be obtained by altering the parameters $c = (c_1, \ldots, c_n)$ and $w = (w_1, \ldots, w_n)$. We chose these parameters randomly from $[0, 1]$. Similarly to Barthelmann et al. [2000], we normalized the c_i such that $\sum_{i=1}^{d} c_i = b_j$, with b_j depending on d, f_j according to

j	1	2	3	4	5	6
b_j	1.5	d	1.85	7.03	20.4	4.3

Furthermore, we normalized the w_i such that $\sum_{i=1}^{d} w_i = 1$.

Figure 7 shows the absolute error

$$e = \max_{i=1,\ldots,1000} |f(y_i) - A_{q,d}(f)(y_i)| \tag{16}$$

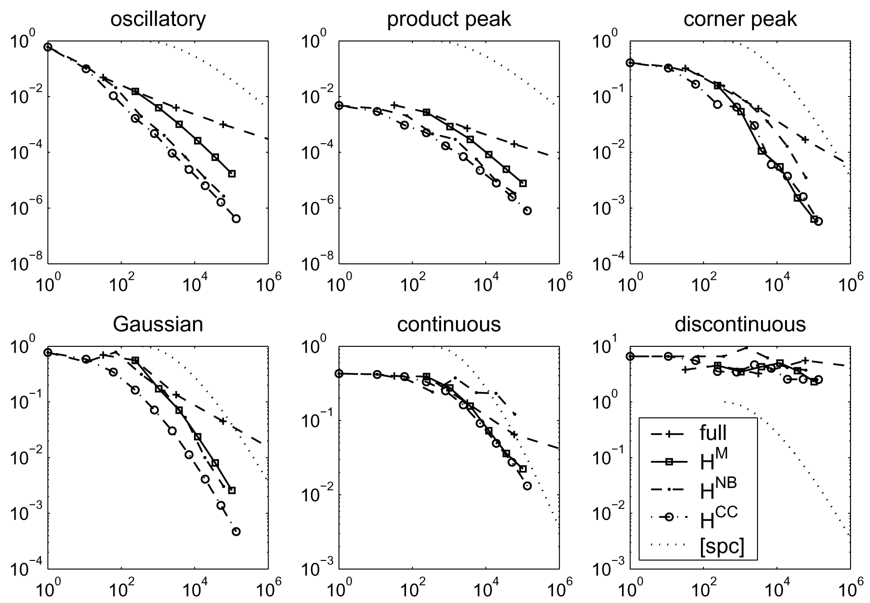


Fig. 7. Error plots for $A_{q,5}(f_{1,\dots,6})$ with H^{M}, H^{NB}, and H^{CC} and increasing q. The absolute error according to Equation (16) is plotted over the number of grid points N. [spc] is the asymptotic convergence rate according to Equation (6). For comparison, we have included the curve "full" showing the interpolation quality obtained with equidistant multilinear interpolation on a full grid.

over the total number of sparse grid points N for $d = 5$, for the six test functions. In Equation (16), we have replaced the error norm from Equation (6) by a discrete approximation. The points $y_1, \dots, y_{1000} \in [0, 1]^d$ were generated randomly. As expected, all schemes show the same asymptotic behavior according to Equation (6) in case of $f \in F$. However, for the oscillatory, the product peak, and the Gaussian-shaped test functions, interpolation with H^{CC} and H^{NB} produced better results than H^{M}, that is, fewer support nodes were needed to achieve similar accuracies. In the case of the corner peak test function, H^{M} sometimes performed slightly better. This difference results from the different distribution of the support nodes (see Figures 1 and 2), which is more dense for H^{M} at the boundary. In case of the continuous and the discontinuous test functions, all three grid types could not provide the asymptotic error decay of Equation (6), since $f \notin F$.

5. IMPLEMENTATION IN MATLAB

In this section, we describe our MATLAB implementation for piecewise multilinear sparse grid interpolation.

5.1 Data Structure

The most important data to be handled by the algorithms provided in the previous section are the sets of multiindices $\Delta S_{k,d}$, $k = 0, \dots, n$, the sets of sparse grid points $\Delta H_{k+d,d}$, $k = 0, \dots, n$, and the set of hierarchical surpluses $Z_{n,d}$.

k	$\Delta S_{k,d}$	*ID*	*Data*	$\Delta H_{k+d,d}$	*ID*	*Data*	Z_k	*ID*	*Data*
0	$\Delta S_{0,3}$	1	1 1 1	$\Delta H_{3,3}$	1	0.5 0.5 0.5	Z_0	1	w_1
1	$\Delta S_{1,3}$	1	2 1 1	$\Delta H_{4,3}$	1	0 0.5 0.5	Z_1	1	w_1
		2	1 2 1		2	1 0.5 0.5		2	w_2
		3	1 1 2		3	0.5 0 0.5		3	w_3
					4	0.5 1 0.5		4	w_4
					5	0.5 0.5 0		5	w_5
					6	0.5 0.5 1		6	w_6
2	$\Delta S_{2,3}$	1	3 1 1	$\Delta H_{5,3}$	1	.25 0.5 0.5	Z_2	1	w_1
		2	2 2 1		2	.75 0.5 0.5		2	w_2
		3	1 3 1		3	0 0 0.5		3	w_3
		4	2 1 2		4	1 0 0.5		4	w_4
		5	1 2 2		5	0 1 0.5		5	w_5
		6	1 1 3		6	1 1 0.5		6	w_6
						⋮			⋮
3	$\Delta S_{3,3}$	1	5 1 1						
		2	4 2 1						
⋮			⋮						

Fig. 8. Data structure of MATLAB implementation, example for $d = 3$.

Our MATLAB implementation stores the information in the following straightforward way:

5.1.1 *Multi-Indices.* $\Delta S_{k,d}$ is a two-dimensional array with d columns and $\#\Delta S_{k,d}$ rows. Each row contains one multiindex. $\Delta S_{k,d}$ is obtained by the internal function `spgetseq` and not permanently stored.

5.1.2 *Grid Points.* Also not permanently stored are the sparse grid points $\Delta H_{k+d,d}$ computed by `spgrid`. They are only internally used by the `spvals` algorithm. The points are consecutively stored as a single row in a two-dimensional double array with d columns. This is done in the same order as the multi-indices in $\Delta S_{k,d}$. Note that the number of points $n(\mathbf{i}) = \#X_{\Delta}^{i_1} \times \cdots \times X_{\Delta}^{i_d}$ per multi-index $\mathbf{i}$ usually varies (see also Figure 8, indicated by corresponding shading of the row indices).

5.1.3 *Hierarchical Surpluses.* Finally, the sets of hierarchical surpluses are organized as a $1 \times (n+1)$ cell array consisting of the subarrays Z_k, $k = 0, \dots, n$. These hold the hierarchical surpluses of level k constructed by the `spvals` algorithm. Figure 8 illustrates the ordering of the hierarchical surpluses according to the sparse grid point ordering.

5.2 User Interface

The user may select any of the grid types described in this article. To make the tool as easy as possible to use, we decided to follow the approach of Shampine and Reichelt [1997]. All sparse grid interpolation routines can be called in the

Table II. Major Options of the MATLAB Implementation

Property	Value	Description
`GridType`	String	The grid type. The default type is `'Clenshaw-Curtis'`; other possible values are `'Maximum'` and `'NoBoundary.'`
`RelTol`	Scalar	A relative error tolerance that applies to all hierarchical surpluses w^k of the current deepest level k of the sparse grid interpolation formula. The default value is 10^{-2} (1% accuracy).
`AbsTol`	Scalar	Absolute error tolerance. The default value is 10^{-6}.
`Vectorized`	on | off	Indicates if the objective function `fun` is available for vectorized evaluation. The default value is `'off.'` Vectorized coding of `fun` can significantly reduce the computation time used by `spvals`.
`MinDepth`	Integer	Minimum number of levels to compute (default: 2).
`MaxDepth`	Integer	Maximum number of levels to compute (default: 8).
`NumberOfOutputs`	Integer	If `fun` produces multiple outputs (all must be scalar), indicate this here to perform the sparse grid computation for many output variables at once. Also see the example `spdemovarout.m`.

same way, and providing additional options is optional. An overview of the most important options is given in Table II. Please refer to the user manual enclosed with the software for additional information on the available routines and their syntax.

5.3 Performance Results

In this section, we demonstrate the performance of the proposed algorithms in MATLAB. The performance tests were carried out using a MATLAB V6.5 implementation running on a Linux i686 1.6 GHz PC.

Figure 9(a) shows the time to compute the hierarchical surpluses depending on the dimension d with respect to the number of sparse grid support nodes N. Figure 9(b) shows the time to compute 1000 interpolated values depending on N. Both curves are plotted for the Clenshaw-Curtis grid. Similar results are obtained for the other two grid types. The dashed lines indicate the expected computational costs according to the formulae Equation (14) and Equation (15) (position of the asymptotic curves chosen for reasons of clarity).

6. CONCLUDING REMARKS

In this article, we have provided detailed algorithms for performing sparse grid interpolation. The algorithms were implemented in an easy to use MATLAB package. Our formulation is based on the notion of *hierarchical surpluses*, which permit the hierarchical construction of increasingly accurate interpolants within a provided error tolerance. This is especially useful when the objective function is expensive to evaluate and the number of function evaluations should be kept at a minimum. By computing and storing the hierarchical surpluses, multiple evaluations of a sparse grid interpolant $A_{n+d,d}(f)$ can be performed efficiently.

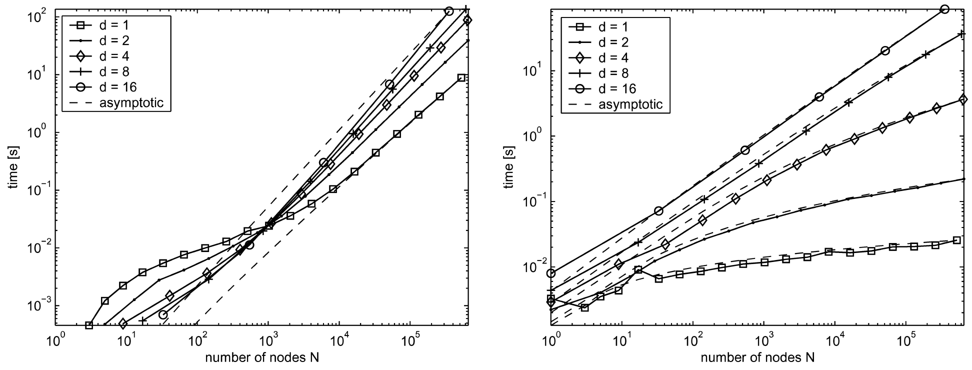


Fig. 9. (a) Time the compute the hierarchical surpluses (CC-grid); (b) time to evaluate interpolant 1000 times (CC-grid).

Here, we have placed an emphasis on sparse grid interpolation based on piecewise linear basis functions. Piecewise linear basis functions provide a good compromise between accuracy and computational cost due to their bounded support. However, the provided algorithms given in Section 3 can be easily applied to other sparse grid types and basis functions, such as higher order polynomial interpolation [Barthelmann et al. 2000], which is better suited in case of very high accuracy requirements and very smooth objective functions. An implementation using multivariate barycentric Lagrange interpolation and an approach to treating very high-dimensional problems are described in Klimke [2006, chapter 3].

An important application of sparse grid interpolants lies in the automatic construction of surrogate functions in optimization problems. For instance, we have successfully used the presented sparse grid interpolation approach in the implementation of the extension principle of fuzzy arithmetic (used in uncertainty modeling), where multiple global optimization problems for d-dimensional search boxes must be solved for an expensive objective function with multiple output parameters [Klimke et al. 2004; Klimke and Wohlmuth 2005].

REFERENCES

Barthelmann, V., Novak, E., and Ritter, K. 2000. High dimensional polynomial interpolation on sparse grids. *Adv. Comput. Math. 12*, 4, 273–288.

Bungartz, H.-J. 1992. Dünne Gitter und deren Anwendung bei der adaptiven Lösung der dreidimensionalen Poissongleichung. Ph.D. dissertation. Technische Universität München, Munich, Germany.

Bungartz, H.-J. 1998. *Finite Elements of Higher Order on Sparse Grids*. Shaker Verlag, Aachen, Germany.

Bungartz, H.-J. and Griebel, M. 2004. Sparse grids. *Acta Numerica 13*, 147–269.

Clenshaw, C. W. and Curtis, A. R. 1960. A method for numerical integration on an automatic computer. *Numer. Math. 2*, 197–205.

Genz, A. C. 1987. A package for testing multiple integration subroutines. In *Numerical Integration*, P. Keast and G. Fairweather, Eds. Kluwer, Dordrecht, The Netherlands, 337–340.

KLIMKE, A. 2006. Uncertainty modeling using sparse grids. PhD dissertation. Universität Stuttgard, Germany.

KLIMKE, A., WILLNER, K., AND WOHLMUTH, B. 2004. Uncertainty modeling using fuzzy arithmetic based on sparse grids: Applications to dynamic systems. *Int. J. Uncertaint. Fuzz. Knowl.-Based Syst. 12*, 6, 745–759.

KLIMKE, A. AND WOHLMUTH, B. 2005. Computing expensive multivariate functions of fuzzy numbers using sparse grids. *Fuzz. Sets Syst. 154*, 3, 432–453.

NOVAK, E. AND RITTER, K. 1996. High-dimensional integration of smooth functions over cubes. *Numer. Math. 75*, 1, 79–97.

SCHREIBER, A. 2000. Smolyak's method for multivariate interpolation. Ph.D. dissertation. Georg-August-Universität Göttingen, Göttingen, Germany.

SHAMPINE, L. F. AND REICHELT, M. W. 1997. The MATLAB ODE suite. *SIAM J. Sci. Comput. 18*, 1, 1–22.

SMOLYAK, S. A. 1963. Quadrature and interpolation formulas for tensor products of certain classes of functions. *Soviet Math. Dokl. 4*, 240–243.

SPRENGEL, F. 1997. Interpolation and wavelets on sparse Gauss-Chebyshev grids. In *Multivariate Approximation*, W. Haussmann et al., Eds. Mathematical Research, vol. 101. Akademie Verlag, Berlin, Germany, 269–286.

SPRENGEL, F. 1998. Periodic interpolation and wavelets on sparse grids. *Numer. Algorith. 17*, 1–2, 147–169.

Received March 2004; revised November 2004, February 2005; accepted March 2005