Technische Universität München
Institut für Informatik
Prof. Dr. Hans-Joachim Bungartz
Alexander Heinecke, M.Sc., M.Sc. with honors

WS 2012/2013

Algorithms of Scientific Computing II

Exercise 5 - Hierarchical Basis, Sparse Grids and Interpolation

In this exercise we want to get a feeling for sparse grids. A good start is to write a sparse grid implementation for a simple application. We want to consider here the d-linear interpolation of a function $f(x_1, \dots, x_d)$ on the interval $[0,1]^d$ with 0 on the borders.

- **1D Basis Functions**
  We start by looking at the one-dimensional case $d = 1$. In Maple we obtain the basis functions $\phi_{l,i}(x)$ depending on the level l and index i with the help of the `piecewise`-Function:

```
phi_li := (l,i) ->
((x) -> piecewise(x<=(i-1)/2^l, 0, x<=i/2^l, 2^l*x+1-i, x<=(i+1)/2^l, i+1-2^l*x));
```

  - To a given level n the function $f(x) := 4x(1-x)$ in the interval $[0,1]$ should be piecewise linear interpolated with the mesh width $h_n := 2^{-n}$. Do this using the node basis. Which $\phi_{l,i}$ are needed here?

    Create a list which contains all basis functions (characterized by l and i). For example, level l consists of the following basis functions:

    `list := [seq([l,i], i=1..2^l-1)];`

    Store the coefficients of the basis functions in a vector u ab. In order to find the corresponding coefficient to a certain basis function $\phi_{l,i}$, use a hash map which returns to a given $[l,i]$ the correct index in u. In Maple this can be done as follows: `hashtable := table();` and for example `hashtable[[4,6]] := 4;`.

  - Now we want to use the hierarchical basis instead. Write a function which returns you the tuples $[l,i]$ for all hierarchical basis.

    Let's now assume, we still had the hat functions of the node basis and fill the coefficient vector u with the function values as done above.

    Write a function, which without additional storage requirements takes the node basis coefficients and computes the hierarchical coefficients (this process is called hierarchical and the resulting coefficients are called hierarchical surpluses). As a reminder, the hierarchical surpluses are defined as:

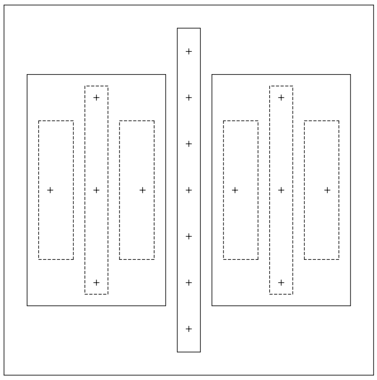
$$u_{l,i} := f(\frac{a+b}{2}) - \frac{f(a)+f(b)}{2}.$$

- **2D Basis Functions**
  Now we want to achieve the same but for higher dimensions. In this exercise we will only consider $d = 2$ as for $d > 2$ the process is similar − and of course, for $d > 2$ we can't draw our function anymore. Our function is now defined as follows: $f(x_1, x_2) := 16x_1(1-x_1)x_2(1-x_2)$.

The basis functions are now

$$\phi_{l_1,i_1,l_2,i_2}(x_1,x_2) := \phi_{l_1,i_1}(x_1)\phi_{l_2,i_2}(x_2).$$

- **Grid Generation.** Write a function which returns (for the sparse grid corresponding to the sparse function space $V_n^{(1)}$) a list with entries $[l_1, i_1, l_2, i_2]$ (grid point) for each basis function. Two possible implementations:
  i) Start by identifying which subspaces $W_{(l_1,l_2)}$ are needed for the sparse grid and create the corresponding grid points.
  ii) The sparse grid in d dimensions and level n can be seen as consisting of a $d-1$-dimensional sparse grid of level n and two d-dimensional sparse grids of level $n-1$:

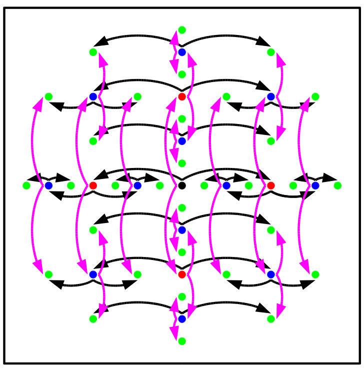


- **Hierarchization** Here again the vector u has to be created to hold the hierarchical coefficients and also a hash map, which returns the correct index to a given basis function, for example:

  `hashtable[[1,1,1,1]]` $\rightarrow$ `1`.

  Fill u with function values of f at the corresponding grid points.

  Write now a function which performs a 2-dimensional hierarchization. After the function is applied on u, each entry will contain the hierarchical coefficient of the corresponding hierarchical basis function. To perform a correct hierarchization, the 1-dimensional hierarchization has to be applied successively in all dimensions:



  Which are the surpluses for $n = 3$? Draw the resulting function.

- **Evaluating the Interpolant**
  Lastly, we want to see how can we evaluate the obtained interpolant $f_N(x_1, x_2)$. In Maple

this is done quite easy by explicitly setting up the interpolant:

$$f_N(x_1, x_2) := \sum_{[l_1, i_1, l_2, i_2] \in \texttt{list}} u[[l_1, i_1, l_2, i_2]] \phi_{l_1,i_1}(x_1) \phi_{l_2,i_2}(x_2)$$

However, in this case a lot the basis functions will be evaluated which are anyway 0 at the evaluation point (no support). How can we skip such evaluations?