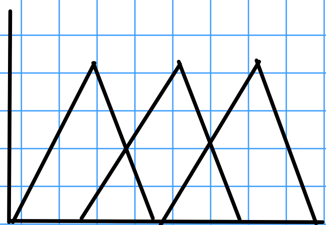
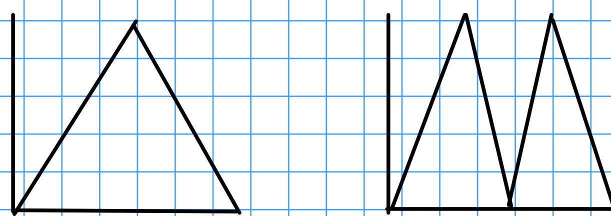
Worksheet 10 – Sparse Grids

Concept of sparse grids derives from the concept of hierarchical basis

Nodal basis

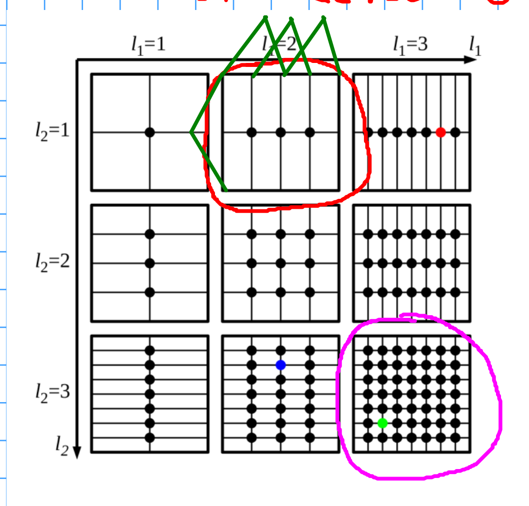


Hierarchical basis



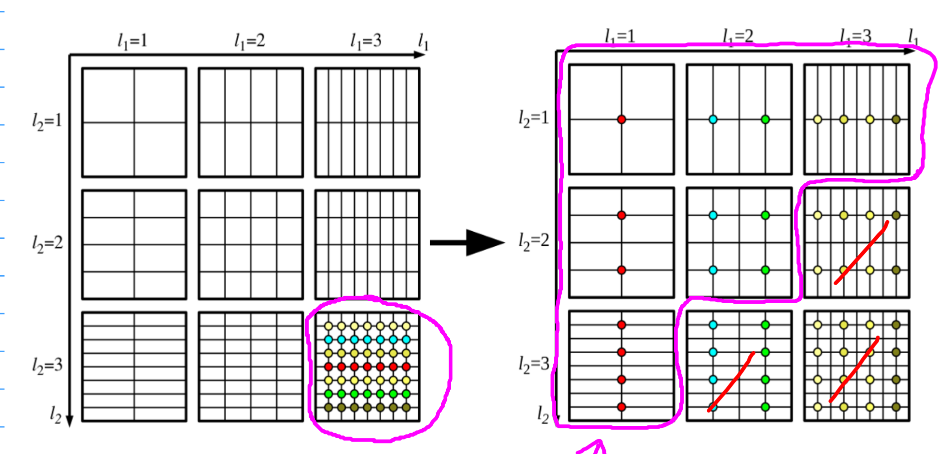
2nd level dim 1 →
1st level dim2 ↓

different grids
different discretization



equivalent of nodal basis representation

$V_{3,3}$ Each grid spans a space



V_3^1

Sparse grid

full grid

The space $V_{3,3}$ can also be spanned by these grids

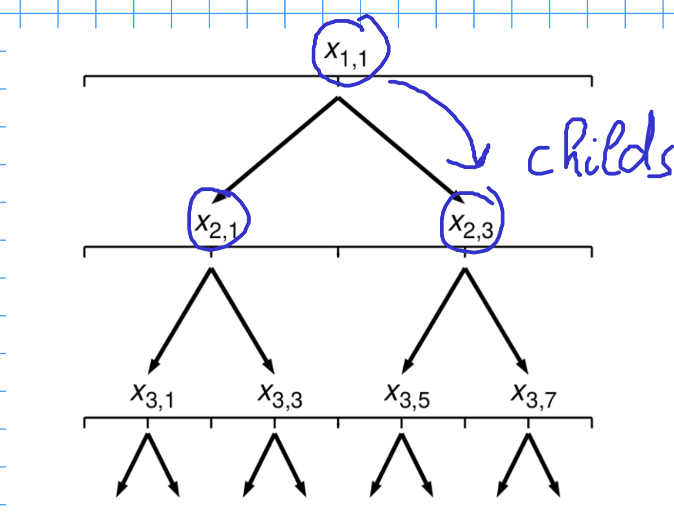
Omit some grids when trying to find an approximation of a function

They don't add much to the solution

→ ratio between cost and accuracy is not good enough to add them

Store grid using different data structures

Binary trees

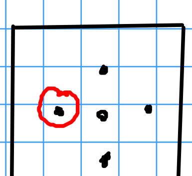


Exercise 1

But here we use a dictionary/map („Sg" in the code)

Map keys to values

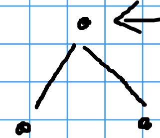
$(\vec{l}, \vec{i}) \rightarrow$ Pagodas



(i) Apply some criterion to the leaves of the sparse grid

Because we refine the leaves → create more children

How to check for a leaf



← if that point does not have children at some dimension → leaf

How to access children? $l+1$

$2i \pm 1$

MinLevelCriterion: Just refine until a certain level (user-defined) is reached

apply(): is „passed" since we just refine everything until min level is reached

finalize(): Create sparse grid structure

Now we have a grid with our basis functions (Pagoda objects) → we only have function values

(ii) Again access left and right child
Apply hierarchization for each child
Compute surplus (height of triangle, see last worksheets)
→ We now have the surpluses stored

(iii) For volume computation see worksheet 9

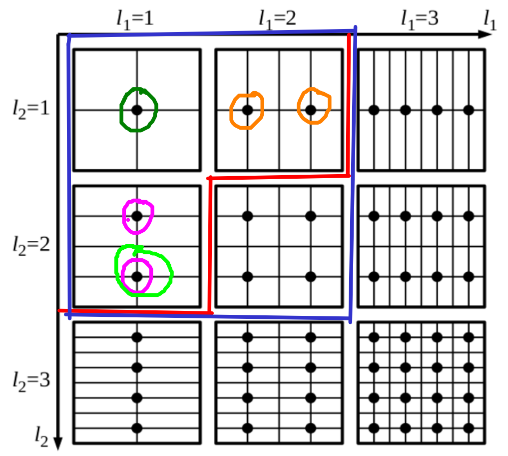
Exercise 2

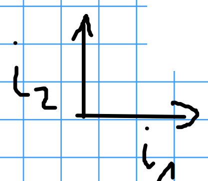
Given : incomplete sparse grid

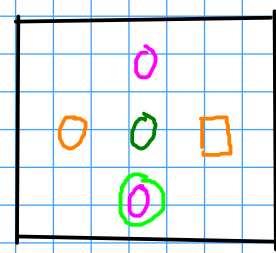
1) $V_2^1 \rightarrow$ What grid we are using

V_2^1

V_2







$\vec{l} = (1,2)$
$\vec{i} = (1,3)$

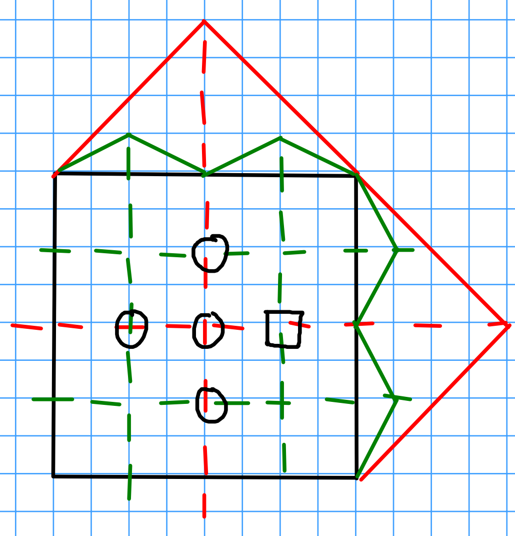
$\vec{l} = (2,1)$
$\vec{i} = (1,1)$

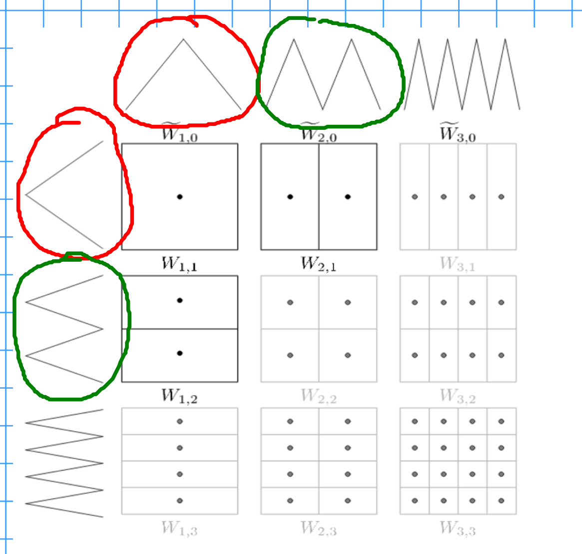
$\vec{l} = (1,1)$
$\vec{i} = (1,1)$

$\vec{l} = (2,1)$
$\vec{i} = (3,1)$

$\vec{l} = (1,2)$
$\vec{i} = (1,1)$

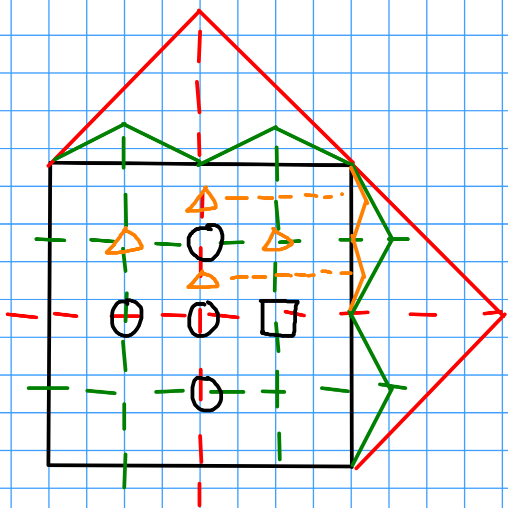
How these points are constructed : combination of the linear basis on each dimension

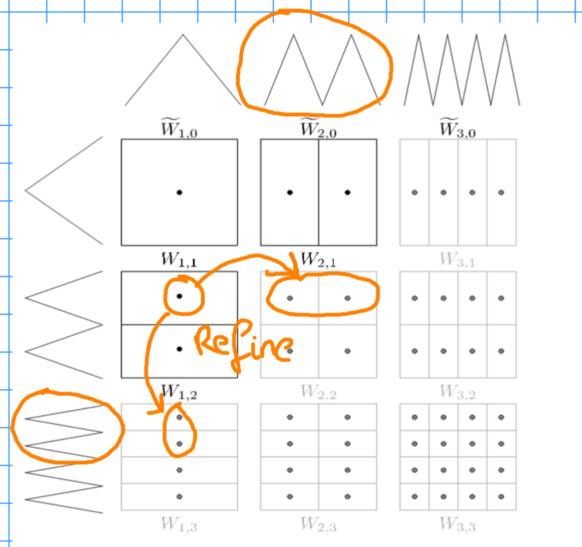




2) a) Refine $\vec{\ell} = (1, 2)$

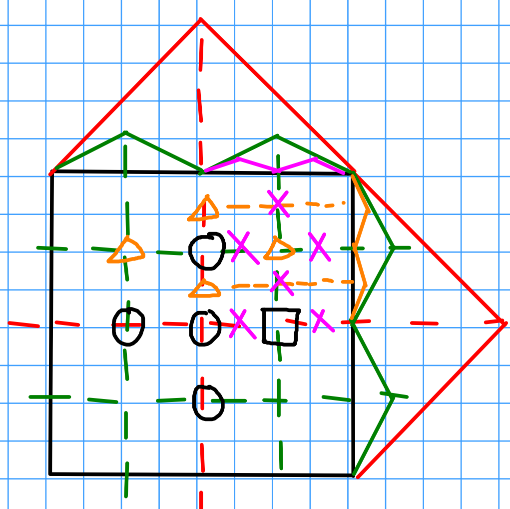
$\vec{i} = (1,3)$

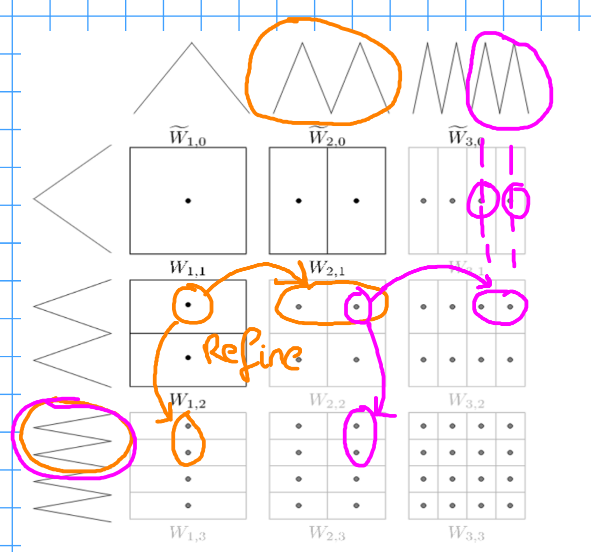




b) $\vec{\ell} = (2,2)$

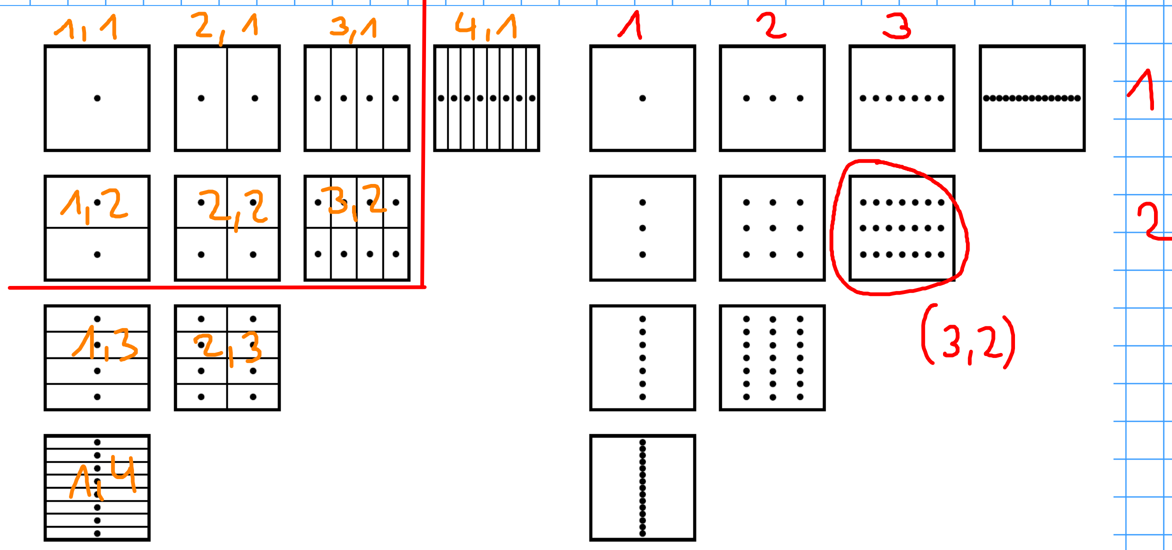
$\vec{i} = (3,3)$





Exercise 3) Combination Technique

(i) spot grid with $u_{3,2}$

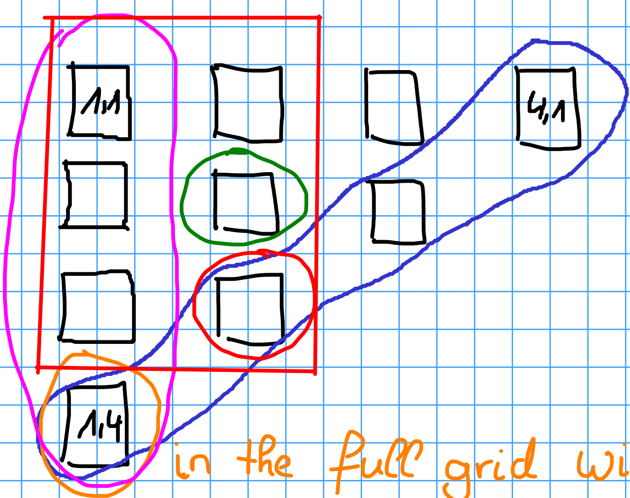


$$u_{\vec{e}} = \sum_{\vec{e}' \leq \vec{e}} w_{\vec{e}'}$$

(ii) $\sum_{|\vec{e}_1| = n+1} u_{\vec{e}}$ $\qquad |\vec{e}|_1 = |e_1| + |e_2| \ldots$

In (i) we found $\sum_{|\vec{e}|_1 = n+1} u_{\vec{e}} = \sum_{|\vec{e}|_1 = n+1} \sum_{\vec{e}' \leq \vec{e}} w_{\vec{e}'}$

→ since we know how many times $w_{\vec{e}}$ is repeated for each $\sum_{|\vec{e}|_1 = n+1} u_{\vec{e}}$ we can remove this sum and rewrite it like in the solution ex3



$|e_1| = n + 1 = 5$

Each $w_{\vec{e}}$ is associated with one of these grids

in the full grid will span these in the hierarchical grid

(iii) Also see Garcke, Jochen: „Sparse Grids in a Nutshell"

Adaptive Implementation

- Add a more complex refinement criterion
- We let the code decide on how many levels to refine
- Important then we have „non-symmetrical functions"