UNIVERSITÀ DEGLI STUDI FIRENZE

DIMAI
DIPARTIMENTO DI MATEMATICA
E INFORMATICA "ULISSE DINI"

Funded by the European Union
NextGenerationEU

Italiadomani
PIANO NAZIONALE DI RIPRESA E RESILIENZA

CN1 - Spoke 6

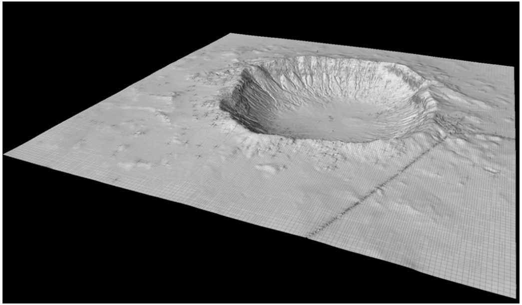
An Algorithmic Introduction to LR B-splines

Francesco Patrizi

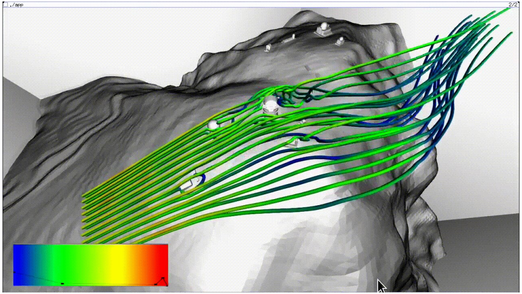
Picture: Acropolis

Athens, Greece

A tool for adaptive approximations

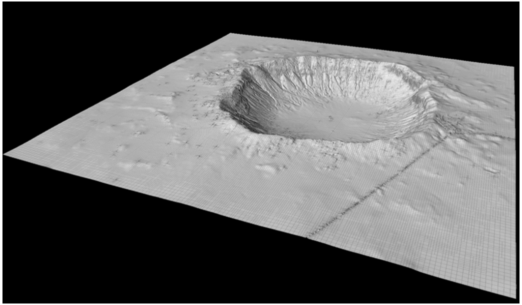


Quasi-Interpolation, Meteor Crater, AZ

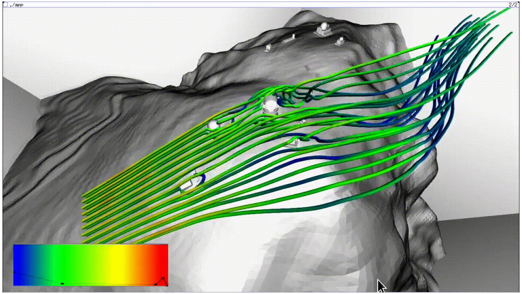


Wind streamlines around a telescope

A tool for adaptive approximations



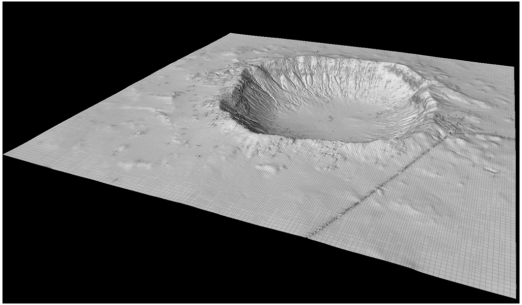
Quasi-Interpolation, Meteor Crater, AZ



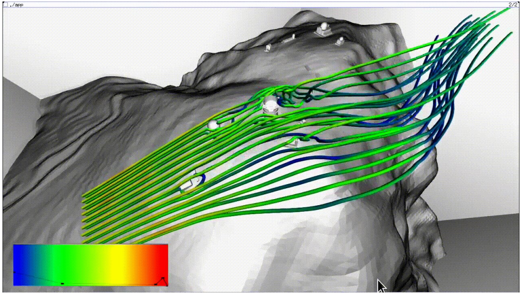
Wind streamlines around a telescope

Created by the vikings

A tool for adaptive approximations



Quasi-Interpolation, Meteor Crater, AZ



Wind streamlines around a telescope

Created by the vikings

T. Dokken

T. Lyche

K. F. Pettersen

Recalling univariate B-splines

(Local) Knot Vector: Given a degree p, $\boldsymbol{t} = \boldsymbol{t}_p$ with $|\boldsymbol{t}_p| = p + 2$ with repetitions

$$\underbrace{t_1 = \cdots = t_{m_1}}_{\max p+1 \text{ times}} < \underbrace{t_{m_1+1} = \cdots = t_{m_1+m_2}}_{\max p+1 \text{ times}} < \cdots$$

Recalling univariate B-splines

(Local) Knot Vector: Given a degree p, $\boldsymbol{t} = \boldsymbol{t}_p$ with $|\boldsymbol{t}_p| = p + 2$ with repetitions

$$\underbrace{t_1 = \cdots = t_{m_1}}_{\max p+1 \text{ times}} < \underbrace{t_{m_1+1} = \cdots = t_{m_1+m_2}}_{\max p+1 \text{ times}} < \cdots$$

Univariate B-spline: Given a degree p, the B-spline of degree p is defined recursively:

$$B[\boldsymbol{t}](t) = \frac{t - t_1}{t_{p+1} - t_1} B[t_1, \ldots, t_{p+1}](t) + \frac{t_{p+2} - t}{t_{p+2} - t_2} B[t_2, \ldots, t_{p+2}](t),$$

where each time a fraction with zero denominator appears, it is taken as zero. The initial B-splines of degree 0 are defined as

$$B[t_i, t_{i+1}](t) := \begin{cases} 1 & \text{if } t_i \leq t < t_{i+1}; \\ 0 & \text{otherwise}; \end{cases}$$

Recalling univariate B-splines

(Local) Knot Vector: Given a degree p, $\boldsymbol{t} = \boldsymbol{t}_p$ with $|\boldsymbol{t}_p| = p + 2$ with repetitions

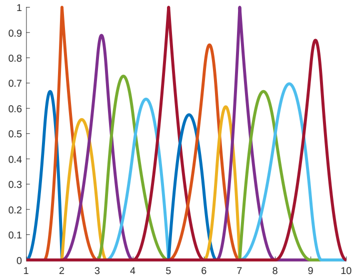
$$\underbrace{t_1 = \cdots = t_{m_1}}_{\text{max } p+1 \text{ times}} < \underbrace{t_{m_1+1} = \cdots = t_{m_1+m_2}}_{\text{max } p+1 \text{ times}} < \cdots$$

Univariate B-spline: Given a degree p, the B-spline of degree p is defined recursively:

$$B[\boldsymbol{t}](t) = \frac{t - t_1}{t_{p+1} - t_1} B[t_1, \ldots, t_{p+1}](t) + \frac{t_{p+2} - t}{t_{p+2} - t_2} B[t_2, \ldots, t_{p+2}](t),$$

where each time a fraction with zero denominator appears, it is taken as zero. The initial B-splines of degree 0 are defined as

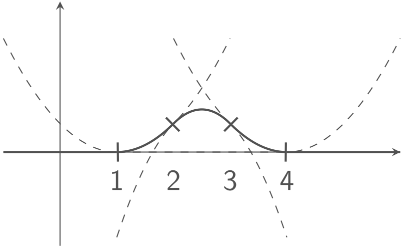
$$B[t_i, t_{i+1}](t) := \begin{cases} 1 & \text{if } t_i \le t < t_{i+1}; \\ 0 & \text{otherwise;} \end{cases}$$



Recalling univariate B-splines

$p = 2$, $\boldsymbol{t} = (1, 2, 3, 4)$

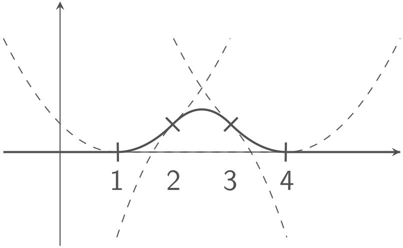
$$B[\boldsymbol{t}](t) = \begin{cases} 0 & t \notin [1,4], \\ \frac{1}{2}t^2 - t + \frac{1}{2} & 1 \leq t < 2, \\ -t^2 + \frac{1}{2}t - \frac{11}{2} & 2 \leq t < 3, \\ \frac{1}{2}t^2 - 4t + 8 & 3 \leq t \leq 4. \end{cases}$$



Recalling univariate B-splines

$p = 2$, $\boldsymbol{t} = (1, 2, 3, 4)$

$$B[\boldsymbol{t}](t) = \begin{cases} 0 & t \notin [1,4], \\ \frac{1}{2}t^2 - t + \frac{1}{2} & 1 \le t < 2, \\ -t^2 + \frac{1}{2}t - \frac{11}{2} & 2 \le t < 3, \\ \frac{1}{2}t^2 - 4t + 8 & 3 \le t \le 4. \end{cases}$$

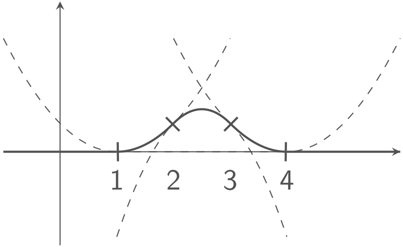


Properties:

Recalling univariate B-splines

$p = 2$, $\boldsymbol{t} = (1, 2, 3, 4)$

$$B[\boldsymbol{t}](t) = \begin{cases} 0 & t \notin [1,4], \\ \frac{1}{2}t^2 - t + \frac{1}{2} & 1 \leq t < 2, \\ -t^2 + \frac{1}{2}t - \frac{11}{2} & 2 \leq t < 3, \\ \frac{1}{2}t^2 - 4t + 8 & 3 \leq t \leq 4. \end{cases}$$

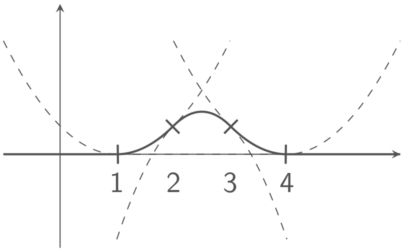


Properties:

- $B[\boldsymbol{t}] \geq 0$,

Recalling univariate B-splines

$p = 2$, $\boldsymbol{t} = (1, 2, 3, 4)$



$$B[\boldsymbol{t}](t) = \begin{cases} 0 & t \notin [1,4], \\ \frac{1}{2}t^2 - t + \frac{1}{2} & 1 \leq t < 2, \\ -t^2 + \frac{1}{2}t - \frac{11}{2} & 2 \leq t < 3, \\ \frac{1}{2}t^2 - 4t + 8 & 3 \leq t \leq 4. \end{cases}$$

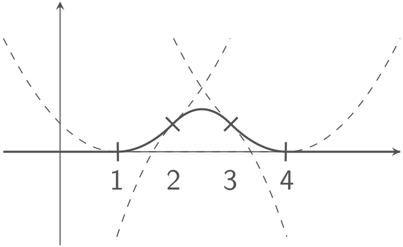
Properties:

- $B[\boldsymbol{t}] \geq 0$,
- $\operatorname{supp} B[\boldsymbol{t}] = [t_1, t_{p+2}]$,

Recalling univariate B-splines

$p = 2$, $\boldsymbol{t} = (1, 2, 3, 4)$

$$B[\boldsymbol{t}](t) = \begin{cases} 0 & t \notin [1,4], \\ \frac{1}{2}t^2 - t + \frac{1}{2} & 1 \le t < 2, \\ -t^2 + \frac{1}{2}t - \frac{11}{2} & 2 \le t < 3, \\ \frac{1}{2}t^2 - 4t + 8 & 3 \le t \le 4. \end{cases}$$



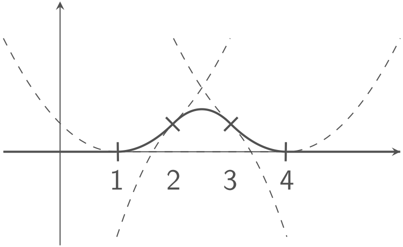
Properties:

- $B[\boldsymbol{t}] \ge 0$,
- $\operatorname{supp} B[\boldsymbol{t}] = [t_1, t_{p+2}]$,
- $B[\boldsymbol{t}]_{|[t_i, t_{i+1})} \in \Pi_p$,

Recalling univariate B-splines

$p = 2$, $\boldsymbol{t} = (1, 2, 3, 4)$

$$B[\boldsymbol{t}](t) = \begin{cases} 0 & t \notin [1,4], \\ \frac{1}{2}t^2 - t + \frac{1}{2} & 1 \le t < 2, \\ -t^2 + \frac{1}{2}t - \frac{11}{2} & 2 \le t < 3, \\ \frac{1}{2}t^2 - 4t + 8 & 3 \le t \le 4. \end{cases}$$



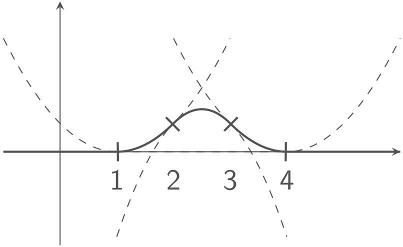
Properties:

- $B[\boldsymbol{t}] \ge 0$,
- $\operatorname{supp} B[\boldsymbol{t}] = [t_1, t_{p+2}]$,
- $B[\boldsymbol{t}]_{|[t_i, t_{i+1})} \in \Pi_p$,
- $B[\boldsymbol{t}]$ is C^{p-m_i}-continuous at t_i,

Recalling univariate B-splines

$p = 2$, $\boldsymbol{t} = (1, 2, 3, 4)$

$$B[\boldsymbol{t}](t) = \begin{cases} 0 & t \notin [1,4], \\ \frac{1}{2}t^2 - t + \frac{1}{2} & 1 \leq t < 2, \\ -t^2 + \frac{1}{2}t - \frac{11}{2} & 2 \leq t < 3, \\ \frac{1}{2}t^2 - 4t + 8 & 3 \leq t \leq 4. \end{cases}$$



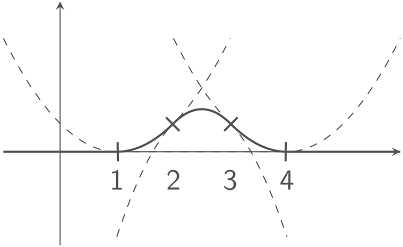
Properties:

- $B[\boldsymbol{t}] \geq 0$,
- $\operatorname{supp} B[\boldsymbol{t}] = [t_1, t_{p+2}]$,
- $B[\boldsymbol{t}]_{|[t_i, t_{i+1})} \in \Pi_p$,
- $B[\boldsymbol{t}]$ is C^{p-m_i}-continuous at t_i,
- locally linearly independent,

Recalling univariate B-splines

$p = 2$, $\boldsymbol{t} = (1, 2, 3, 4)$

$$B[\boldsymbol{t}](t) = \begin{cases} 0 & t \notin [1,4], \\ \frac{1}{2}t^2 - t + \frac{1}{2} & 1 \le t < 2, \\ -t^2 + \frac{1}{2}t - \frac{11}{2} & 2 \le t < 3, \\ \frac{1}{2}t^2 - 4t + 8 & 3 \le t \le 4. \end{cases}$$



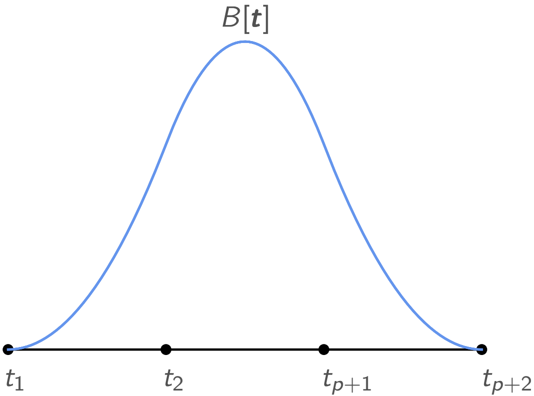
Properties:

- $B[\boldsymbol{t}] \ge 0$,
- $\operatorname{supp} B[\boldsymbol{t}] = [t_1, t_{p+2}]$,
- $B[\boldsymbol{t}]_{|[t_i, t_{i+1})} \in \Pi_p$,
- $B[\boldsymbol{t}]$ is C^{p-m_i}-continuous at t_i,
- locally linearly independent,
- form a partition of unity.

Knot Insertion

Knot Insertion: Suppose we insert a knot $\hat{t} \to \boldsymbol{t}$. We obtain two knot vectors $\boldsymbol{t}_1$ and $\boldsymbol{t}_2$, considering the first and the last $p+2$ knots respectively in $(t_1, \dots, \hat{t}, \dots, t_{p+2})$. Then

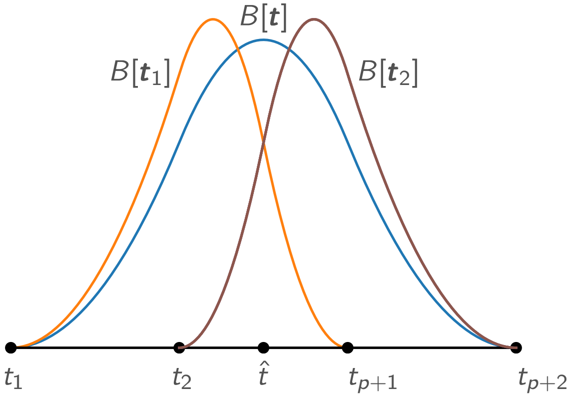
$$B[\boldsymbol{t}] = \alpha_1 B[\boldsymbol{t}_1] + \alpha_2 B[\boldsymbol{t}_2] \quad \text{with } \alpha_1, \alpha_2 \in (0, 1]$$



Knot Insertion

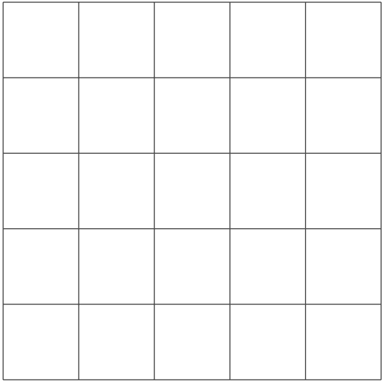
Knot Insertion: Suppose we insert a knot $\hat{t} \to \boldsymbol{t}$. We obtain two knot vectors $\boldsymbol{t}_1$ and $\boldsymbol{t}_2$, considering the first and the last $p+2$ knots respectively in $(t_1, \dots, \hat{t}, \dots, t_{p+2})$. Then

$$B[\boldsymbol{t}] = \alpha_1 B[\boldsymbol{t}_1] + \alpha_2 B[\boldsymbol{t}_2] \quad \text{with } \alpha_1, \alpha_2 \in (0, 1]$$



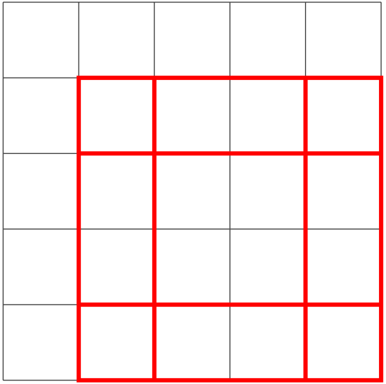
B-splines on Tensor Meshes

Given a tensor mesh $\mathcal{N}$ and a bidegree (p_1, p_2) (for instance $(2, 2)$),



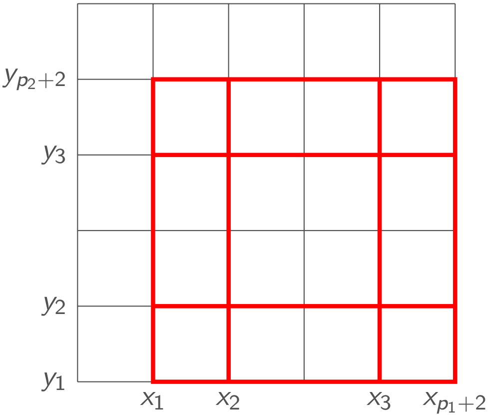
B-splines on Tensor Meshes

Given a tensor mesh $\mathcal{N}$ and a bidegree (p_1, p_2) (for instance $(2, 2)$),
let $\mathcal{N}_B$ be a subcollection of meshlines in $\mathcal{N}$ forming a sub-grid of $p_1 + 2$ vertical lines and $p_2 + 2$ horizontal lines.



B-splines on Tensor Meshes

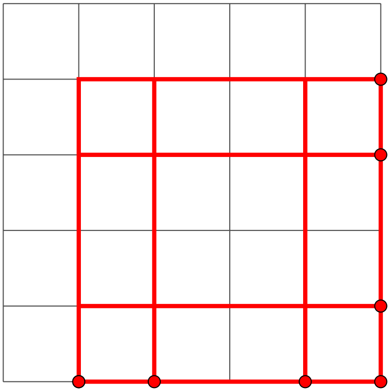
Given a tensor mesh $\mathcal{N}$ and a bidegree (p_1, p_2) (for instance $(2, 2)$),
let $\mathcal{N}_B$ be a subcollection of meshlines in $\mathcal{N}$ forming a sub-grid of $p_1 + 2$ vertical lines and $p_2 + 2$ horizontal lines.



Such vertical and horizontal lines can be parametrized as $\{x_i\} \times [y_1, y_{p_2+2}]$ and $[x_1, x_{p_1+2}] \times \{y_j\}$ with $\boldsymbol{x} := (x_i)_{i=1}^{p_1+2}$ and $\boldsymbol{y} = (y_j)_{j=1}^{p_2+2}$.

B-splines on Tensor Meshes

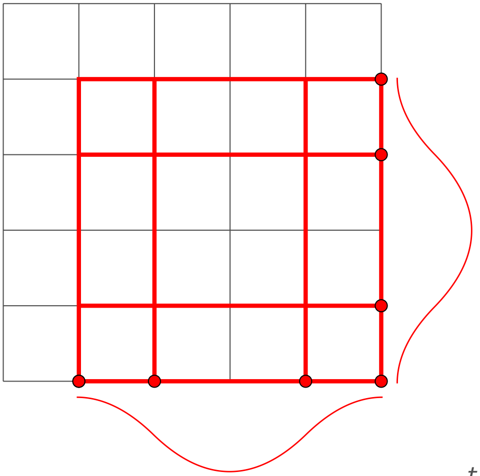
$\boldsymbol{x}$ and $\boldsymbol{y}$ are knot vectors on top on which we define univariate B-splines of degrees p_1 and p_2.



$$B[t_1, \ldots, t_{p+2}](t) = \frac{t - t_1}{t_{p+1} - t_1} B[t_2, \ldots, t_{p+2}](t) + \frac{t_{p+2} - t}{t_{p+2} - t_2} B[t_1, \ldots, t_{p+1}](t).$$

B-splines on Tensor Meshes

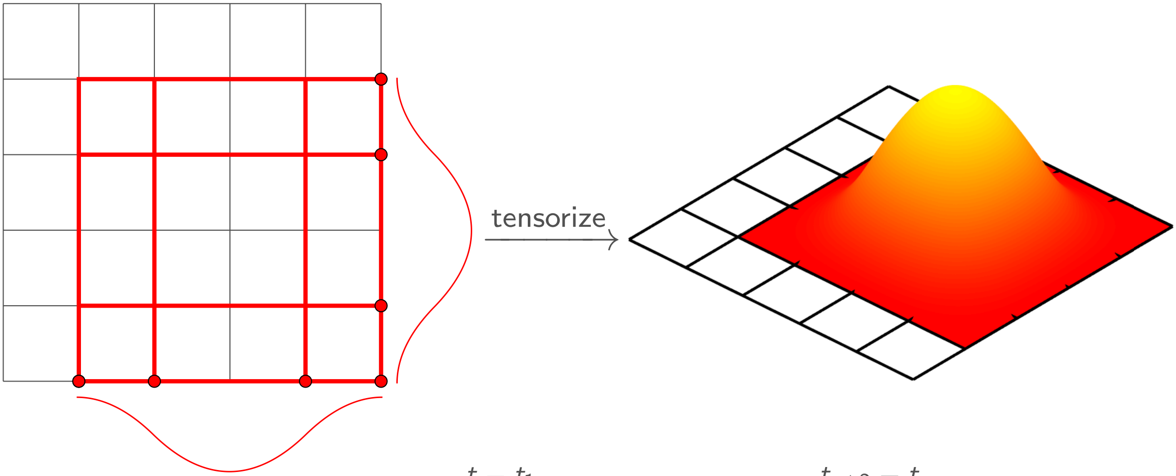
$\boldsymbol{x}$ and $\boldsymbol{y}$ are knot vectors on top on which we define univariate B-splines of degrees p_1 and p_2.



$$B[t_1, \ldots, t_{p+2}](t) = \frac{t - t_1}{t_{p+1} - t_1} B[t_2, \ldots, t_{p+2}](t) + \frac{t_{p+2} - t}{t_{p+2} - t_2} B[t_1, \ldots, t_{p+1}](t).$$

B-splines on Tensor Meshes

$\boldsymbol{x}$ and $\boldsymbol{y}$ are knot vectors on top on which we define univariate B-splines of degrees p_1 and p_2.

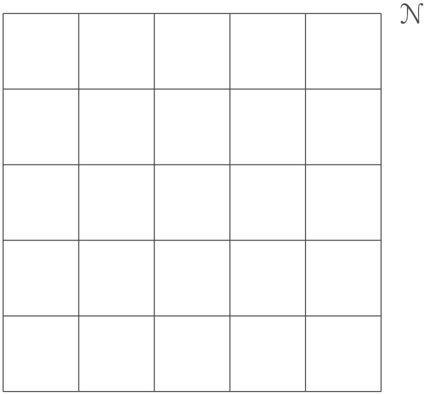


$$B[t_1, \ldots, t_{p+2}](t) = \frac{t - t_1}{t_{p+1} - t_1} B[t_2, \ldots, t_{p+2}](t) + \frac{t_{p+2} - t}{t_{p+2} - t_2} B[t_1, \ldots, t_{p+1}](t).$$

B-splines of Minimal Support

Each $\mathcal{N}_B$ corresponds to a B-spline B defined on $\mathcal{N}$.

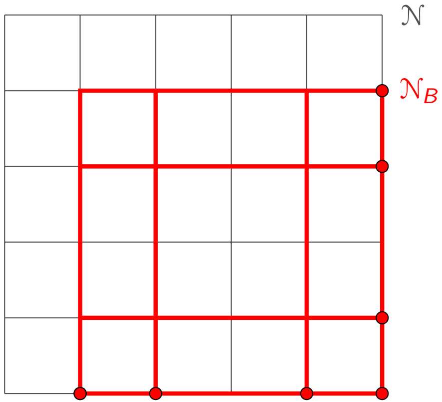
If no line in $\mathcal{N} \backslash \mathcal{N}_B$ traverses $\text{int}(\text{supp}\, B)$ then we say that B has minimal support on $\mathcal{N}$.



B-splines of Minimal Support

Each $\mathcal{N}_B$ corresponds to a B-spline B defined on $\mathcal{N}$.

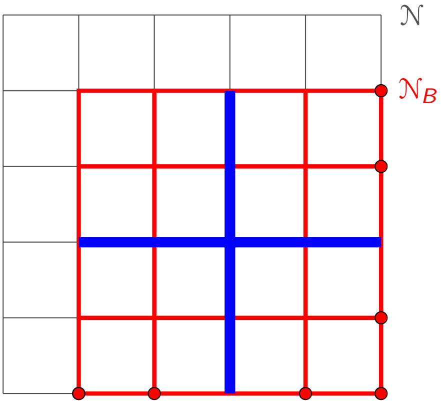
If no line in $\mathcal{N} \backslash \mathcal{N}_B$ traverses $\mathrm{int}(\mathrm{supp}\, B)$ then we say that B has minimal support on $\mathcal{N}$.



B-splines of Minimal Support

Each $\mathcal{N}_B$ corresponds to a B-spline B defined on $\mathcal{N}$.

If no line in $\mathcal{N} \backslash \mathcal{N}_B$ traverses $\mathrm{int}(\mathrm{supp}\, B)$ then we say that B has minimal support on $\mathcal{N}$.

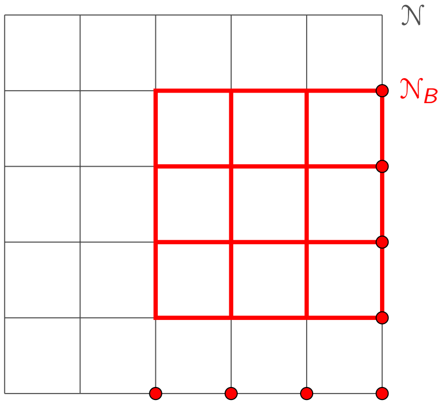


no minimal support on $\mathcal{N}$

B-splines of Minimal Support

Each $\mathcal{N}_B$ corresponds to a B-spline B defined on $\mathcal{N}$.

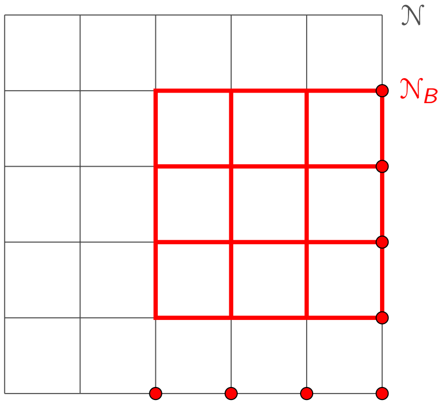
If no line in $\mathcal{N} \backslash \mathcal{N}_B$ traverses $\mathrm{int}(\mathrm{supp}\, B)$ then we say that B has minimal support on $\mathcal{N}$.



minimal support on $\mathcal{N}$

B-splines of Minimal Support

Each $\mathcal{N}_B$ corresponds to a B-spline B defined on $\mathcal{N}$.
If no line in $\mathcal{N} \backslash \mathcal{N}_B$ traverses $\mathrm{int}(\mathrm{supp}\, B)$ then we say that B has minimal support on $\mathcal{N}$.

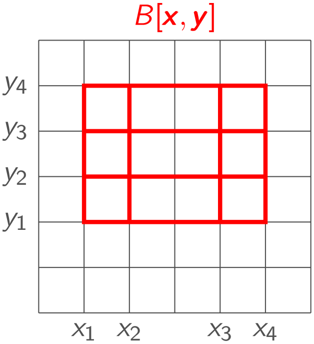


minimal support on $\mathcal{N}$

We call B-spline set on $\mathcal{N}$ the set of all the minimal support B-splines on $\mathcal{N}$.

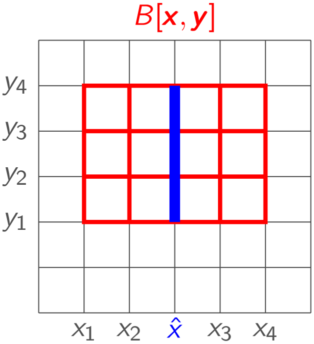
Knot insertion

Assume $B[\boldsymbol{x}, \boldsymbol{y}]$ no minimal support on $\mathcal{N}$ because of a vertical line at $x = \hat{x}$ with $\hat{x} \notin \boldsymbol{x}$.



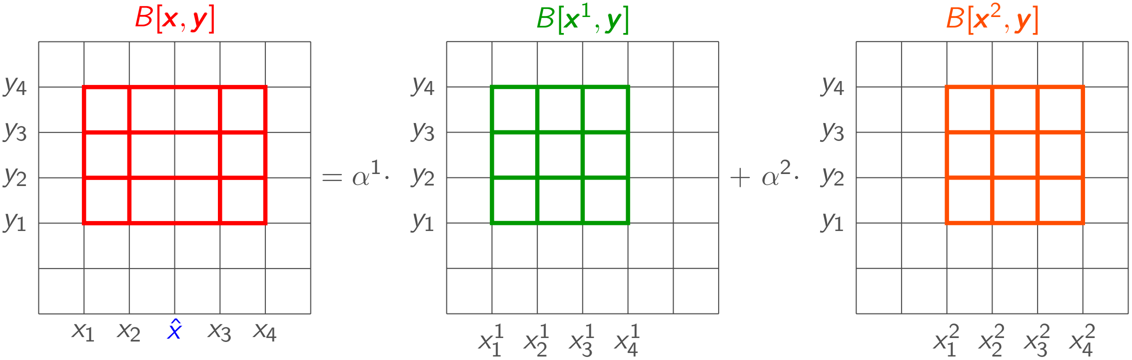
Knot insertion

Assume $B[\mathbf{x}, \mathbf{y}]$ no minimal support on $\mathcal{N}$ because of a vertical line at $x = \hat{x}$ with $\hat{x} \notin \mathbf{x}$.



Knot insertion

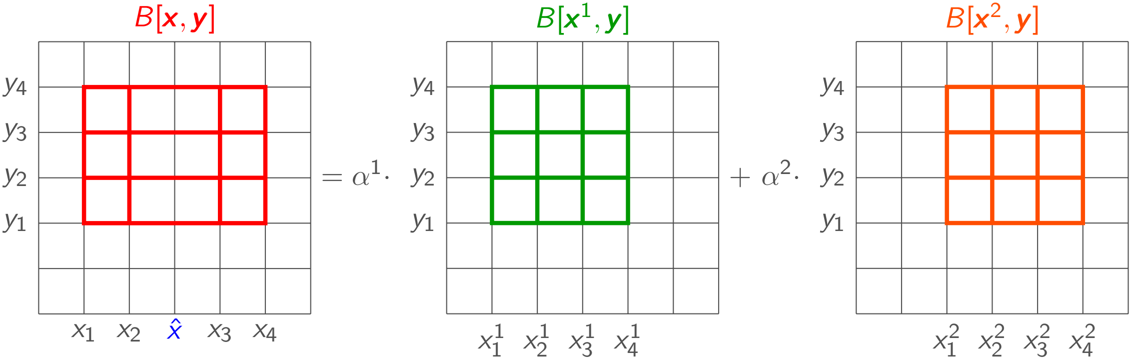
Assume $B[\mathbf{x}, \mathbf{y}]$ no minimal support on $\mathcal{N}$ because of a vertical line at $x = \hat{x}$ with $\hat{x} \notin \mathbf{x}$.



with $\alpha^1, \alpha^2 \in (0, 1]$.

Knot insertion

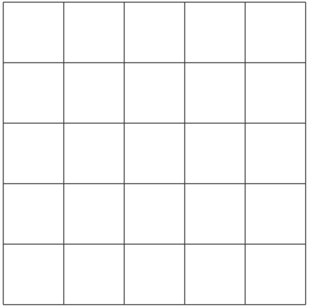
Assume $B[\mathbf{x},\mathbf{y}]$ no minimal support on $\mathcal{N}$ because of a vertical line at $x = \hat{x}$ with $\hat{x} \notin \mathbf{x}$.



with $\alpha^1, \alpha^2 \in (0,1]$. $B[\mathbf{x},\mathbf{y}]$ is expressed in terms of B-splines of minimal support on $\mathcal{N}$.

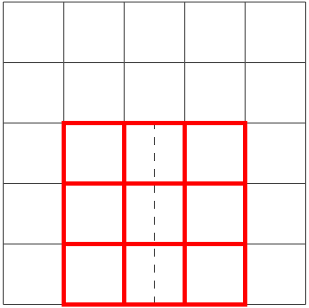
LR meshes and LR B-splines

Let $\mathcal{N}$ be a tensor mesh and $\mathcal{B}$ be the set of (minimal support) B-splines on $\mathcal{N}$.



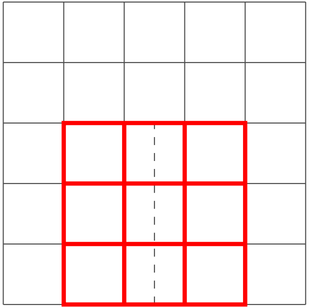
LR meshes and LR B-splines

Let $\mathcal{N}$ be a tensor mesh and $\mathcal{B}$ be the set of (minimal support) B-splines on $\mathcal{N}$.
We insert a new line γ, traversing the support of at least one B-spline $B \in \mathcal{B}$.



LR meshes and LR B-splines

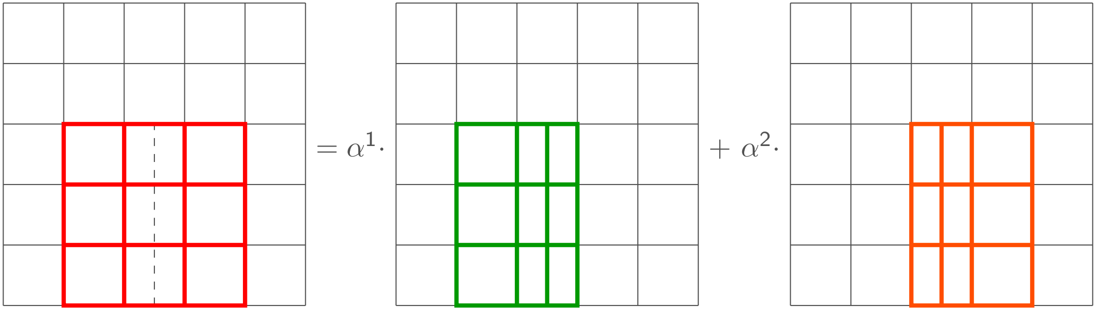
Let $\mathcal{N}$ be a tensor mesh and $\mathcal{B}$ be the set of (minimal support) B-splines on $\mathcal{N}$.
We insert a new line γ, traversing the support of at least one B-spline $B \in \mathcal{B}$.



By construction B has not minimal support on the new mesh $\mathcal{N}' = \mathcal{N} \cup \gamma$.

LR meshes and LR B-splines

Let $\mathcal{N}$ be a tensor mesh and $\mathcal{B}$ be the set of (minimal support) B-splines on $\mathcal{N}$.
We insert a new line γ, traversing the support of at least one B-spline $B \in \mathcal{B}$.



By construction B has not minimal support on the new mesh $\mathcal{N}' = \mathcal{N} \cup \gamma$. By knot insertion we replace B with the B-splines B^1 and B^2 of minimal support on $\mathcal{N}'$. This operation creates a new set $\mathcal{B}'$ of minimal support B-splines on $\mathcal{N}'$.

LR meshes and LR B-splines

LR mesh $\mathcal{N}'$ (recursive definition):

LR meshes and LR B-splines

LR mesh $\mathcal{N}'$ (recursive definition): It is either

LR meshes and LR B-splines

LR mesh $\mathcal{N}'$ (recursive definition): It is either

- a tensor mesh,
- obtained by insertion of a new line from the LR mesh $\mathcal{N}$, traversing at least one support.

LR meshes and LR B-splines

LR mesh $\mathcal{N}'$ (recursive definition): It is either

- a tensor mesh,
- obtained by insertion of a new line from the LR mesh $\mathcal{N}$, traversing at least one support.

LR B-spline set $\mathcal{B}'$ (recursive definition):

LR meshes and LR B-splines

LR mesh $\mathcal{N}'$ (recursive definition): It is either

- a tensor mesh,
- obtained by insertion of a new line from the LR mesh $\mathcal{N}$, traversing at least one support.

LR B-spline set $\mathcal{B}'$ (recursive definition): It is either

LR meshes and LR B-splines

LR mesh $\mathcal{N}'$ (recursive definition): It is either

- a tensor mesh,
- obtained by insertion of a new line from the LR mesh $\mathcal{N}$, traversing at least one support.

LR B-spline set $\mathcal{B}'$ (recursive definition): It is either

- the B-spline set on $\mathcal{N}'$ if $\mathcal{N}'$ is a tensor mesh,
- an update via knot insertion of the LR B-spline set $\mathcal{B}$ on $\mathcal{N}$.

LR meshes and LR B-splines

LR mesh $\mathcal{N}'$ (recursive definition): It is either

- a tensor mesh,
- obtained by insertion of a new line from the LR mesh $\mathcal{N}$, traversing at least one support.

LR B-spline set $\mathcal{B}'$ (recursive definition): It is either

- the B-spline set on $\mathcal{N}'$ if $\mathcal{N}'$ is a tensor mesh,
- an update via knot insertion of the LR B-spline set $\mathcal{B}$ on $\mathcal{N}$.

LR mesh $\mathcal{N}'$ (iterative definition):

LR meshes and LR B-splines

LR mesh $\mathcal{N}'$ (recursive definition): It is either

- a tensor mesh,
- obtained by insertion of a new line from the LR mesh $\mathcal{N}$, traversing at least one support.

LR B-spline set $\mathcal{B}'$ (recursive definition): It is either

- the B-spline set on $\mathcal{N}'$ if $\mathcal{N}'$ is a tensor mesh,
- an update via knot insertion of the LR B-spline set $\mathcal{B}$ on $\mathcal{N}$.

LR mesh $\mathcal{N}'$ (iterative definition): $\mathcal{N}' = \mathcal{N}_N$ with

LR meshes and LR B-splines

LR mesh $\mathcal{N}'$ (recursive definition): It is either

- a tensor mesh,
- obtained by insertion of a new line from the LR mesh $\mathcal{N}$, traversing at least one support.

LR B-spline set $\mathcal{B}'$ (recursive definition): It is either

- the B-spline set on $\mathcal{N}'$ if $\mathcal{N}'$ is a tensor mesh,
- an update via knot insertion of the LR B-spline set $\mathcal{B}$ on $\mathcal{N}$.

LR mesh $\mathcal{N}'$ (iterative definition): $\mathcal{N}' = \mathcal{N}_N$ with

$$\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$$

LR meshes and LR B-splines

LR mesh $\mathcal{N}'$ (recursive definition): It is either

- a tensor mesh,
- obtained by insertion of a new line from the LR mesh $\mathcal{N}$, traversing at least one support.

LR B-spline set $\mathcal{B}'$ (recursive definition): It is either

- the B-spline set on $\mathcal{N}'$ if $\mathcal{N}'$ is a tensor mesh,
- an update via knot insertion of the LR B-spline set $\mathcal{B}$ on $\mathcal{N}$.

LR mesh $\mathcal{N}'$ (iterative definition): $\mathcal{N}' = \mathcal{N}_N$ with

$$\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$$

LR B-spline set $\mathcal{B}'$ (iterative definition):

LR meshes and LR B-splines

LR mesh $\mathcal{N}'$ (recursive definition): It is either

- a tensor mesh,
- obtained by insertion of a new line from the LR mesh $\mathcal{N}$, traversing at least one support.

LR B-spline set $\mathcal{B}'$ (recursive definition): It is either

- the B-spline set on $\mathcal{N}'$ if $\mathcal{N}'$ is a tensor mesh,
- an update via knot insertion of the LR B-spline set $\mathcal{B}$ on $\mathcal{N}$.

LR mesh $\mathcal{N}'$ (iterative definition): $\mathcal{N}' = \mathcal{N}_N$ with

$$\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$$

LR B-spline set $\mathcal{B}'$ (iterative definition): $\mathcal{B}' = \mathcal{B}_N$ with

LR meshes and LR B-splines

LR mesh $\mathcal{N}'$ (recursive definition): It is either

- a tensor mesh,
- obtained by insertion of a new line from the LR mesh $\mathcal{N}$, traversing at least one support.

LR B-spline set $\mathcal{B}'$ (recursive definition): It is either

- the B-spline set on $\mathcal{N}'$ if $\mathcal{N}'$ is a tensor mesh,
- an update via knot insertion of the LR B-spline set $\mathcal{B}$ on $\mathcal{N}$.

LR mesh $\mathcal{N}'$ (iterative definition): $\mathcal{N}' = \mathcal{N}_N$ with

$$\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$$

LR B-spline set $\mathcal{B}'$ (iterative definition): $\mathcal{B}' = \mathcal{B}_N$ with

$$\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \backslash \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$$

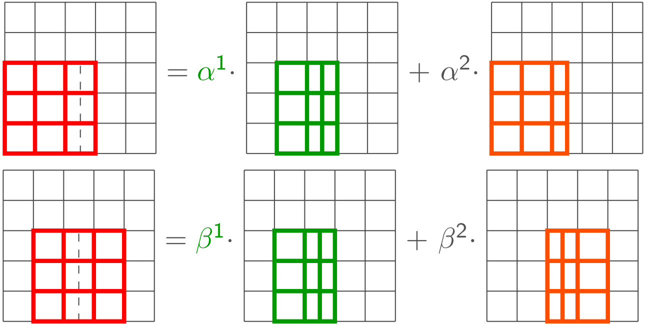
with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.

LR meshes and LR B-splines

Partition of Unity Weights: sum of the knot insertion coefficients

LR meshes and LR B-splines

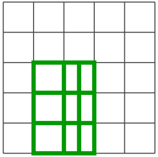
Partition of Unity Weights: sum of the knot insertion coefficients



$= \alpha^1 \cdot \quad + \alpha^2 \cdot$

$= \beta^1 \cdot \quad + \beta^2 \cdot$

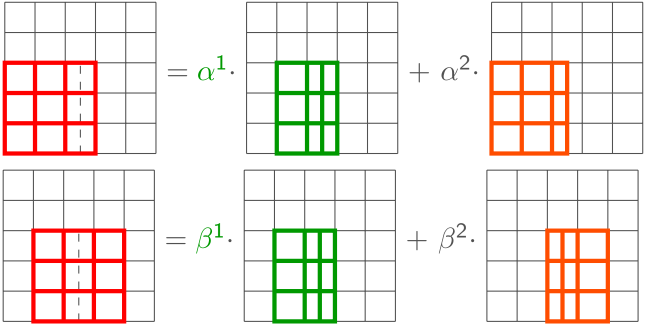
The PoU weight for

$B =$ 

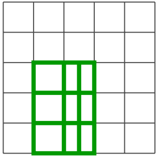
is $w_B = \alpha^1 + \beta^1$.

LR meshes and LR B-splines

Partition of Unity Weights: sum of the knot insertion coefficients



The PoU weight for

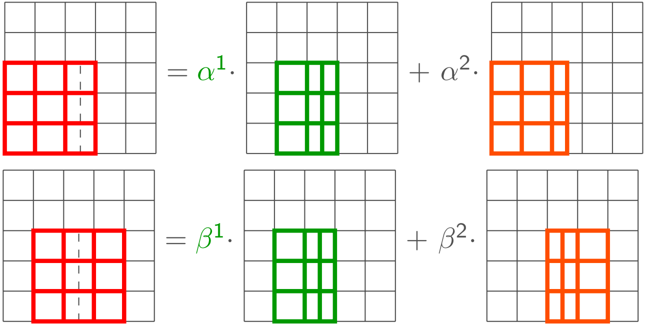
$B =$ 

is $w_B = \alpha^1 + \beta^1$.

Remark: If not specified otherwise, we consider internal meshlines of multiplicity 1 and boundary meshlines of multiplicity $p_k + 1$, for $k = 1, 2$, for vertical and horizontal meshlines respectively.

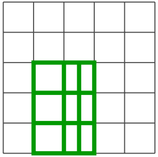
LR meshes and LR B-splines

Partition of Unity Weights: sum of the knot insertion coefficients



The PoU weight for

$B =$



is $w_B = \alpha^1 + \beta^1$.

Remark: If not specified otherwise, we consider internal meshlines of multiplicity 1 and boundary meshlines of multiplicity $p_k + 1$, for $k = 1, 2$, for vertical and horizontal meshlines respectively.

Remark: Meshline insertion ordering can often be changed. However, the final LR B-spline set is well defined because independent of such insertion ordering.

LR meshes and LR B-splines

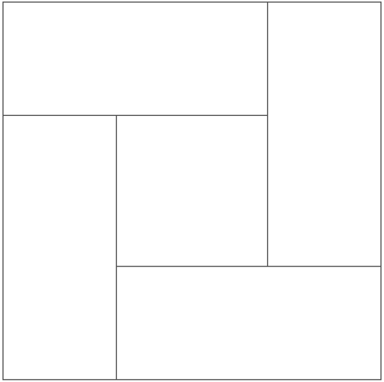
Remark: Not all meshes with local lines are LR meshes

LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

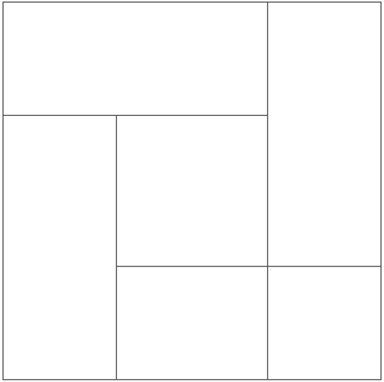
LR meshes and LR B-splines

Remark: Not all meshes with local lines are LR meshes

LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$



Not LR mesh



LR mesh

LR meshes and LR B-splines

Remark: Secondary splits may be needed.

LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \backslash \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.

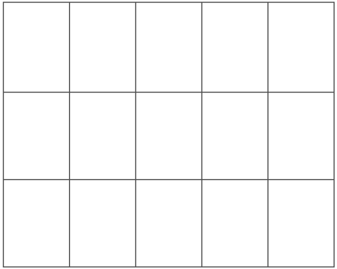
LR meshes and LR B-splines

Remark: Secondary splits may be needed.

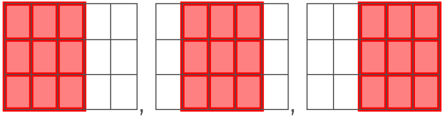
LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \backslash \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.



(a) current mesh



(b) $\mathcal{B}_0$

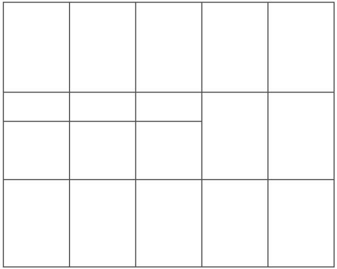
LR meshes and LR B-splines

Remark: Secondary splits may be needed.

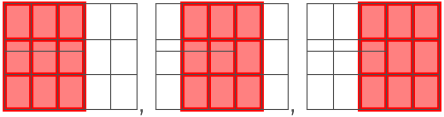
LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \backslash \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.



(a) current mesh



(b) $\mathcal{B}_0$

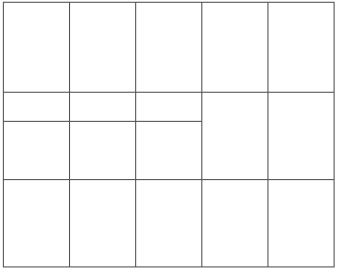
LR meshes and LR B-splines

Remark: Secondary splits may be needed.

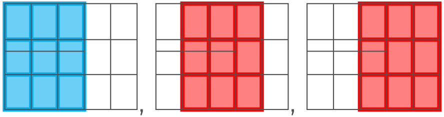
LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \backslash \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.



(a) current mesh



(b) $\mathcal{B}_0$

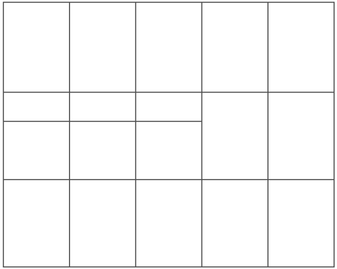
LR meshes and LR B-splines

Remark: Secondary splits may be needed.

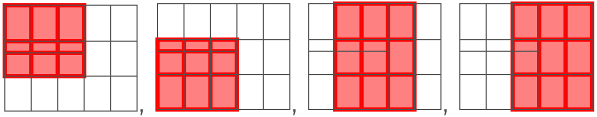
LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \setminus \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.



(a) current mesh



(b) $\mathcal{B}_1$

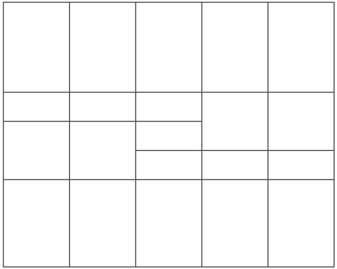
LR meshes and LR B-splines

Remark: Secondary splits may be needed.

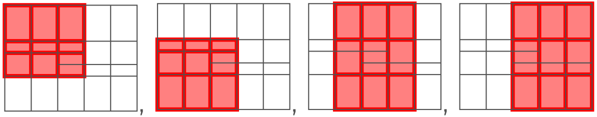
LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \setminus \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.



(a) current mesh



(b) $\mathcal{B}_1$

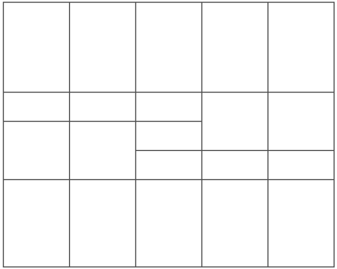
LR meshes and LR B-splines

Remark: Secondary splits may be needed.

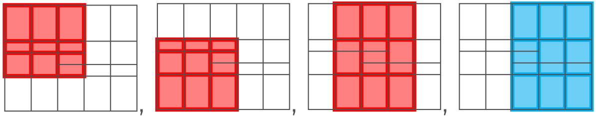
LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \backslash \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.



(a) current mesh



(b) $\mathcal{B}_1$

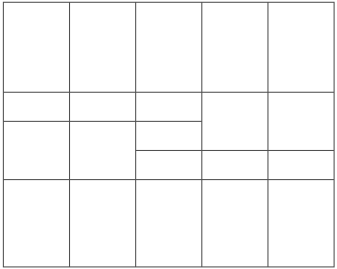
LR meshes and LR B-splines

Remark: Secondary splits may be needed.

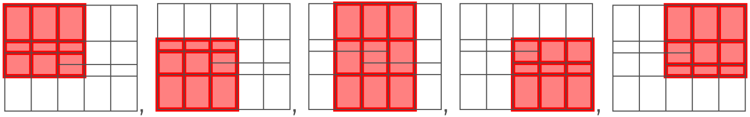
LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \backslash \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.



(a) current mesh



(b) $\mathcal{B}_2$

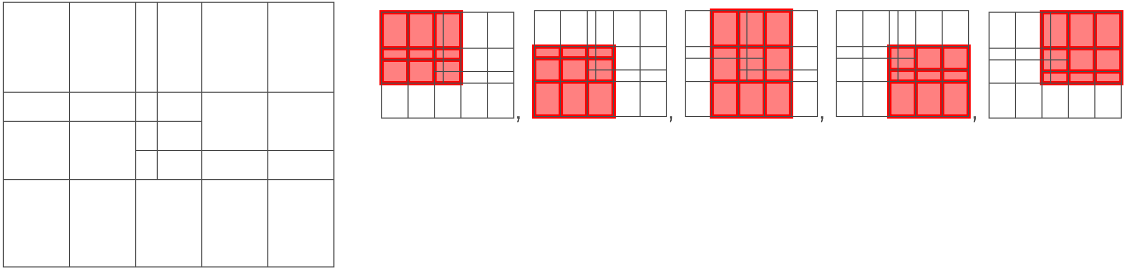
LR meshes and LR B-splines

Remark: Secondary splits may be needed.

LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \backslash \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.



(a) current mesh

(b) $\mathcal{B}_2$

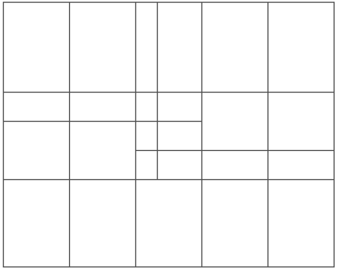
LR meshes and LR B-splines

Remark: Secondary splits may be needed.

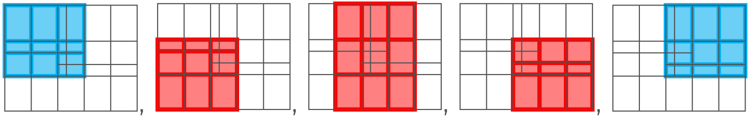
LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \backslash \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.



(a) current mesh



(b) $\mathcal{B}_2$

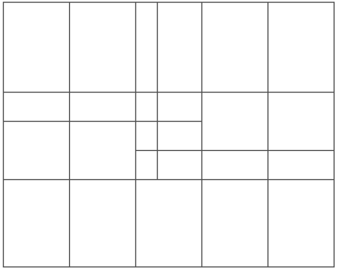
LR meshes and LR B-splines

Remark: Secondary splits may be needed.

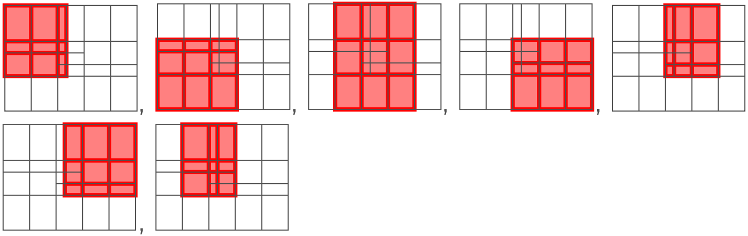
LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \backslash \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.



(a) current mesh



(b) $\mathcal{B}_3$

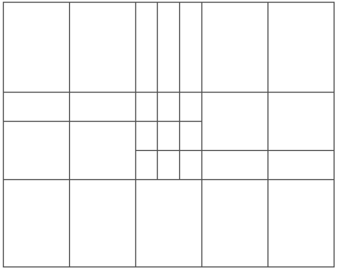
LR meshes and LR B-splines

Remark: Secondary splits may be needed.

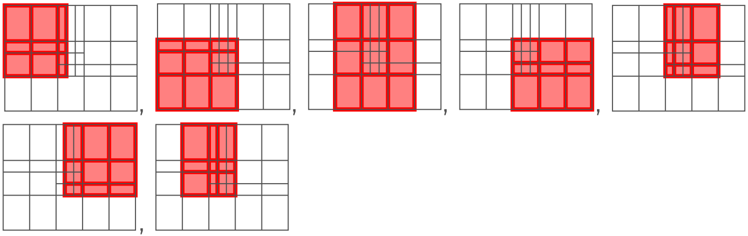
LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \backslash \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.



(a) current mesh



(b) $\mathcal{B}_3$

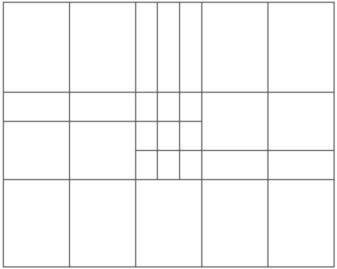
LR meshes and LR B-splines

Remark: Secondary splits may be needed.

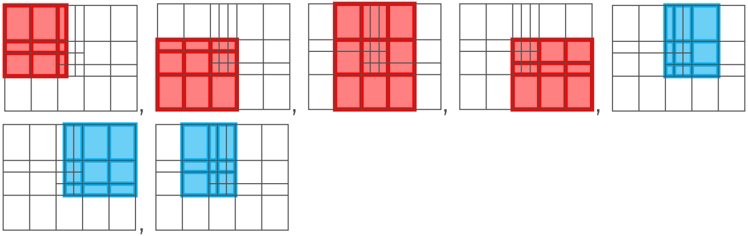
LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \backslash \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.



(a) current mesh



(b) $\mathcal{B}_3$

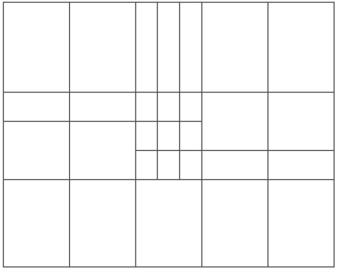
LR meshes and LR B-splines

Remark: Secondary splits may be needed.

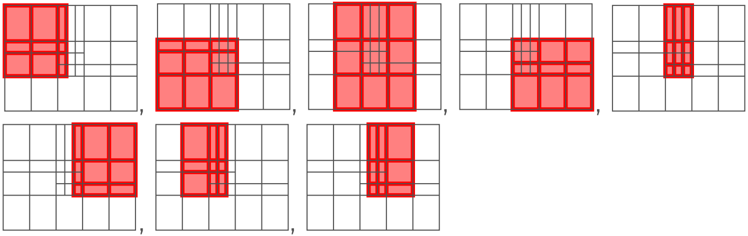
LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \backslash \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.



(a) current mesh



(b) $\mathcal{B}_3$

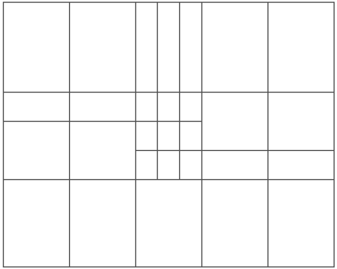
LR meshes and LR B-splines

Remark: Secondary splits may be needed.

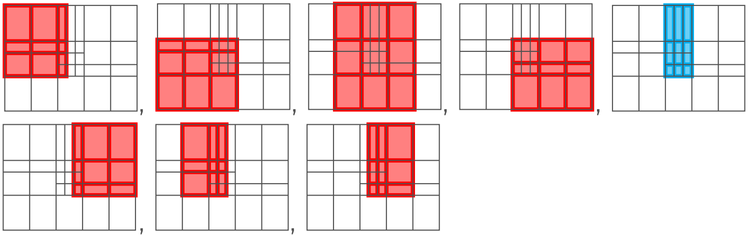
LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \backslash \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.



(a) current mesh



(b) $\mathcal{B}_3$

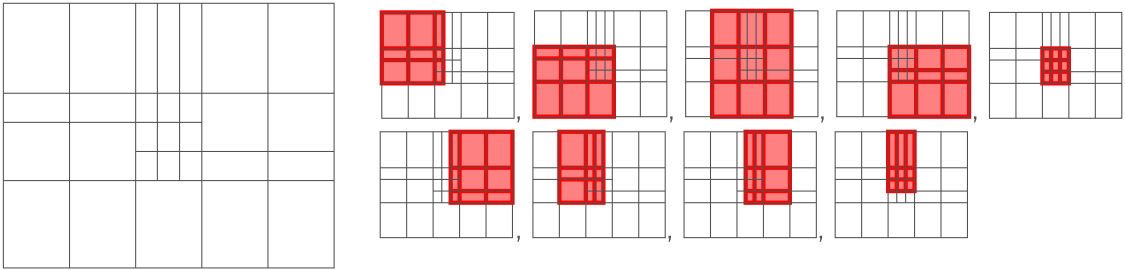
LR meshes and LR B-splines

Remark: Secondary splits may be needed.

LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \backslash \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.



(a) current mesh **(b)** $\mathcal{B}_4$

LR meshes and LR B-splines

Remark: LR B-spline set $\neq$ Minimal Support B-spline set.

LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \backslash \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.

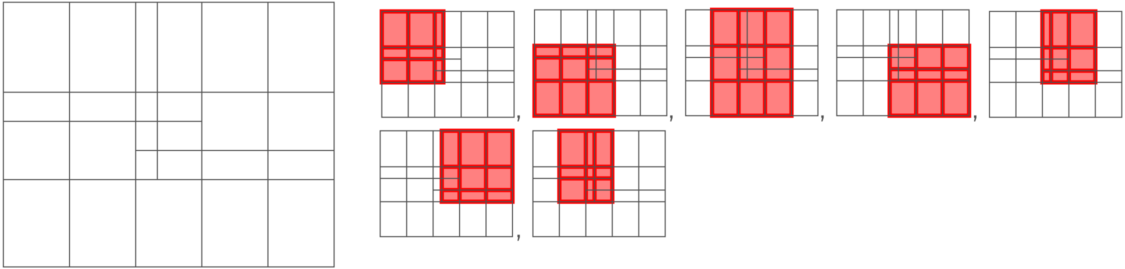
LR meshes and LR B-splines

Remark: LR B-spline set $\neq$ Minimal Support B-spline set.

LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \setminus \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.



(a) current mesh

(b) $\mathcal{B}_3$

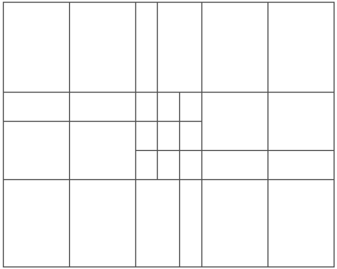
LR meshes and LR B-splines

Remark: LR B-spline set $\neq$ Minimal Support B-spline set.

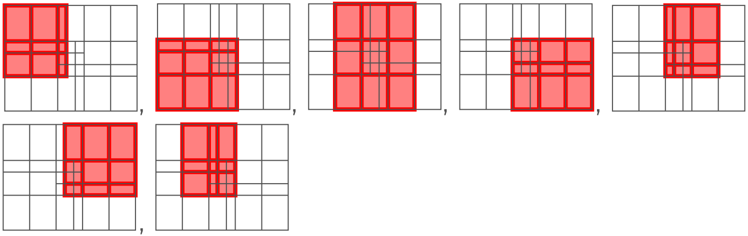
LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \backslash \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.



(a) current mesh



(b) $\mathcal{B}_3$

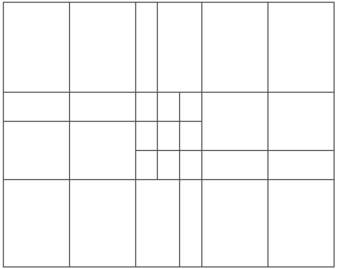
LR meshes and LR B-splines

Remark: LR B-spline set $\neq$ Minimal Support B-spline set.

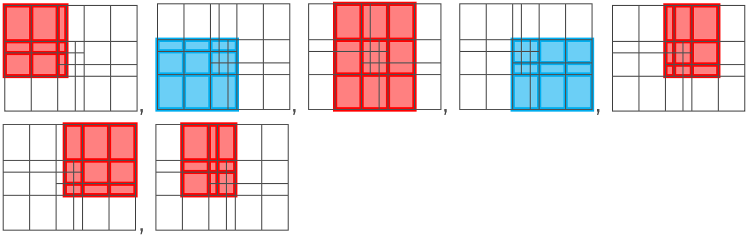
LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \backslash \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.



(a) current mesh



(b) $\mathcal{B}_3$

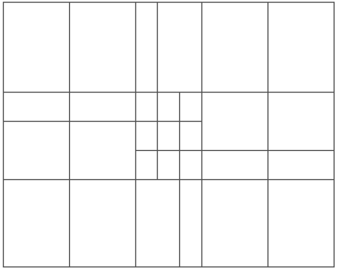
LR meshes and LR B-splines

Remark: LR B-spline set $\neq$ Minimal Support B-spline set.

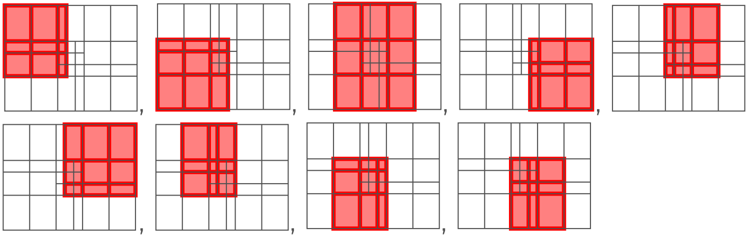
LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \backslash \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.



(a) current mesh



(b) $\mathcal{B}_4$

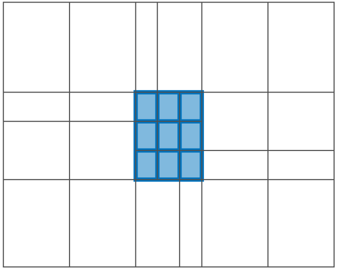
LR meshes and LR B-splines

Remark: LR B-spline set $\subsetneq$ Minimal Support B-spline set.

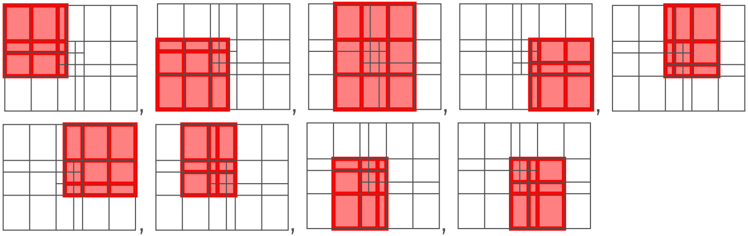
LR mesh: $\begin{cases} \mathcal{N}_{i+1} = \mathcal{N}_i \cup \gamma_i & \gamma_i \text{ new line, traversing at least one support} \\ \mathcal{N}_0 & \text{tensor mesh,} \end{cases}$

LR B-splines: $\begin{cases} \mathcal{B}_{i+1} = (\mathcal{B}_i \backslash \mathcal{B}_i(\gamma_i)) + \mathcal{K}(\mathcal{B}_i(\gamma_i)) & \mathcal{B}_i(\gamma_i) := \text{ B-splines in } \mathcal{B}_i \text{ traversed by } \gamma_i \\ \mathcal{B}_0 & \text{tensor B-splines on } \mathcal{N}_0 \end{cases}$

with $\mathcal{K}$ all the refinements via knot insertion needed to have minimal supports.



(a) current mesh



(b) $\mathcal{B}_4$

Minimum Span Refinement strategy

Input: Bunch of boxes where a larger error is committed is some sense.
For each of such boxes

Minimum Span Refinement strategy

Input: Bunch of boxes where a larger error is committed is some sense.
For each of such boxes

1. Among all the LR B-splines on that box, select those with smallest support (semi-perimeter),

Minimum Span Refinement strategy

Input: Bunch of boxes where a larger error is committed is some sense.
For each of such boxes

1. Among all the LR B-splines on that box, select those with smallest support (semi-perimeter),
2. Pick one randomly and insert a cross centered at the box to split it in 4.

Minimum Span Refinement strategy

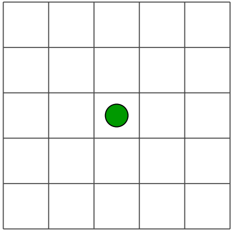
Input: Bunch of boxes where a larger error is committed is some sense.
For each of such boxes

1. Among all the LR B-splines on that box, select those with smallest support (semi-perimeter),
2. Pick one randomly and insert a cross centered at the box to split it in 4.

Minimum Span Refinement strategy

Input: Bunch of boxes where a larger error is committed is some sense.
For each of such boxes

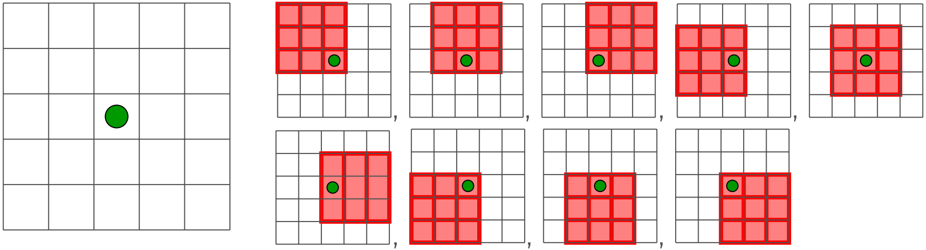
1. Among all the LR B-splines on that box, select those with smallest support (semi-perimeter),
2. Pick one randomly and insert a cross centered at the box to split it in 4.



Minimum Span Refinement strategy

Input: Bunch of boxes where a larger error is committed is some sense.
For each of such boxes

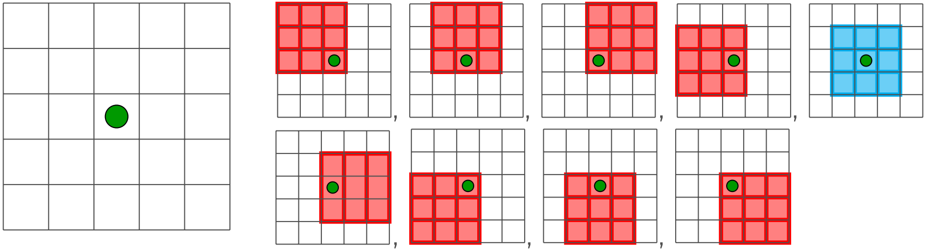
1. Among all the LR B-splines on that box, select those with smallest support (semi-perimeter),
2. Pick one randomly and insert a cross centered at the box to split it in 4.



Minimum Span Refinement strategy

Input: Bunch of boxes where a larger error is committed is some sense.
For each of such boxes

1. Among all the LR B-splines on that box, select those with smallest support (semi-perimeter),
2. Pick one randomly and insert a cross centered at the box to split it in 4.

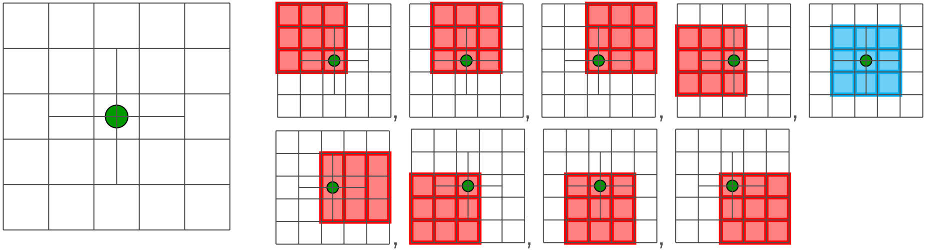


Minimum Span Refinement strategy

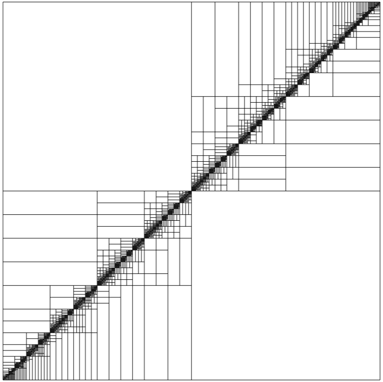
Input: Bunch of boxes where a larger error is committed is some sense.

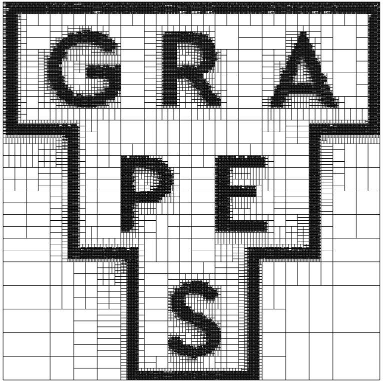
For each of such boxes

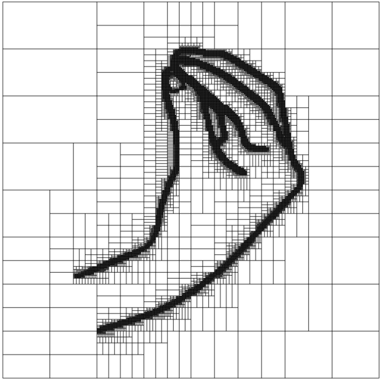
1. Among all the LR B-splines on that box, select those with smallest support (semi-perimeter),
2. Pick one randomly and insert a cross centered at the box to split it in 4.



Minimum Span Refinement strategy: Examples







Full Span Refinement strategy

Input: Bunch of boxes where a larger error is committed is some sense.
For each of such boxes

Full Span Refinement strategy

Input: Bunch of boxes where a larger error is committed is some sense.
For each of such boxes

1. Select all the LR B-splines on that box,

Full Span Refinement strategy

Input: Bunch of boxes where a larger error is committed is some sense.
For each of such boxes

1. Select all the LR B-splines on that box,
2. Insert a cross centered at the box and long enough to traverse all of such LR B-splines.

Full Span Refinement strategy

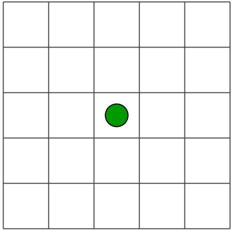
Input: Bunch of boxes where a larger error is committed is some sense.
For each of such boxes

1. Select all the LR B-splines on that box,
2. Insert a cross centered at the box and long enough to traverse all of such LR B-splines.

Full Span Refinement strategy

Input: Bunch of boxes where a larger error is committed is some sense.
For each of such boxes

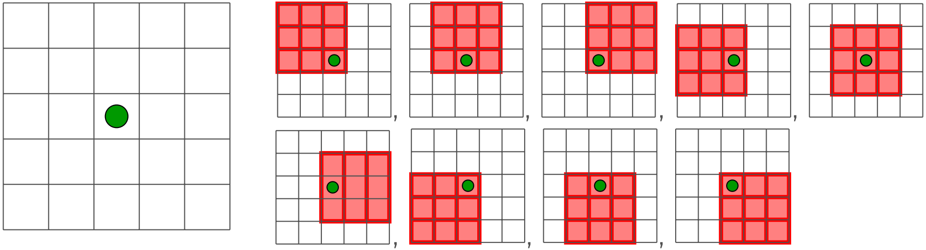
1. Select all the LR B-splines on that box,
2. Insert a cross centered at the box and long enough to traverse all of such LR B-splines.



Full Span Refinement strategy

Input: Bunch of boxes where a larger error is committed is some sense.
For each of such boxes

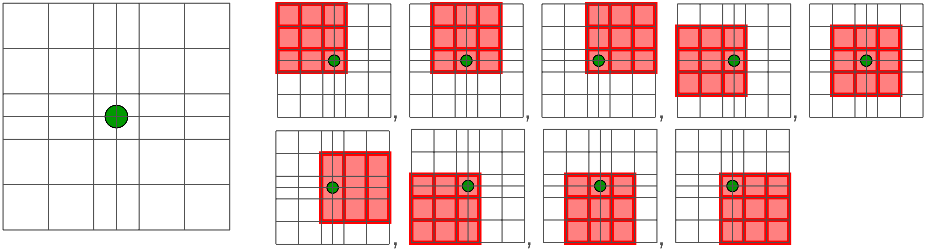
1. Select all the LR B-splines on that box,
2. Insert a cross centered at the box and long enough to traverse all of such LR B-splines.



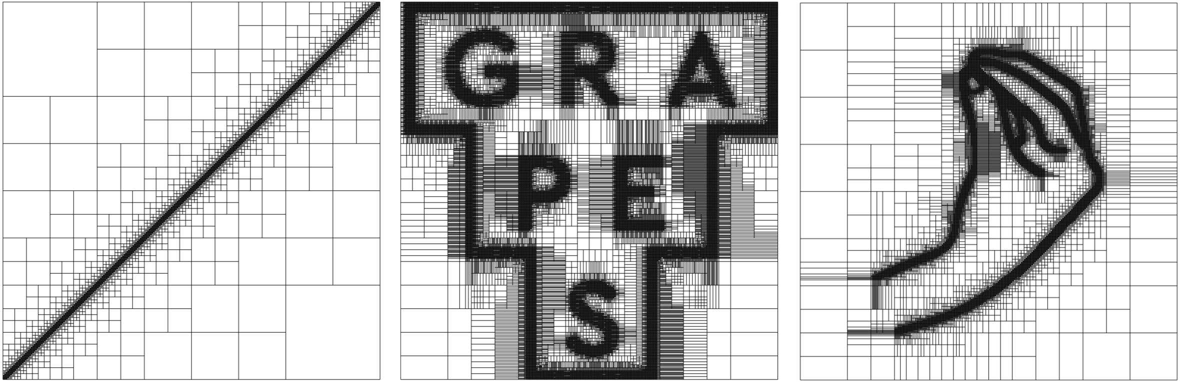
Full Span Refinement strategy

Input: Bunch of boxes where a larger error is committed is some sense.
For each of such boxes

1. Select all the LR B-splines on that box,
2. Insert a cross centered at the box and long enough to traverse all of such LR B-splines.



Full Span Refinement strategy: Examples



Structured Mesh Refinement strategy

Input: Bunch of LR B-splines where a larger error is committed is some sense.

For each of such LR B-splines

Structured Mesh Refinement strategy

Input: Bunch of LR B-splines where a larger error is committed is some sense.
For each of such LR B-splines

1. Split in 4 all the boxes in their tensor mesh

Structured Mesh Refinement strategy

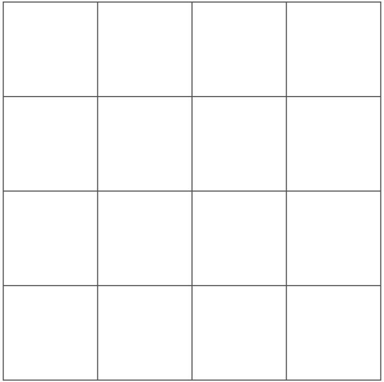
Input: Bunch of LR B-splines where a larger error is committed is some sense.
For each of such LR B-splines

1. Split in 4 all the boxes in their tensor mesh

Structured Mesh Refinement strategy

Input: Bunch of LR B-splines where a larger error is committed is some sense.
For each of such LR B-splines

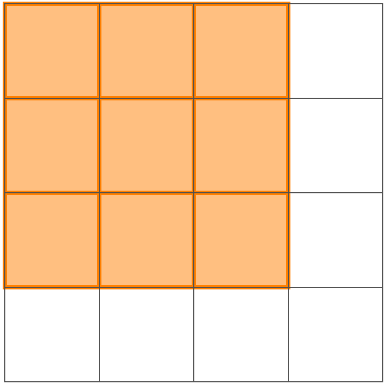
1. Split in 4 all the boxes in their tensor mesh



Structured Mesh Refinement strategy

Input: Bunch of LR B-splines where a larger error is committed is some sense.
For each of such LR B-splines

1. Split in 4 all the boxes in their tensor mesh

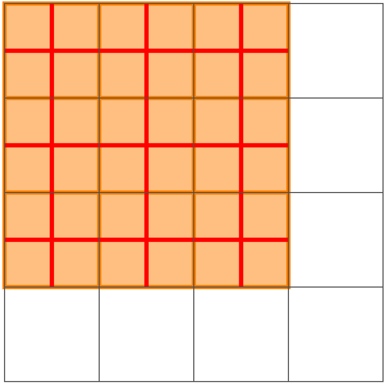


Structured Mesh Refinement strategy

Input: Bunch of LR B-splines where a larger error is committed is some sense.

For each of such LR B-splines

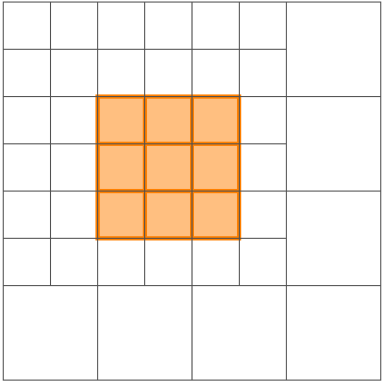
1. Split in 4 all the boxes in their tensor mesh



Structured Mesh Refinement strategy

Input: Bunch of LR B-splines where a larger error is committed is some sense.
For each of such LR B-splines

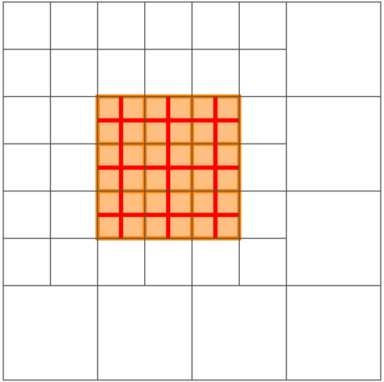
1. Split in 4 all the boxes in their tensor mesh



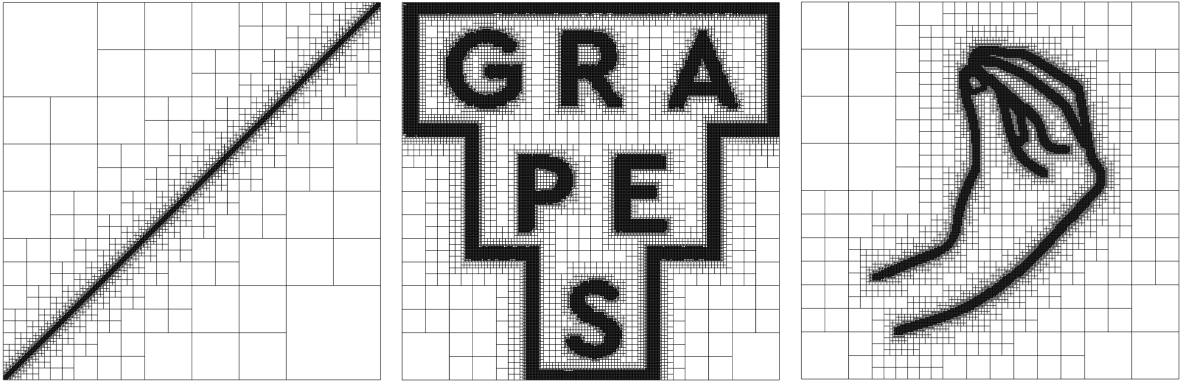
Structured Mesh Refinement strategy

Input: Bunch of LR B-splines where a larger error is committed is some sense.
For each of such LR B-splines

1. Split in 4 all the boxes in their tensor mesh



Structured Mesh Refinement strategy: Examples

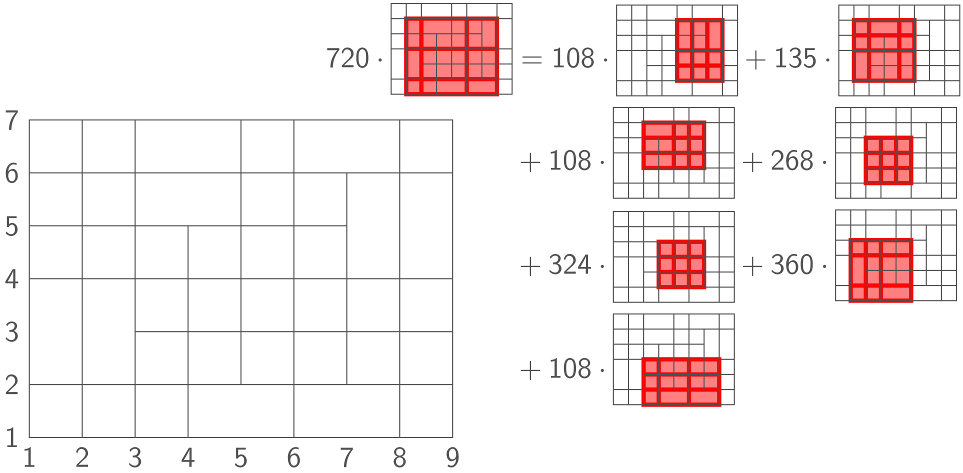


The Linear Dependence Problem

Unfortunately, linear dependence relations may arise in the LR B-spline set.

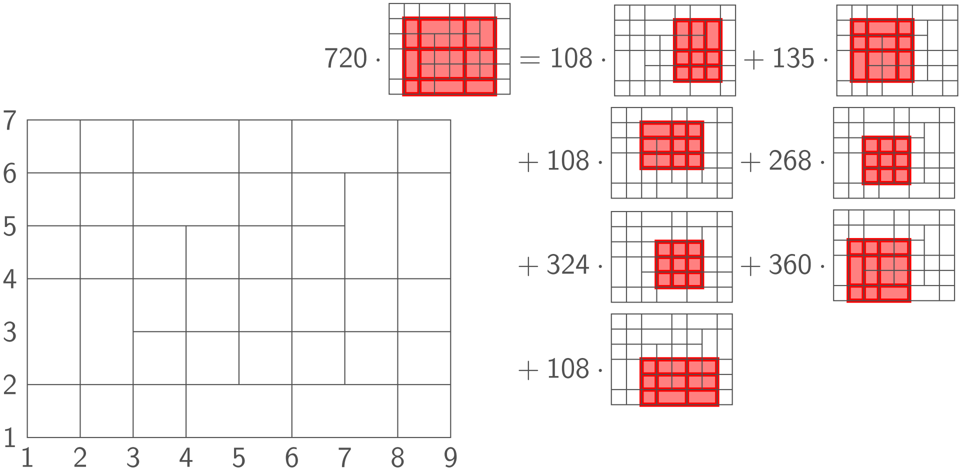
The Linear Dependence Problem

Unfortunately, linear dependence relations may arise in the LR B-spline set.



The Linear Dependence Problem

Unfortunately, linear dependence relations may arise in the LR B-spline set.



Minimum Span, Full Span and Structured Mesh may have linear dependence

Conjecture: the latter only for $(p_1, p_2) \geq (4, 4)$.

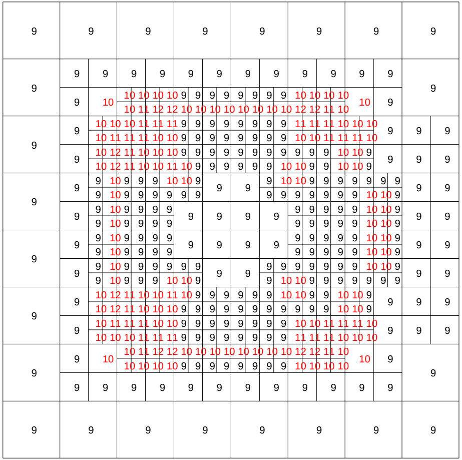
Seek and Destroy Linear Dependence: Peeling Algorithm

Remark: In every box we span the polynomial space $\Pi_{\boldsymbol{p}}$ $\Rightarrow$ each box is in at least $(p_1 + 1)(p_2 + 1) = \dim \Pi_{\boldsymbol{p}}$ LR B-spline supports.

Seek and Destroy Linear Dependence: Peeling Algorithm

Remark: In every box we span the polynomial space $\Pi_{\boldsymbol{p}}$ $\Rightarrow$ each box is in at least $(p_1+1)(p_2+1) = \dim \Pi_{\boldsymbol{p}}$ LR B-spline supports.

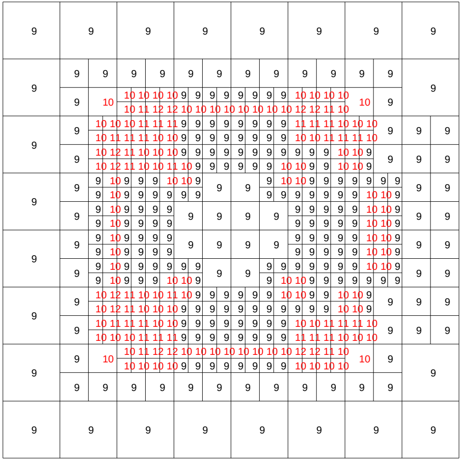
Overloaded box: box contained in more than $(p_1+1)(p_2+1)$ LR B-spline supports.



Seek and Destroy Linear Dependence: Peeling Algorithm

Remark: In every box we span the polynomial space $\Pi_{\boldsymbol{p}} \Rightarrow$ each box is in at least $(p_1+1)(p_2+1) = \dim \Pi_{\boldsymbol{p}}$ LR B-spline supports.

Overloaded box: box contained in more than $(p_1+1)(p_2+1)$ LR B-spline supports.

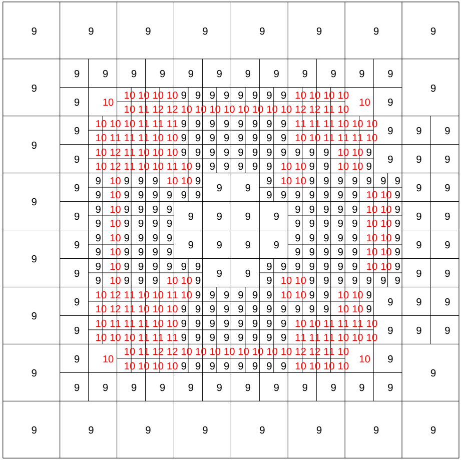


Overloaded LR B-spline: all the boxes in its support are overloaded.

Seek and Destroy Linear Dependence: Peeling Algorithm

Remark: In every box we span the polynomial space $\Pi_{\boldsymbol{p}} \Rightarrow$ each box is in at least $(p_1+1)(p_2+1) = \dim \Pi_{\boldsymbol{p}}$ LR B-spline supports.

Overloaded box: box contained in more than $(p_1+1)(p_2+1)$ LR B-spline supports.



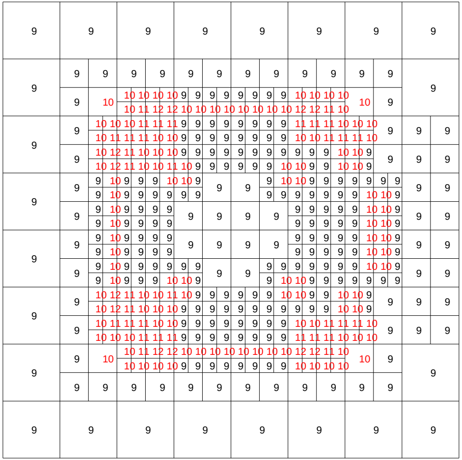
Overloaded LR B-spline: all the boxes in its support are overloaded.

💡 Only overloaded LR B-splines can be in a linear dependence relation.

Seek and Destroy Linear Dependence: Peeling Algorithm

Remark: In every box we span the polynomial space $\Pi_{\boldsymbol{p}}$ $\Rightarrow$ each box is in at least $(p_1+1)(p_2+1) = \dim \Pi_{\boldsymbol{p}}$ LR B-spline supports.

Overloaded box: box contained in more than $(p_1+1)(p_2+1)$ LR B-spline supports.



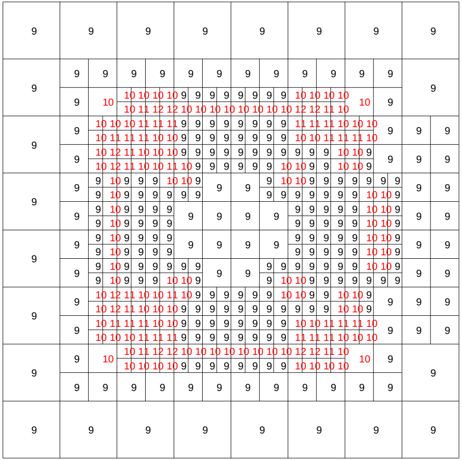
Overloaded LR B-spline: all the boxes in its support are overloaded.

- Only overloaded LR B-splines can be in a linear dependence relation.
- A linear dependence relation needs at least 2 functions.

Seek and Destroy Linear Dependence: Peeling Algorithm

Remark: In every box we span the polynomial space $\Pi_{\boldsymbol{p}}$ $\Rightarrow$ each box is in at least $(p_1+1)(p_2+1) = \dim \Pi_{\boldsymbol{p}}$ LR B-spline supports.

Overloaded box: box contained in more than $(p_1+1)(p_2+1)$ LR B-spline supports.



Overloaded LR B-spline: all the boxes in its support are overloaded.

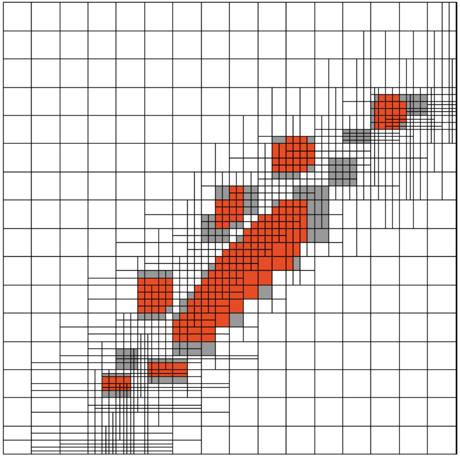
- Only overloaded LR B-splines can be in a linear dependence relation.
- A linear dependence relation needs at least 2 functions.

$\Rightarrow$ If a box is just in one overloaded LR B-spline B, then B cannot be part of a linear dependence relation.

Seek and Destroy Linear Dependence: Peeling Algorithm

Peeling Algorithm

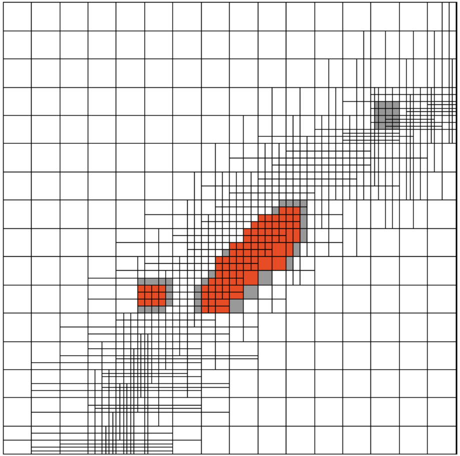
1 Create the set $\mathcal{B}^O$ of overloaded LR B-splines;
2 Let $\mathcal{E}^O$ be the boxes of the LR B-splines in $\mathcal{B}^O$;
3 **for** *every box in* $\mathcal{E}^O$ **do**
4   Identify the set $\mathcal{B}_1^O \subseteq \mathcal{B}^O$ of those having a box covered by no other LR B-splines in $\mathcal{B}^O$;
5 **if** $\mathcal{B}^O \backslash \mathcal{B}_1^O = \emptyset$ **then**
6   linear independence, break.
7 **else**
8   **if** $\mathcal{B}_1^O = \emptyset$ **then**
9   break, but might have linear dependence
10     $\mathcal{B}^O \leftarrow \mathcal{B}^O \backslash \mathcal{B}_1^O$;
11   Go to 2;



Seek and Destroy Linear Dependence: Peeling Algorithm

Peeling Algorithm

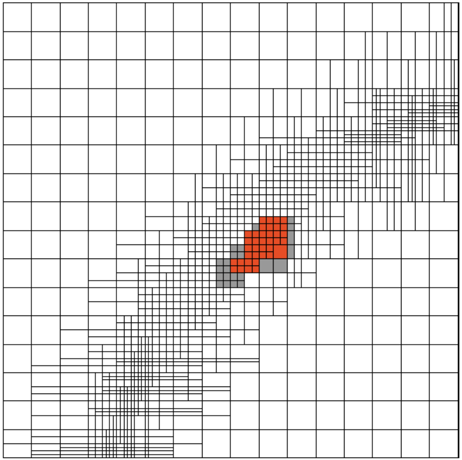
1 Create the set $\mathcal{B}^O$ of overloaded LR B-splines;
2 Let $\mathcal{E}^O$ be the boxes of the LR B-splines in $\mathcal{B}^O$;
3 **for** *every box in* $\mathcal{E}^O$ **do**
4 Identify the set $\mathcal{B}_1^O \subseteq \mathcal{B}^O$ of those having a box covered by no other LR B-splines in $\mathcal{B}^O$;
5 **if** $\mathcal{B}^O \backslash \mathcal{B}_1^O = \emptyset$ **then**
6 linear independence, break.
7 **else**
8 **if** $\mathcal{B}_1^O = \emptyset$ **then**
9 break, but might have linear dependence
10 $\mathcal{B}^O \leftarrow \mathcal{B}^O \backslash \mathcal{B}_1^O$;
11 Go to 2;



Seek and Destroy Linear Dependence: Peeling Algorithm

Peeling Algorithm

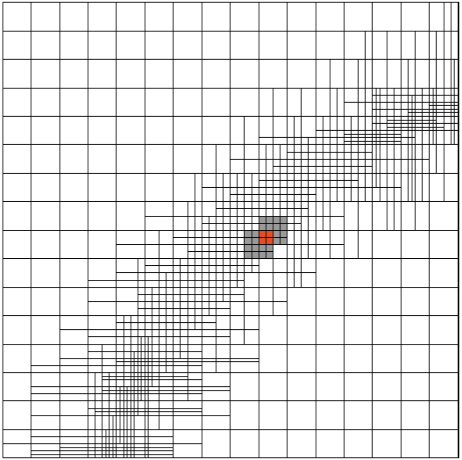
1 Create the set $\mathcal{B}^O$ of overloaded LR B-splines;
2 Let $\mathcal{E}^O$ be the boxes of the LR B-splines in $\mathcal{B}^O$;
3 **for** *every box in* $\mathcal{E}^O$ **do**
4   Identify the set $\mathcal{B}_1^O \subseteq \mathcal{B}^O$ of those having a box covered by no other LR B-splines in $\mathcal{B}^O$;
5 **if** $\mathcal{B}^O \backslash \mathcal{B}_1^O = \emptyset$ **then**
6   linear independence, break.
7 **else**
8   **if** $\mathcal{B}_1^O = \emptyset$ **then**
9   break, but might have linear dependence
10     $\mathcal{B}^O \leftarrow \mathcal{B}^O \backslash \mathcal{B}_1^O$;
11   Go to 2;



Seek and Destroy Linear Dependence: Peeling Algorithm

Peeling Algorithm

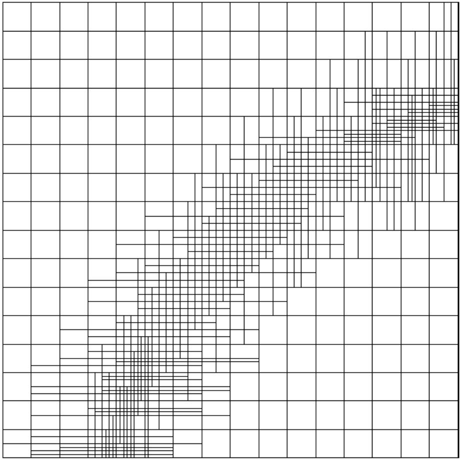
1 Create the set $\mathcal{B}^O$ of overloaded LR B-splines;
2 Let $\mathcal{E}^O$ be the boxes of the LR B-splines in $\mathcal{B}^O$;
3 **for** *every box in* $\mathcal{E}^O$ **do**
4   Identify the set $\mathcal{B}_1^O \subseteq \mathcal{B}^O$ of those having a box covered by no other LR B-splines in $\mathcal{B}^O$;
5 **if** $\mathcal{B}^O \backslash \mathcal{B}_1^O = \emptyset$ **then**
6   linear independence, break.
7 **else**
8   **if** $\mathcal{B}_1^O = \emptyset$ **then**
9   break, but might have linear dependence
10     $\mathcal{B}^O \leftarrow \mathcal{B}^O \backslash \mathcal{B}_1^O$;
11   Go to 2;



Seek and Destroy Linear Dependence: Peeling Algorithm

Peeling Algorithm

```
1  Create the set 𝓑^O of overloaded LR B-splines;
2  Let 𝓔^O be the boxes of the LR B-splines in 𝓑^O;
3  for every box in 𝓔^O do
4  |   Identify the set 𝓑_1^O ⊆ 𝓑^O of those having a box
   |     covered by no other LR B-splines in 𝓑^O;
5  if 𝓑^O \ 𝓑_1^O = ∅ then
6  |   linear independence, break.
7  else
8  |   if 𝓑_1^O = ∅ then
9  |   |   break, but might have linear dependence
10 |   𝓑^O ← 𝓑^O \ 𝓑_1^O;
11 |   Go to 2;
```



Seek and Destroy Linear Dependence: Peeling Algorithm++

Peeling Algorithm

1 Create the set $\mathcal{B}^O$ of overloaded LR B-splines;
2 Let $\mathcal{E}^O$ be the boxes of the LR B-splines in $\mathcal{B}^O$;
3 **for** *every box in* $\mathcal{E}^O$ **do**
4   Identify the set $\mathcal{B}_1^O \subseteq \mathcal{B}^O$ of those having a box covered by no other LR B-splines in $\mathcal{B}^O$;
5 **if** $\mathcal{B}^O \backslash \mathcal{B}_1^O = \emptyset$ **then**
6   linear independence, break.
7 **else**
8   **if** $\mathcal{B}_1^O = \emptyset$ **then**
9   break, but might have linear dependence
10     $\mathcal{B}^O \leftarrow \mathcal{B}^O \backslash \mathcal{B}_1^O$;
11   Go to 2;

Seek and Destroy Linear Dependence: Peeling Algorithm++

Peeling Algorithm

1 Create the set $\mathcal{B}^O$ of overloaded LR B-splines;
2 Let $\mathcal{E}^O$ be the boxes of the LR B-splines in $\mathcal{B}^O$;
3 **for** *every box in* $\mathcal{E}^O$ **do**
4   Identify the set $\mathcal{B}_1^O \subseteq \mathcal{B}^O$ of those having a box covered by no other LR B-splines in $\mathcal{B}^O$;
5 **if** $\mathcal{B}^O \backslash \mathcal{B}_1^O = \emptyset$ **then**
6   linear independence, break.
7 **else**
8   **if** $\mathcal{B}_1^O = \emptyset$ **then**
9   break, but might have linear dependence
10     $\mathcal{B}^O \leftarrow \mathcal{B}^O \backslash \mathcal{B}_1^O$;
11   Go to 2;

Peeling Algorithm++

1 Create the set $\mathcal{B}^O$ of overloaded LR B-splines;
2 Let $\mathcal{E}^O$ be the boxes of the LR B-splines in $\mathcal{B}^O$;
3 Let $\mathcal{V}^O$ be the T-vertices of the LR B-splines in $\mathcal{B}^O$;
4 **for** *every box in* $\mathcal{E}^O$ **do**
5   Identify the set $\mathcal{B}_1^O \subseteq \mathcal{B}^O$ of those having a box covered by no other LR B-splines in $\mathcal{B}^O$;
6 **for** *every T-vertex* $\boldsymbol{v}$ *in* $\mathcal{V}^O$ **do**
7   If $\boldsymbol{v}$ is only in one $B \in \mathcal{B}^O$, add B to $\mathcal{B}_1^O$;
8 **if** $|\mathcal{B}^O \backslash \mathcal{B}_1^O| < 8$ **then**
9   linear independence, break.
10 **else**
11   **if** $\mathcal{B}_1^O = \emptyset$ **then**
12   break, but might have linear dependence
13     $\mathcal{B}^O \leftarrow \mathcal{B}^O \backslash \mathcal{B}_1^O$;
14   Go to 2;

Seek and Destroy Linear Dependence: Peeling Algorithm++

Peeling Algorithm

1 Create the set $\mathcal{B}^O$ of overloaded LR B-splines;
2 Let $\mathcal{E}^O$ be the boxes of the LR B-splines in $\mathcal{B}^O$;
3 **for** *every box in* $\mathcal{E}^O$ **do**
4   Identify the set $\mathcal{B}_1^O \subseteq \mathcal{B}^O$ of those having a box covered by no other LR B-splines in $\mathcal{B}^O$;
5 **if** $\mathcal{B}^O \backslash \mathcal{B}_1^O = \emptyset$ **then**
6   linear independence, break.
7 **else**
8   **if** $\mathcal{B}_1^O = \emptyset$ **then**
9   break, but might have linear dependence
10     $\mathcal{B}^O \leftarrow \mathcal{B}^O \backslash \mathcal{B}_1^O$;
11   Go to 2;

Peeling Algorithm++

1 Create the set $\mathcal{B}^O$ of overloaded LR B-splines;
2 Let $\mathcal{E}^O$ be the boxes of the LR B-splines in $\mathcal{B}^O$;
3 Let $\mathcal{V}^O$ be the T-vertices of the LR B-splines in $\mathcal{B}^O$;
4 **for** *every box in* $\mathcal{E}^O$ **do**
5   Identify the set $\mathcal{B}_1^O \subseteq \mathcal{B}^O$ of those having a box covered by no other LR B-splines in $\mathcal{B}^O$;
6 **for** *every T-vertex* $\boldsymbol{v}$ *in* $\mathcal{V}^O$ **do**
7   If $\boldsymbol{v}$ is only in one $B \in \mathcal{B}^O$, add B to $\mathcal{B}_1^O$;
8 **if** $|\mathcal{B}^O \backslash \mathcal{B}_1^O| < 8$ **then**
9   linear independence, break.
10 **else**
11   **if** $\mathcal{B}_1^O = \emptyset$ **then**
12   break, but might have linear dependence
13     $\mathcal{B}^O \leftarrow \mathcal{B}^O \backslash \mathcal{B}_1^O$;
14   Go to 2;

Conjecture: Peeling Algorithm++ sorts out all cases, if $\mathcal{B}_1^O = \emptyset$ then there is a linear dependece relation.

Local linear independence and N_2S property

On the other hand, local linear independence has been characterized by Bressan and Jüttler.

Local linear independence and N_2S property

On the other hand, local linear independence has been characterized by Bressan and Jüttler.

Local linear independence: $\nexists$ open set $A \subseteq \Omega : \sum_{B \in \mathcal{B}} \alpha_A B|_A = 0$, with α_A not all zero,

Local linear independence and $\mathrm{N_2S}$ property

On the other hand, local linear independence has been characterized by Bressan and Jüttler.

Local linear independence: $\nexists$ open set $A \subseteq \Omega : \sum_{B \in \mathcal{B}} \alpha_A B|_A = 0$, with α_A not all zero,

$$\Updownarrow$$

No overloading: $\forall$ cell in the mesh β, $\#\{B \in \mathcal{B} : \beta \in \operatorname{supp} B\} = (p_1 + 1)(p_2 + 1)$,

Local linear independence and N_2S property

On the other hand, local linear independence has been characterized by Bressan and Jüttler.

Local linear independence: $\nexists$ open set $A \subseteq \Omega$: $\sum_{B \in \mathcal{B}} \alpha_A B|_A = 0$, with α_A not all zero,

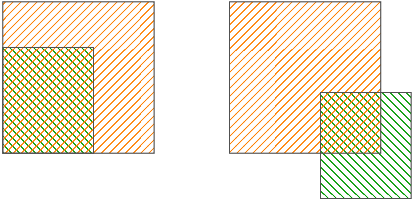
$$\Updownarrow$$

No overloading: $\forall$ cell in the mesh β, $\#\{B \in \mathcal{B} : \beta \in \operatorname{supp} B\} = (p_1 + 1)(p_2 + 1)$,

$$\Updownarrow$$

Non-Nested-Support (N_2S) property of the mesh:

☹ ☺



$\nexists$ B^1, B^2 such that $\operatorname{supp} B^2 \subseteq \operatorname{supp} B^1$.

Local linear independence and N_2S property

On the other hand, local linear independence has been characterized by Bressan and Jüttler.

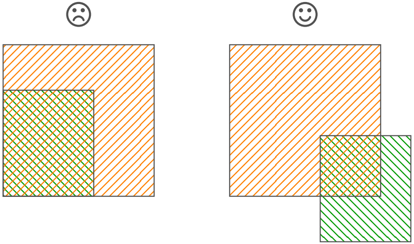
Local linear independence: $\nexists$ open set $A \subseteq \Omega : \sum_{B \in \mathcal{B}} \alpha_A B|_A = 0$, with α_A not all zero,

$$\Updownarrow$$

No overloading: $\forall$ cell in the mesh β, $\#\{B \in \mathcal{B} : \beta \in \operatorname{supp} B\} = (p_1+1)(p_2+1)$,

$$\Updownarrow$$

Non-Nested-Support (N_2S) property of the mesh:



$\nexists\, B^1, B^2$ such that $\operatorname{supp} B^2 \subseteq \operatorname{supp} B^1$.

How do we build LR meshes with the N_2S property?

Non-Nested Support Structured (N_2S_2) Mesh Refinement

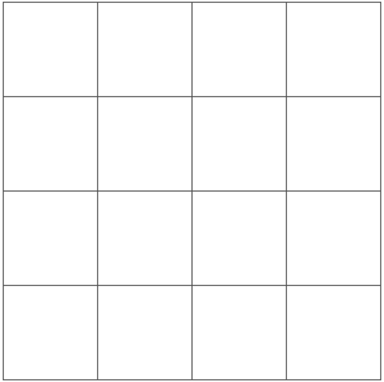
1. select the LR B-spline contributing more to the approximation error (in some sense),

Non-Nested Support Structured (N_2S_2) Mesh Refinement

1. select the LR B-spline contributing more to the approximation error (in some sense),
2. split in 4 all the boxes in their supports (i.e., insert new meshlines),

Non-Nested Support Structured (N_2S_2) Mesh Refinement

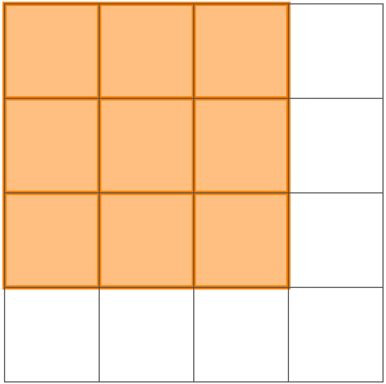
1. select the LR B-spline contributing more to the approximation error (in some sense),
2. split in 4 all the boxes in their supports (i.e., insert new meshlines),



bidegree (2,2)

Non-Nested Support Structured (N_2S_2) Mesh Refinement

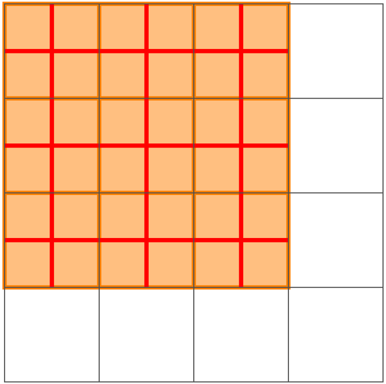
1. select the LR B-spline contributing more to the approximation error (in some sense),
2. split in 4 all the boxes in their supports (i.e., insert new meshlines),



bidegree (2,2)

Non-Nested Support Structured (N_2S_2) Mesh Refinement

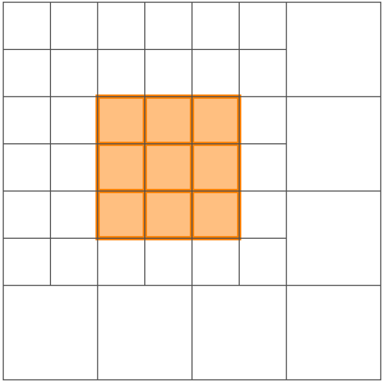
1. select the LR B-spline contributing more to the approximation error (in some sense),
2. split in 4 all the boxes in their supports (i.e., insert new meshlines),



bidegree (2,2)

Non-Nested Support Structured (N_2S_2) Mesh Refinement

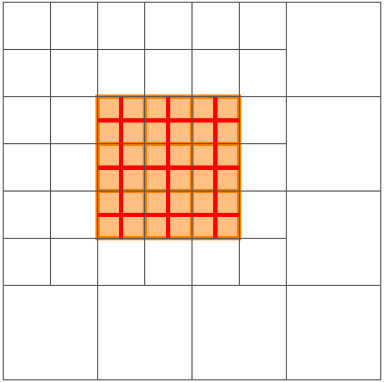
1. select the LR B-spline contributing more to the approximation error (in some sense),
2. split in 4 all the boxes in their supports (i.e., insert new meshlines),



bidegree (2,2)

Non-Nested Support Structured (N_2S_2) Mesh Refinement

1. select the LR B-spline contributing more to the approximation error (in some sense),
2. split in 4 all the boxes in their supports (i.e., insert new meshlines),

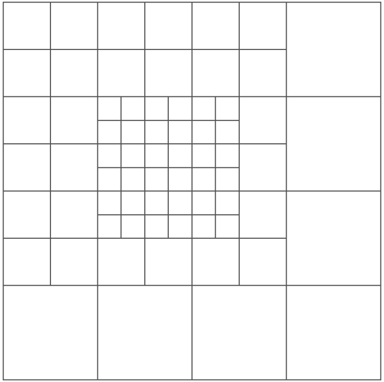


bidegree (2,2)

Non-Nested Support Structured (N_2S_2) Mesh Refinement

1. select the LR B-spline contributing more to the approximation error (in some sense),

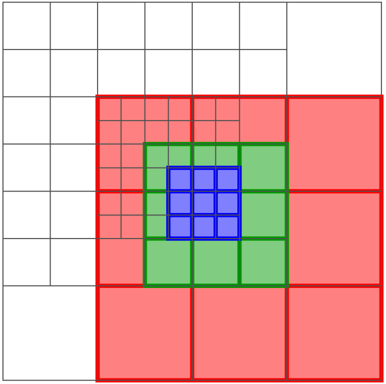
2. split in 4 all the boxes in their supports (i.e., insert new meshlines),



final mesh

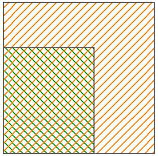
Non-Nested Support Structured (N_2S_2) Mesh Refinement

1. select the LR B-spline contributing more to the approximation error (in some sense),
2. split in 4 all the boxes in their supports (i.e., insert new meshlines),



final mesh (no N_2S property)

recall, NOT OK:



Non-Nested Support Structured (N_2S_2) Mesh Refinement

1. select the LR B-spline contributing more to the approximation error (in some sense),
2. split in 4 all the boxes in their supports (i.e., insert new meshlines),

Non-Nested Support Structured (N_2S_2) Mesh Refinement

1. select the LR B-spline contributing more to the approximation error (in some sense),
2. split in 4 all the boxes in their supports (i.e., insert new meshlines),

B-splines defined on a plain tensor mesh are locally linearly independent. The meshes generated with 1.–2. are locally tensor meshes far from the boundary of the region where the refinement is applied.

Non-Nested Support Structured (N_2S_2) Mesh Refinement

1. select the LR B-spline contributing more to the approximation error (in some sense),
2. split in 4 all the boxes in their supports (i.e., insert new meshlines),

B-splines defined on a plain tensor mesh are locally linearly independent. The meshes generated with 1.–2. are locally tensor meshes far from the boundary of the region where the refinement is applied.

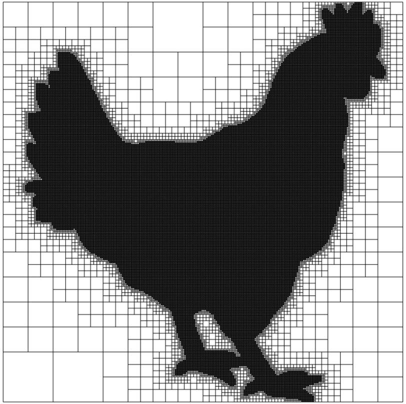
⇒ The LR B-splines defined in these zones of the mesh behave like the standard B-splines, and therefore are locally linearly independent.

Non-Nested Support Structured (N_2S_2) Mesh Refinement

1. select the LR B-spline contributing more to the approximation error (in some sense),
2. split in 4 all the boxes in their supports (i.e., insert new meshlines),

- B-splines defined on a plain tensor mesh are locally linearly independent. The meshes generated with 1.–2. are locally tensor meshes far from the boundary of the region where the refinement is applied.

$\Rightarrow$ The LR B-splines defined in these zones of the mesh behave like the standard B-splines, and therefore are locally linearly independent.

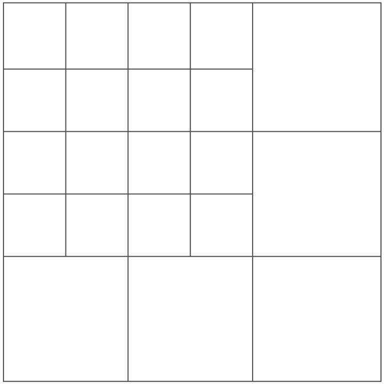


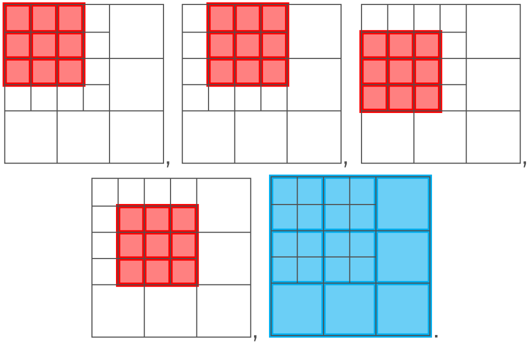
Non-Nested Support Structured (N_2S_2) Mesh Refinement

1. select the LR B-spline contributing more to the approximation error (in some sense),
2. split in 4 all the boxes in their supports (i.e., insert new meshlines),
3. modify the boundary of the region where the refinement is applied

Non-Nested Support Structured (N_2S_2) Mesh Refinement

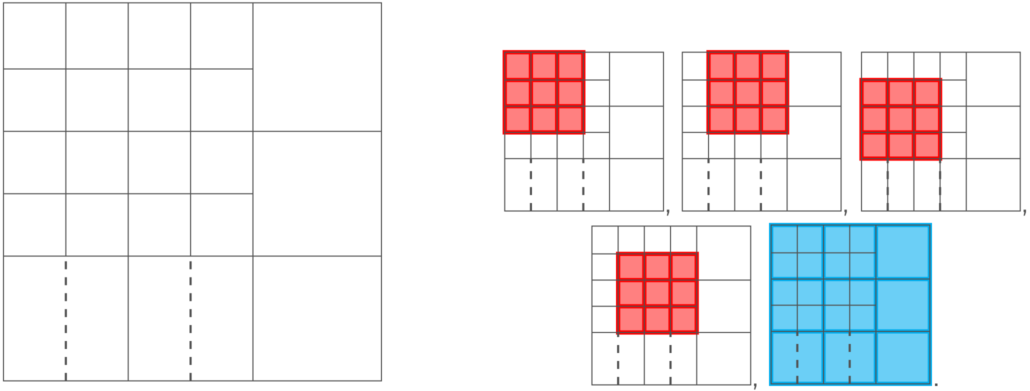
1. select the LR B-spline contributing more to the approximation error (in some sense),
2. split in 4 all the boxes in their supports (i.e., insert new meshlines),
3. modify the boundary of the region where the refinement is applied





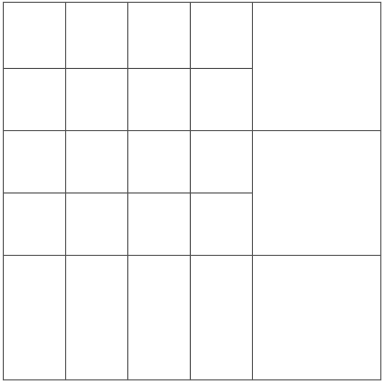
Non-Nested Support Structured (N_2S_2) Mesh Refinement

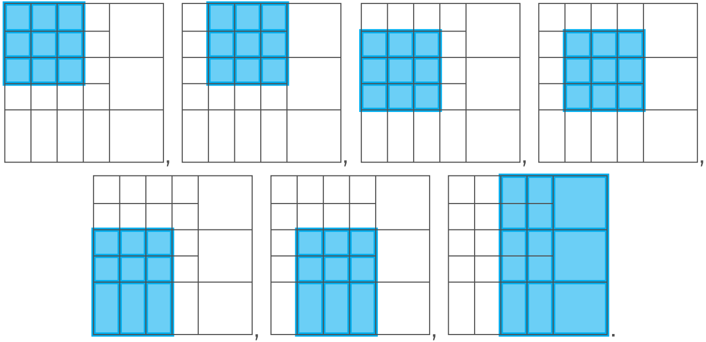
1. select the LR B-spline contributing more to the approximation error (in some sense),
2. split in 4 all the boxes in their supports (i.e., insert new meshlines),
3. modify the boundary of the region where the refinement is applied



Non-Nested Support Structured (N_2S_2) Mesh Refinement

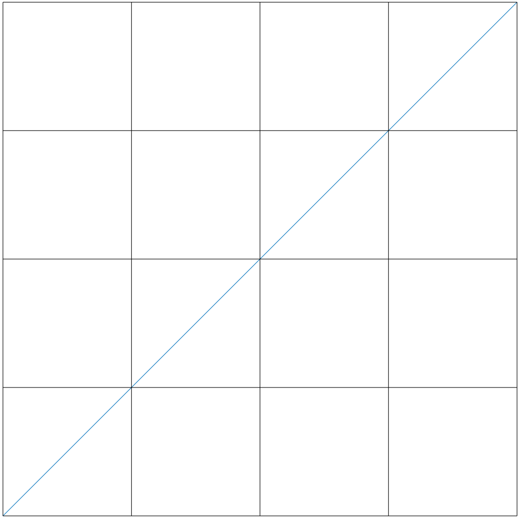
1. select the LR B-spline contributing more to the approximation error (in some sense),
2. split in 4 all the boxes in their supports (i.e., insert new meshlines),
3. modify the boundary of the region where the refinement is applied
4. apply the LR B-splines generation algorithm to refine the space.





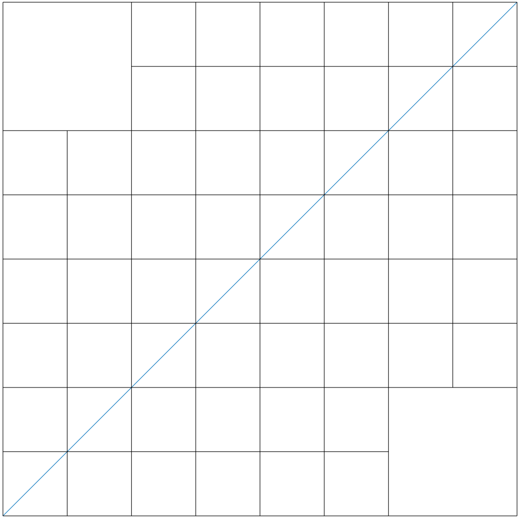
Non-Nested Support Structured (N_2S_2) Mesh Refinement

Example: degree $(2, 2)$.



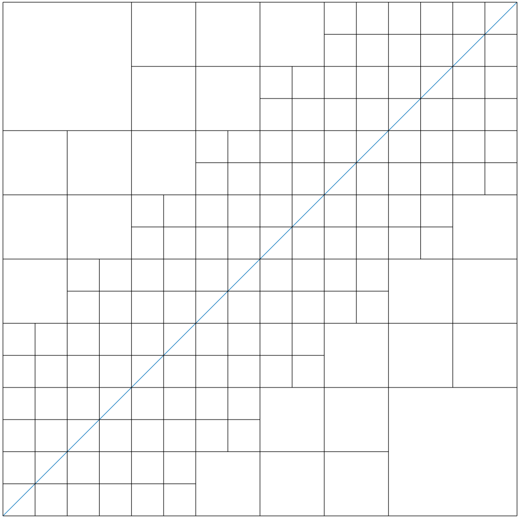
Non-Nested Support Structured (N_2S_2) Mesh Refinement

Example: degree $(2, 2)$.



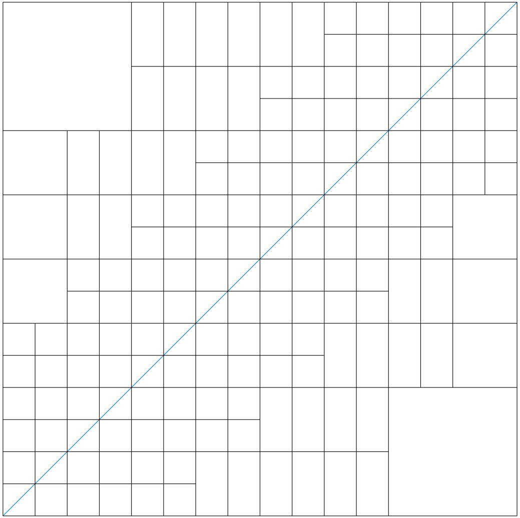
Non-Nested Support Structured (N_2S_2) Mesh Refinement

Example: degree $(2, 2)$.



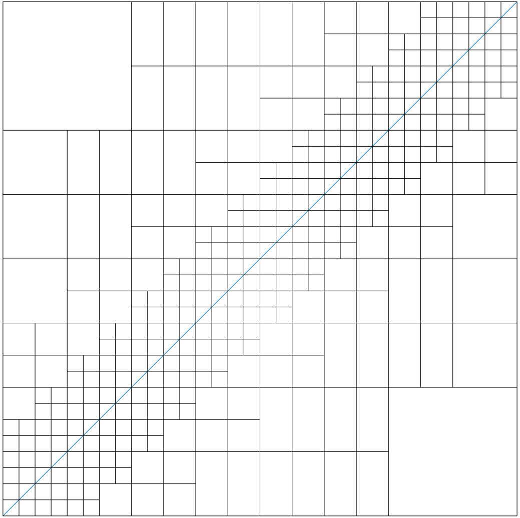
Non-Nested Support Structured (N_2S_2) Mesh Refinement

Example: degree $(2, 2)$.



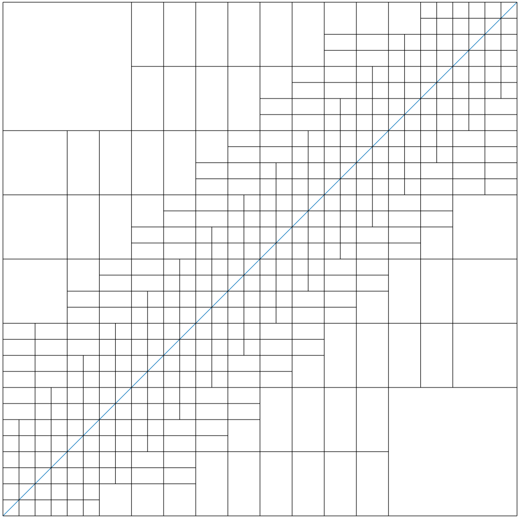
Non-Nested Support Structured (N_2S_2) Mesh Refinement

Example: degree $(2, 2)$.



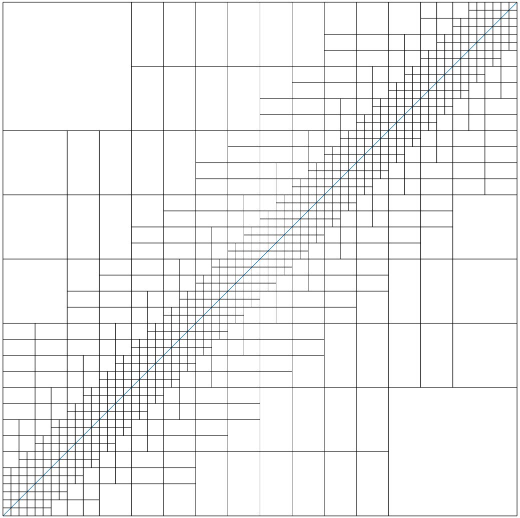
Non-Nested Support Structured (N_2S_2) Mesh Refinement

Example: degree $(2, 2)$.



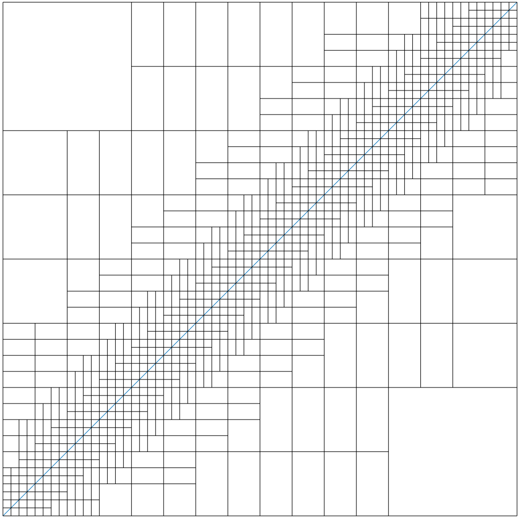
Non-Nested Support Structured (N_2S_2) Mesh Refinement

Example: degree $(2,2)$.



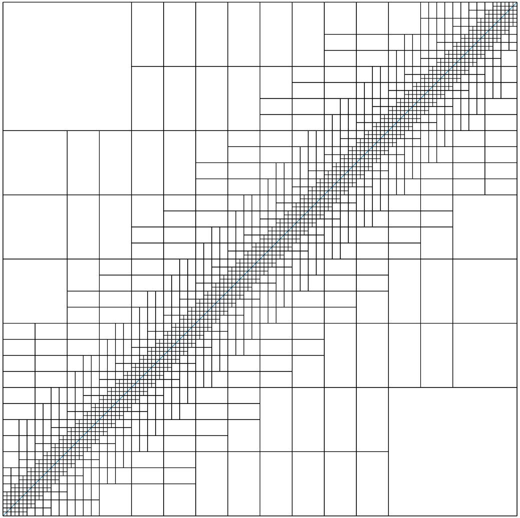
Non-Nested Support Structured (N_2S_2) Mesh Refinement

Example: degree $(2, 2)$.



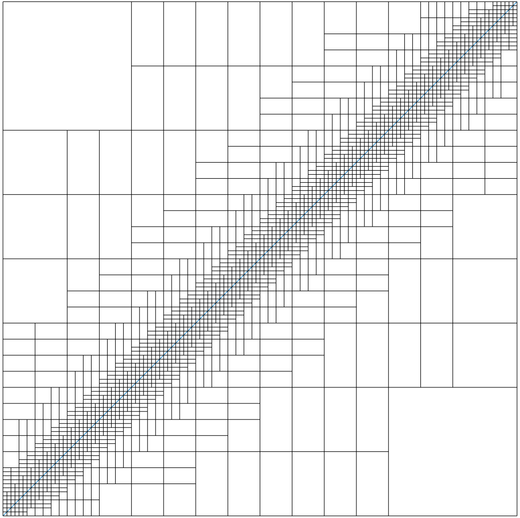
Non-Nested Support Structured (N_2S_2) Mesh Refinement

Example: degree $(2, 2)$.

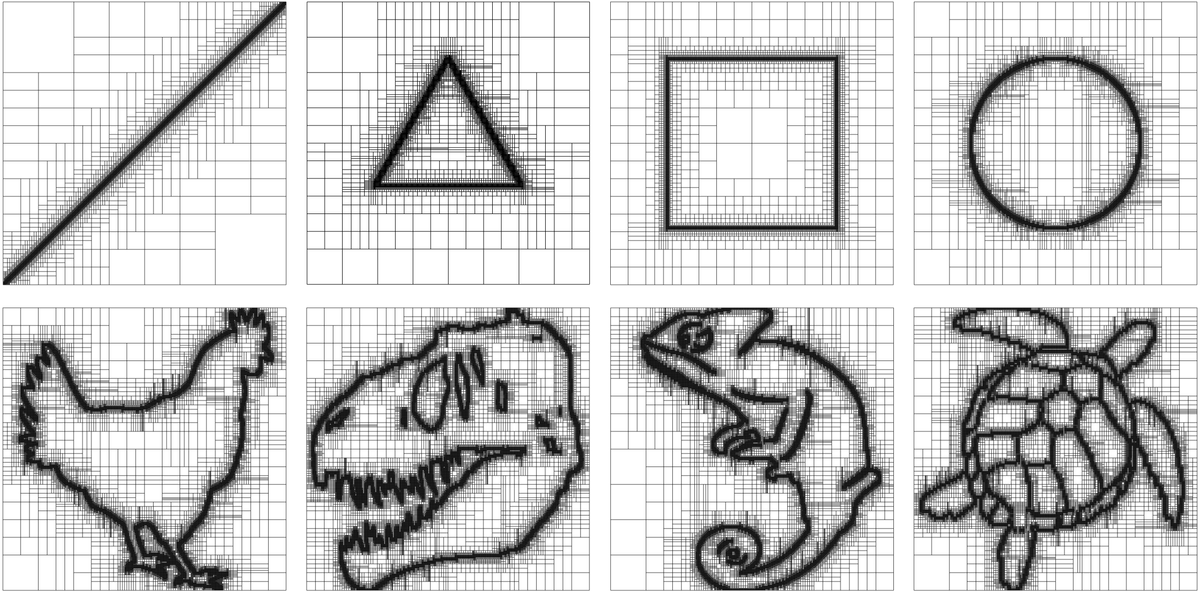


Non-Nested Support Structured (N_2S_2) Mesh Refinement

Example: degree $(2, 2)$.



Non-Nested Support Structured (N_2S_2) Mesh Refinement



Hierarchical LR-Mesh Costruction

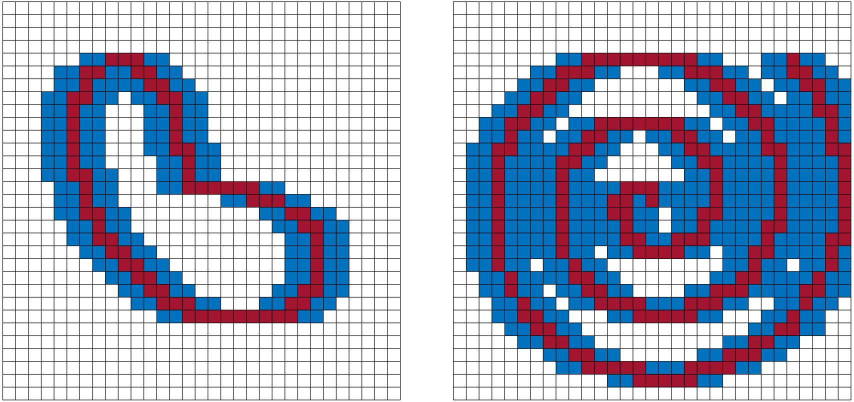
Shadow Map:

Hierarchical LR-Mesh Costruction

Shadow Map: A bunch of boxes in a tensor mesh $\mathcal{N}$, the horizontal shadow of A, $\mathcal{S}A$ is the superset obtained moving the boundary outward of p_1 more boxes in the horizontal direction.

Hierarchical LR-Mesh Costruction

Shadow Map: A bunch of boxes in a tensor mesh $\mathcal{N}$, the horizontal shadow of A, $\mathcal{S}A$ is the superset obtained moving the boundary outward of p_1 more boxes in the horizontal direction.



Hierarchical LR-Mesh Costruction

Shadow Map: A bunch of boxes in a tensor mesh $\mathcal{N}$, the horizontal shadow of A, $\mathcal{S}A$ is the superset obtained moving the boundary outward of p_1 more boxes in the horizontal direction.

- $\mathcal{N}^{\ell+1}$ tensor mesh obtained bisecting the boxes of $\mathcal{N}^{\ell}$ in one direction alternately on ℓ,
- $\Omega = \Omega^0 \supseteq \ldots \supseteq \Omega^m$ sequence of nested domains with Ω^ℓ union of boxes in $\mathcal{N}^\ell$,

Hierarchical LR-Mesh Costruction

Shadow Map: A bunch of boxes in a tensor mesh $\mathcal{N}$, the horizontal shadow of A, $\mathcal{S}A$ is the superset obtained moving the boundary outward of p_1 more boxes in the horizontal direction.

- $\mathcal{N}^{\ell+1}$ tensor mesh obtained bisecting the boxes of $\mathcal{N}^{\ell}$ in one direction alternately on ℓ,
- $\Omega = \Omega^0 \supseteq \ldots \supseteq \Omega^m$ sequence of nested domains with Ω^ℓ union of boxes in $\mathcal{N}^\ell$,

Hierarchical LR-mesh: $\bigcup_{\ell=0}^{m} \{\beta \text{ boxes of } \mathcal{N}^\ell \text{ inside } \Omega^\ell \backslash \Omega^{\ell+1}\}$

Hierarchical LR-Mesh Costruction

Shadow Map: A bunch of boxes in a tensor mesh $\mathcal{N}$, the horizontal shadow of A, $\mathcal{S}A$ is the superset obtained moving the boundary outward of p_1 more boxes in the horizontal direction.

- $\mathcal{N}^{\ell+1}$ tensor mesh obtained bisecting the boxes of $\mathcal{N}^{\ell}$ in one direction alternately on ℓ,
- $\Omega = \Omega^0 \supseteq \ldots \supseteq \Omega^m$ sequence of nested domains with Ω^ℓ union of boxes in $\mathcal{N}^\ell$,

Hierarchical LR-mesh: $\bigcup_{\ell=0}^{m} \{\beta \text{ boxes of } \mathcal{N}^\ell \text{ inside } \Omega^\ell \backslash \Omega^{\ell+1}\}$

Theorem: If $\Omega^\ell \supseteq \mathcal{S}\Omega^{\ell+1}$ for every ℓ then the Hierarchical LR-mesh has the N_2S property.

Hierarchical LR-Mesh Costruction

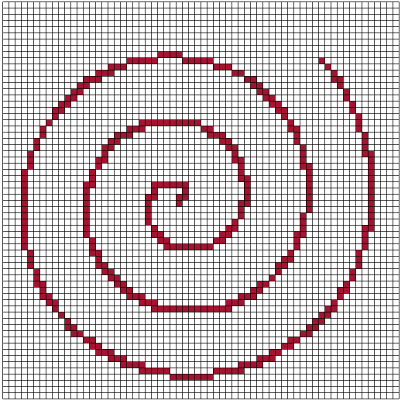
Shadow Map: A bunch of boxes in a tensor mesh $\mathcal{N}$, the horizontal shadow of A, $\mathcal{S}A$ is the superset obtained moving the boundary outward of p_1 more boxes in the horizontal direction.

- $\mathcal{N}^{\ell+1}$ tensor mesh obtained bisecting the boxes of $\mathcal{N}^{\ell}$ in one direction alternately on ℓ,
- $\Omega = \Omega^0 \supseteq \ldots \supseteq \Omega^m$ sequence of nested domains with Ω^{ℓ} union of boxes in $\mathcal{N}^{\ell}$,

Hierarchical LR-mesh: $\bigcup_{\ell=0}^{m} \{\beta \text{ boxes of } \mathcal{N}^{\ell} \text{ inside } \Omega^{\ell} \backslash \Omega^{\ell+1}\}$

Theorem: If $\Omega^{\ell} \supseteq \mathcal{S}\Omega^{\ell+1}$ for every ℓ then the Hierarchical LR-mesh has the N_2S property.

1. select the refinement region and max resolution m,



Hierarchical LR-Mesh Costruction

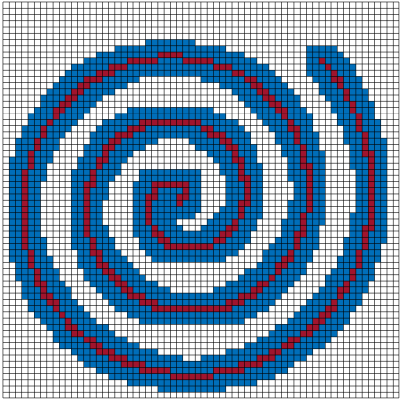
Shadow Map: A bunch of boxes in a tensor mesh $\mathcal{N}$, the horizontal shadow of A, $\mathcal{S}A$ is the superset obtained moving the boundary outward of p_1 more boxes in the horizontal direction.

- $\mathcal{N}^{\ell+1}$ tensor mesh obtained bisecting the boxes of $\mathcal{N}^{\ell}$ in one direction alternately on ℓ,
- $\Omega = \Omega^0 \supseteq \ldots \supseteq \Omega^m$ sequence of nested domains with Ω^ℓ union of boxes in $\mathcal{N}^\ell$,

Hierarchical LR-mesh: $\bigcup_{\ell=0}^{m} \{\beta \text{ boxes of } \mathcal{N}^\ell \text{ inside } \Omega^\ell \backslash \Omega^{\ell+1}\}$

Theorem: If $\Omega^\ell \supseteq \mathcal{S}\Omega^{\ell+1}$ for every ℓ then the Hierarchical LR-mesh has the N_2S property.

1. select the refinement region and max resolution m,
2. select the boxes in the supports of the tensor product B-splines on $\mathcal{N}^m$ touching the region,

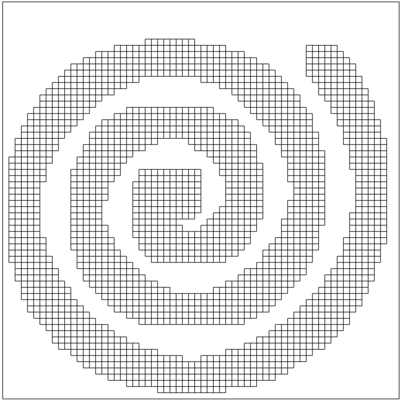


Hierarchical LR-Mesh Costruction

Shadow Map: A bunch of boxes in a tensor mesh $\mathcal{N}$, the horizontal shadow of A, $\mathcal{S}A$ is the superset obtained moving the boundary outward of p_1 more boxes in the horizontal direction.

- $\mathcal{N}^{\ell+1}$ tensor mesh obtained bisecting the boxes of $\mathcal{N}^{\ell}$ in one direction alternately on ℓ,
- $\Omega = \Omega^0 \supseteq \ldots \supseteq \Omega^m$ sequence of nested domains with Ω^ℓ union of boxes in $\mathcal{N}^\ell$,

Hierarchical LR-mesh: $\displaystyle\bigcup_{\ell=0}^{m} \{\beta \text{ boxes of } \mathcal{N}^\ell \text{ inside } \Omega^\ell \setminus \Omega^{\ell+1}\}$



Theorem: If $\Omega^\ell \supseteq \mathcal{S}\Omega^{\ell+1}$ for every ℓ then the Hierarchical LR-mesh has the N_2S property.

1. select the refinement region and max resolution m,
2. select the boxes in the supports of the tensor product B-splines on $\mathcal{N}^m$ touching the region,

Hierarchical LR-Mesh Costruction

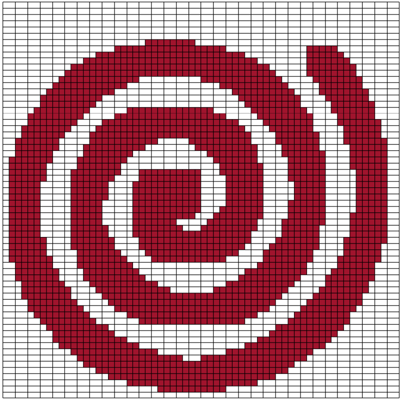
Shadow Map: A bunch of boxes in a tensor mesh $\mathcal{N}$, the horizontal shadow of A, $\mathcal{S}A$ is the superset obtained moving the boundary outward of p_1 more boxes in the horizontal direction.

- $\mathcal{N}^{\ell+1}$ tensor mesh obtained bisecting the boxes of $\mathcal{N}^{\ell}$ in one direction alternately on ℓ,
- $\Omega = \Omega^0 \supseteq \ldots \supseteq \Omega^m$ sequence of nested domains with Ω^ℓ union of boxes in $\mathcal{N}^\ell$,

Hierarchical LR-mesh: $\bigcup_{\ell=0}^{m} \{\beta \text{ boxes of } \mathcal{N}^\ell \text{ inside } \Omega^\ell \setminus \Omega^{\ell+1}\}$

Theorem: If $\Omega^\ell \supseteq \mathcal{S}\Omega^{\ell+1}$ for every ℓ then the Hierarchical LR-mesh has the N_2S property.

1. select the refinement region and max resolution m,
2. select the boxes in the supports of the tensor product B-splines on $\mathcal{N}^m$ touching the region,
3. embed this new region inside $\mathcal{N}^{m-1}$ which is coarser in one direction,

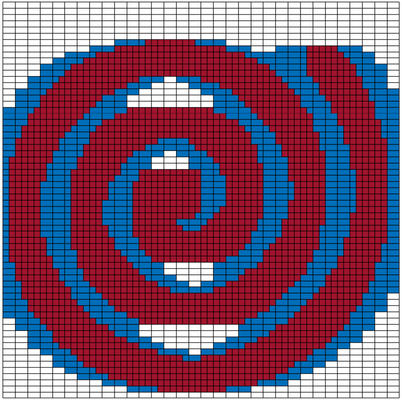


Hierarchical LR-Mesh Costruction

Shadow Map: A bunch of boxes in a tensor mesh $\mathcal{N}$, the horizontal shadow of A, $\mathcal{S}A$ is the superset obtained moving the boundary outward of p_1 more boxes in the horizontal direction.

- $\mathcal{N}^{\ell+1}$ tensor mesh obtained bisecting the boxes of $\mathcal{N}^{\ell}$ in one direction alternately on ℓ,
- $\Omega = \Omega^0 \supseteq \ldots \supseteq \Omega^m$ sequence of nested domains with Ω^ℓ union of boxes in $\mathcal{N}^\ell$,

Hierarchical LR-mesh: $\displaystyle\bigcup_{\ell=0}^{m}\{\beta \text{ boxes of } \mathcal{N}^\ell \text{ inside } \Omega^\ell \backslash \Omega^{\ell+1}\}$



Theorem: If $\Omega^\ell \supseteq \mathcal{S}\Omega^{\ell+1}$ for every ℓ then the Hierarchical LR-mesh has the N_2S property.

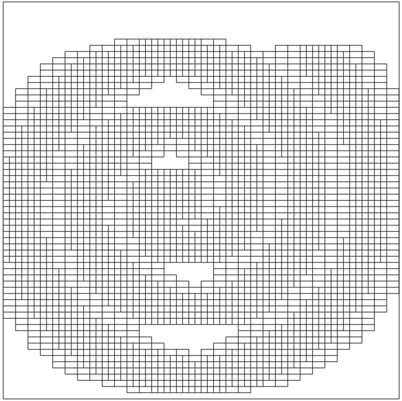
1. select the refinement region and max resolution m,
2. select the boxes in the supports of the tensor product B-splines on $\mathcal{N}^m$ touching the region,
3. embed this new region inside $\mathcal{N}^{m-1}$ which is coarser in one direction,
4. compute the shadow along the coarser direction in $\mathcal{N}^{m-1}$,

Hierarchical LR-Mesh Costruction

Shadow Map: A bunch of boxes in a tensor mesh $\mathcal{N}$, the horizontal shadow of A, $\mathcal{S}A$ is the superset obtained moving the boundary outward of p_1 more boxes in the horizontal direction.

- $\mathcal{N}^{\ell+1}$ tensor mesh obtained bisecting the boxes of $\mathcal{N}^{\ell}$ in one direction alternately on ℓ,
- $\Omega = \Omega^0 \supseteq \ldots \supseteq \Omega^m$ sequence of nested domains with Ω^ℓ union of boxes in $\mathcal{N}^\ell$,

Hierarchical LR-mesh: $\displaystyle\bigcup_{\ell=0}^{m}\{\beta \text{ boxes of } \mathcal{N}^\ell \text{ inside } \Omega^\ell \backslash \Omega^{\ell+1}\}$



Theorem: If $\Omega^\ell \supseteq \mathcal{S}\Omega^{\ell+1}$ for every ℓ then the Hierarchical LR-mesh has the N_2S property.

1. select the refinement region and max resolution m,
2. select the boxes in the supports of the tensor product B-splines on $\mathcal{N}^m$ touching the region,
3. embed this new region inside $\mathcal{N}^{m-1}$ which is coarser in one direction,
4. compute the shadow along the coarser direction in $\mathcal{N}^{m-1}$,
5. select the boxes enclosed in this new boundary,

Hierarchical LR-Mesh Costruction

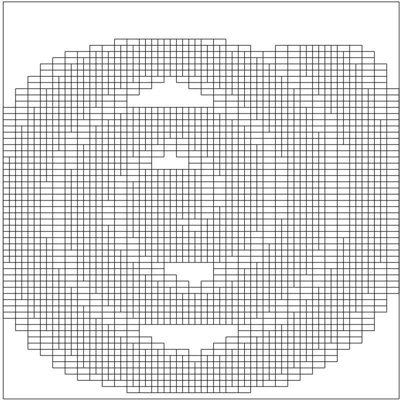
Shadow Map: A bunch of boxes in a tensor mesh $\mathcal{N}$, the horizontal shadow of A, $\mathcal{S}A$ is the superset obtained moving the boundary outward of p_1 more boxes in the horizontal direction.

- $\mathcal{N}^{\ell+1}$ tensor mesh obtained bisecting the boxes of $\mathcal{N}^{\ell}$ in one direction alternately on ℓ,
- $\Omega = \Omega^0 \supseteq \ldots \supseteq \Omega^m$ sequence of nested domains with Ω^ℓ union of boxes in $\mathcal{N}^\ell$,

Hierarchical LR-mesh: $\displaystyle\bigcup_{\ell=0}^{m} \{\beta \text{ boxes of } \mathcal{N}^\ell \text{ inside } \Omega^\ell \backslash \Omega^{\ell+1}\}$

Theorem: If $\Omega^\ell \supseteq \mathcal{S}\Omega^{\ell+1}$ for every ℓ then the Hierarchical LR-mesh has the N_2S property.

1. select the refinement region and max resolution m,
2. select the boxes in the supports of the tensor product B-splines on $\mathcal{N}^m$ touching the region,
3. embed this new region inside $\mathcal{N}^{m-1}$ which is coarser in one direction,
4. compute the shadow along the coarser direction in $\mathcal{N}^{m-1}$,
5. select the boxes enclosed in this new boundary,
6. repeat 3.–5. for coarser meshes switching the shadow direction until the mesh is complete.

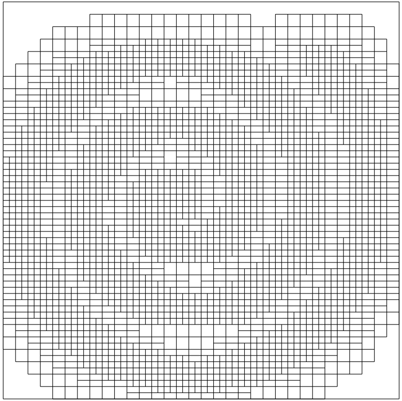


Hierarchical LR-Mesh Costruction

Shadow Map: A bunch of boxes in a tensor mesh $\mathcal{N}$, the horizontal shadow of A, $\mathcal{S}A$ is the superset obtained moving the boundary outward of p_1 more boxes in the horizontal direction.

- $\mathcal{N}^{\ell+1}$ tensor mesh obtained bisecting the boxes of $\mathcal{N}^\ell$ in one direction alternately on ℓ,
- $\Omega = \Omega^0 \supseteq \ldots \supseteq \Omega^m$ sequence of nested domains with Ω^ℓ union of boxes in $\mathcal{N}^\ell$,

Hierarchical LR-mesh: $\bigcup_{\ell=0}^{m} \{\beta \text{ boxes of } \mathcal{N}^\ell \text{ inside } \Omega^\ell \backslash \Omega^{\ell+1}\}$



Theorem: If $\Omega^\ell \supseteq \mathcal{S}\Omega^{\ell+1}$ for every ℓ then the Hierarchical LR-mesh has the N_2S property.

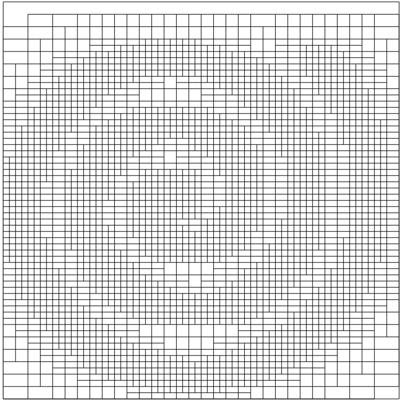
1. select the refinement region and max resolution m,
2. select the boxes in the supports of the tensor product B-splines on $\mathcal{N}^m$ touching the region,
3. embed this new region inside $\mathcal{N}^{m-1}$ which is coarser in one direction,
4. compute the shadow along the coarser direction in $\mathcal{N}^{m-1}$,
5. select the boxes enclosed in this new boundary,
6. repeat 3.–5. for coarser meshes switching the shadow direction until the mesh is complete.

Hierarchical LR-Mesh Costruction

Shadow Map: A bunch of boxes in a tensor mesh $\mathcal{N}$, the horizontal shadow of A, $\mathcal{S}A$ is the superset obtained moving the boundary outward of p_1 more boxes in the horizontal direction.

- $\mathcal{N}^{\ell+1}$ tensor mesh obtained bisecting the boxes of $\mathcal{N}^\ell$ in one direction alternately on ℓ,
- $\Omega = \Omega^0 \supseteq \ldots \supseteq \Omega^m$ sequence of nested domains with Ω^ℓ union of boxes in $\mathcal{N}^\ell$,

Hierarchical LR-mesh: $\bigcup_{\ell=0}^{m} \{\beta \text{ boxes of } \mathcal{N}^\ell \text{ inside } \Omega^\ell \setminus \Omega^{\ell+1}\}$



Theorem: If $\Omega^\ell \supseteq \mathcal{S}\Omega^{\ell+1}$ for every ℓ then the Hierarchical LR-mesh has the N_2S property.

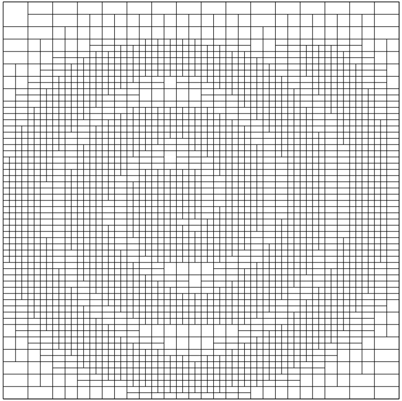
1. select the refinement region and max resolution m,
2. select the boxes in the supports of the tensor product B-splines on $\mathcal{N}^m$ touching the region,
3. embed this new region inside $\mathcal{N}^{m-1}$ which is coarser in one direction,
4. compute the shadow along the coarser direction in $\mathcal{N}^{m-1}$,
5. select the boxes enclosed in this new boundary,
6. repeat 3.–5. for coarser meshes switching the shadow direction until the mesh is complete.

Hierarchical LR-Mesh Costruction

Shadow Map: A bunch of boxes in a tensor mesh $\mathcal{N}$, the horizontal shadow of A, $\mathcal{S}A$ is the superset obtained moving the boundary outward of p_1 more boxes in the horizontal direction.

- $\mathcal{N}^{\ell+1}$ tensor mesh obtained bisecting the boxes of $\mathcal{N}^{\ell}$ in one direction alternately on ℓ,
- $\Omega = \Omega^0 \supseteq \ldots \supseteq \Omega^m$ sequence of nested domains with Ω^ℓ union of boxes in $\mathcal{N}^\ell$,

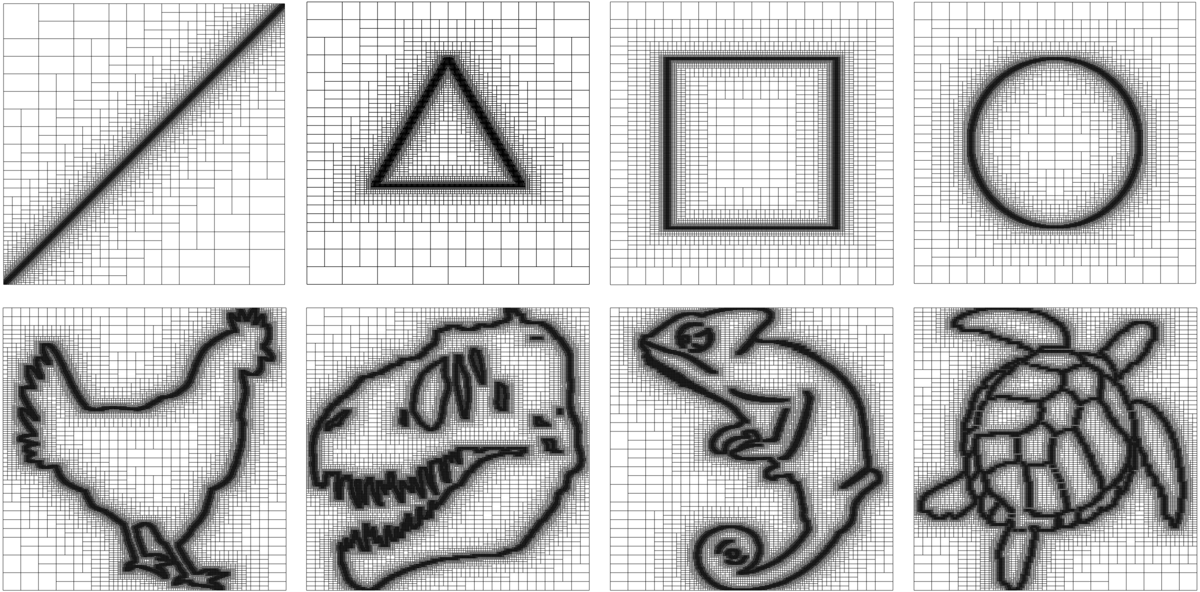
Hierarchical LR-mesh: $\bigcup_{\ell=0}^{m} \{\beta \text{ boxes of } \mathcal{N}^\ell \text{ inside } \Omega^\ell \backslash \Omega^{\ell+1}\}$



Theorem: If $\Omega^\ell \supseteq \mathcal{S}\Omega^{\ell+1}$ for every ℓ then the Hierarchical LR-mesh has the N_2S property.

1. select the refinement region and max resolution m,
2. select the boxes in the supports of the tensor product B-splines on $\mathcal{N}^m$ touching the region,
3. embed this new region inside $\mathcal{N}^{m-1}$ which is coarser in one direction,
4. compute the shadow along the coarser direction in $\mathcal{N}^{m-1}$,
5. select the boxes enclosed in this new boundary,
6. repeat 3.–5. for coarser meshes switching the shadow direction until the mesh is complete.

Hierarchical LR-Mesh Costruction



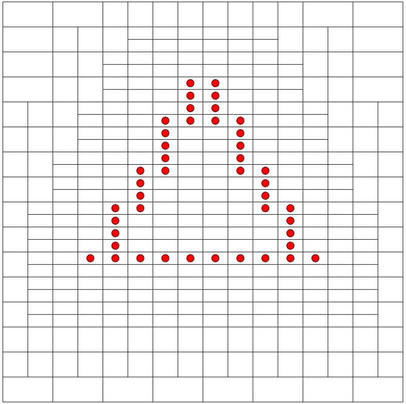
Effective Grading Refinement Strategy

Refinement macro-step:

Effective Grading Refinement Strategy

Refinement macro-step:

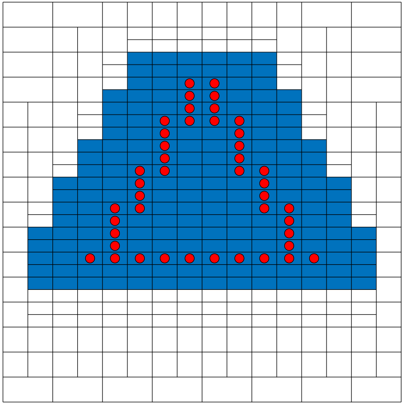
1. Given a set of boxes marked for refinement,



Effective Grading Refinement Strategy

Refinement macro-step:

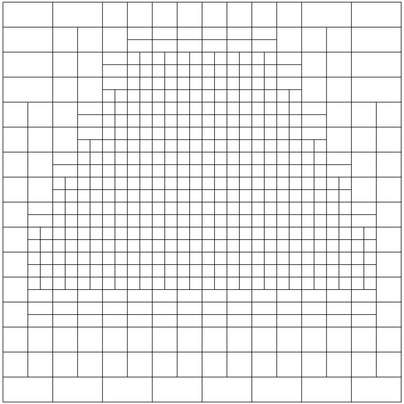
1. Given a set of boxes marked for refinement,
2. Collect all the LR B-splines whose support intersects the marked boxes,



Effective Grading Refinement Strategy

Refinement macro-step:

1. Given a set of boxes marked for refinement,
2. Collect all the LR B-splines whose support intersects the marked boxes,
3. Halve the boxes of largest diameter in their support (in this case all the boxes in the colored region),

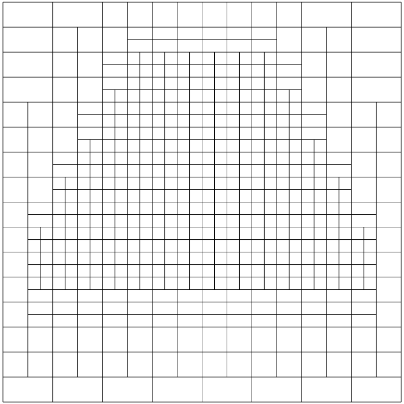


Effective Grading Refinement Strategy

Refinement macro-step:

1. Given a set of boxes marked for refinement,
2. Collect all the LR B-splines whose support intersects the marked boxes,
3. Halve the boxes of largest diameter in their support (in this case all the boxes in the colored region),

N_2S reinstatement macro-step:



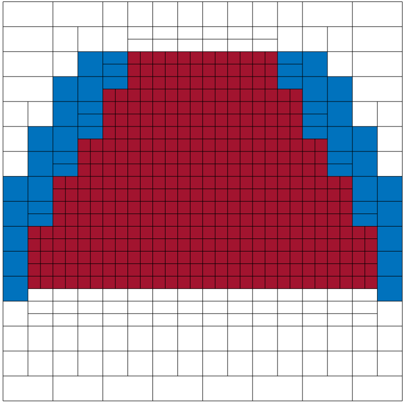
Effective Grading Refinement Strategy

Refinement macro-step:

1. Given a set of boxes marked for refinement,
2. Collect all the LR B-splines whose support intersects the marked boxes,
3. Halve the boxes of largest diameter in their support (in this case all the boxes in the colored region),

N_2S reinstatement macro-step:

4. Consider the smallest boxes on the mesh and compute the shadow of such region (horizontal if square boxes, vertical if rectangular boxes),



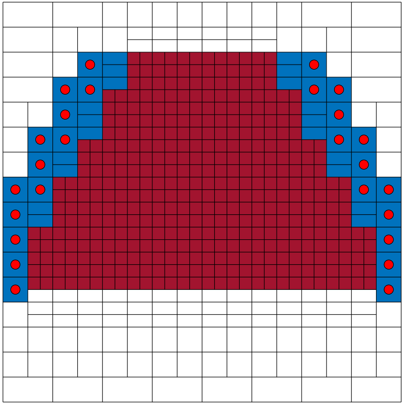
Effective Grading Refinement Strategy

Refinement macro-step:

1. Given a set of boxes marked for refinement,
2. Collect all the LR B-splines whose support intersects the marked boxes,
3. Halve the boxes of largest diameter in their support (in this case all the boxes in the colored region),

N_2S reinstatement macro-step:

4. Consider the smallest boxes on the mesh and compute the shadow of such region (horizontal if square boxes, vertical if rectangular boxes),
5. Mark for refinement those boxes in the shadow that are “too large” compared to the boxes in the red region,



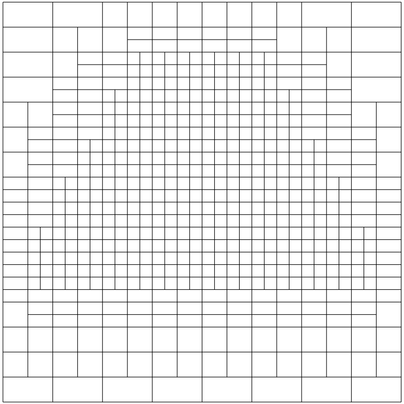
Effective Grading Refinement Strategy

Refinement macro-step:

1. Given a set of boxes marked for refinement,
2. Collect all the LR B-splines whose support intersects the marked boxes,
3. Halve the boxes of largest diameter in their support (in this case all the boxes in the colored region),

N_2S reinstatement macro-step:

4. Consider the smallest boxes on the mesh and compute the shadow of such region (horizontal if square boxes, vertical if rectangular boxes),
5. Mark for refinement those boxes in the shadow that are "too large" compared to the boxes in the red region,
6. Halve such larger boxes,



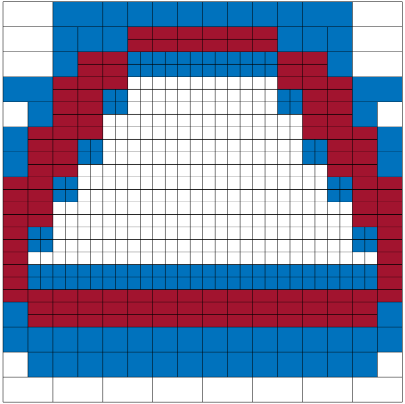
Effective Grading Refinement Strategy

Refinement macro-step:

1. Given a set of boxes marked for refinement,
2. Collect all the LR B-splines whose support intersects the marked boxes,
3. Halve the boxes of largest diameter in their support (in this case all the boxes in the colored region),

N_2S reinstatement macro-step:

4. Consider the smallest boxes on the mesh and compute the shadow of such region (horizontal if square boxes, vertical if rectangular boxes),
5. Mark for refinement those boxes in the shadow that are "too large" compared to the boxes in the red region,
6. Halve such larger boxes,
7. Iterate over all the boxes from the smaller to the larger.



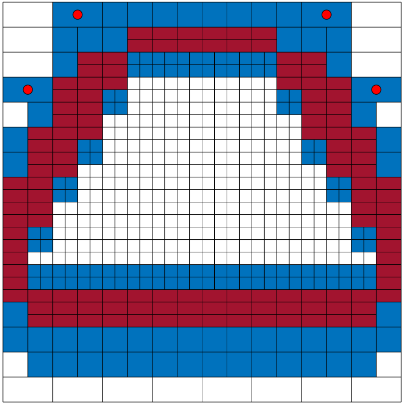
Effective Grading Refinement Strategy

Refinement macro-step:

1. Given a set of boxes marked for refinement,
2. Collect all the LR B-splines whose support intersects the marked boxes,
3. Halve the boxes of largest diameter in their support (in this case all the boxes in the colored region),

N_2S reinstatement macro-step:

4. Consider the smallest boxes on the mesh and compute the shadow of such region (horizontal if square boxes, vertical if rectangular boxes),
5. Mark for refinement those boxes in the shadow that are "too large" compared to the boxes in the red region,
6. Halve such larger boxes,
7. Iterate over all the boxes from the smaller to the larger.



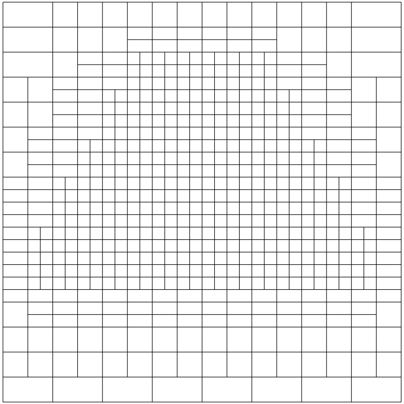
Effective Grading Refinement Strategy

Refinement macro-step:

1. Given a set of boxes marked for refinement,
2. Collect all the LR B-splines whose support intersects the marked boxes,
3. Halve the boxes of largest diameter in their support (in this case all the boxes in the colored region),

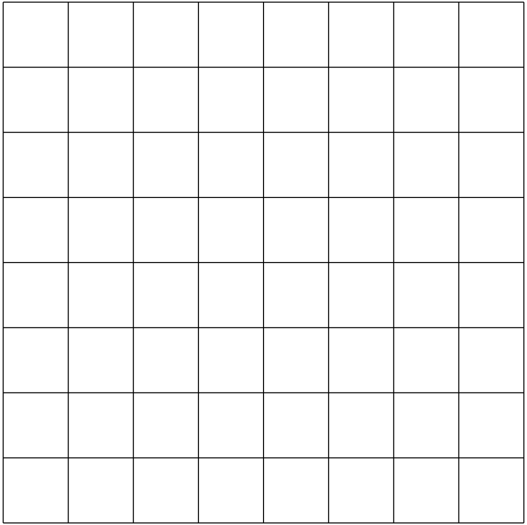
N_2S reinstatement macro-step:

4. Consider the smallest boxes on the mesh and compute the shadow of such region (horizontal if square boxes, vertical if rectangular boxes),
5. Mark for refinement those boxes in the shadow that are "too large" compared to the boxes in the red region,
6. Halve such larger boxes,
7. Iterate over all the boxes from the smaller to the larger.



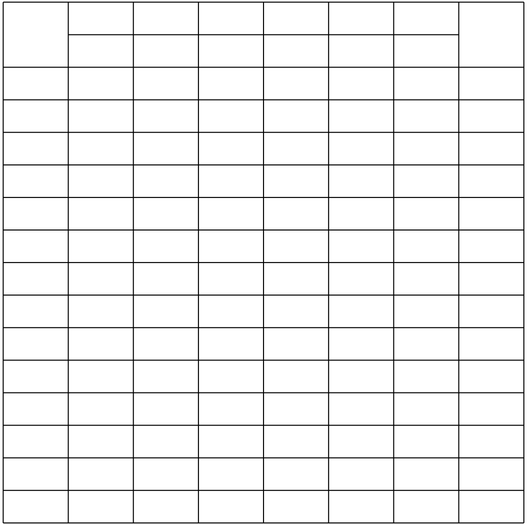
Effective Grading Refinement Strategy

Example: degree $(2, 2)$, Triangle $\rightarrow$ Circle $\rightarrow$ Square



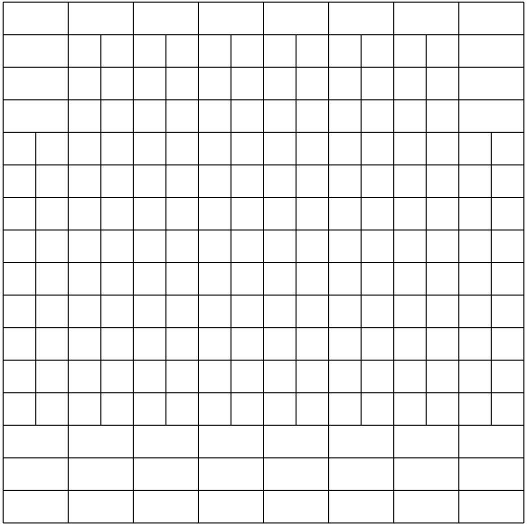
Effective Grading Refinement Strategy

Example: degree $(2,2)$, Triangle $\rightarrow$ Circle $\rightarrow$ Square



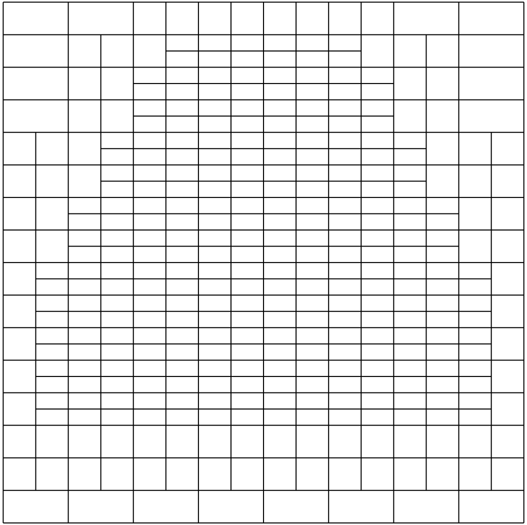
Effective Grading Refinement Strategy

Example: degree $(2,2)$, Triangle $\rightarrow$ Circle $\rightarrow$ Square



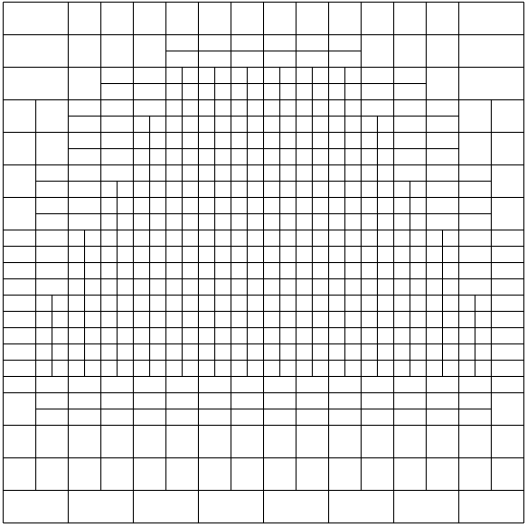
Effective Grading Refinement Strategy

Example: degree $(2,2)$, Triangle $\to$ Circle $\to$ Square



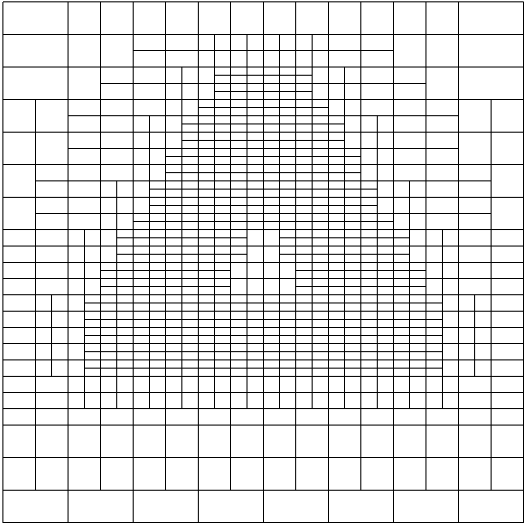
Effective Grading Refinement Strategy

Example: degree $(2,2)$, Triangle $\rightarrow$ Circle $\rightarrow$ Square



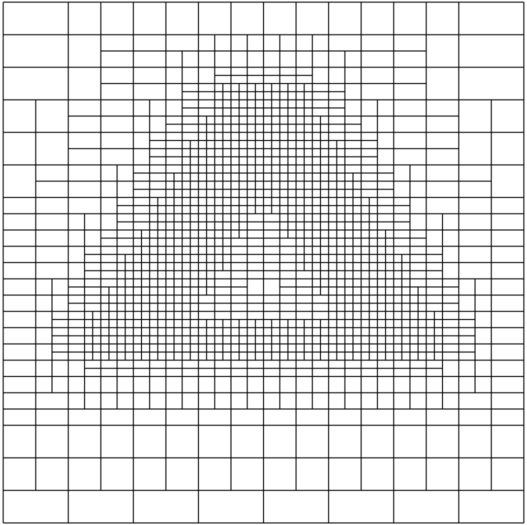
Effective Grading Refinement Strategy

Example: degree $(2,2)$, Triangle $\rightarrow$ Circle $\rightarrow$ Square



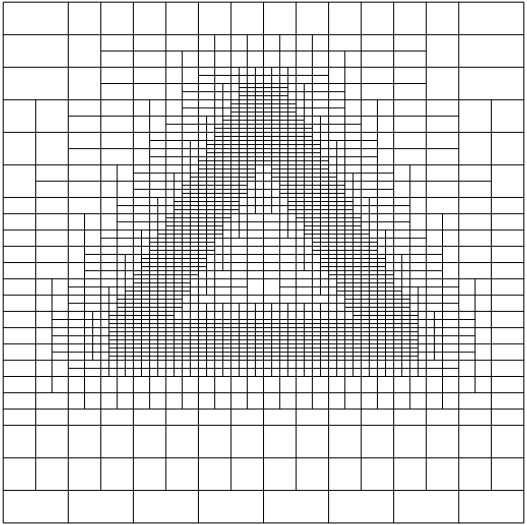
Effective Grading Refinement Strategy

Example: degree $(2, 2)$, Triangle $\to$ Circle $\to$ Square



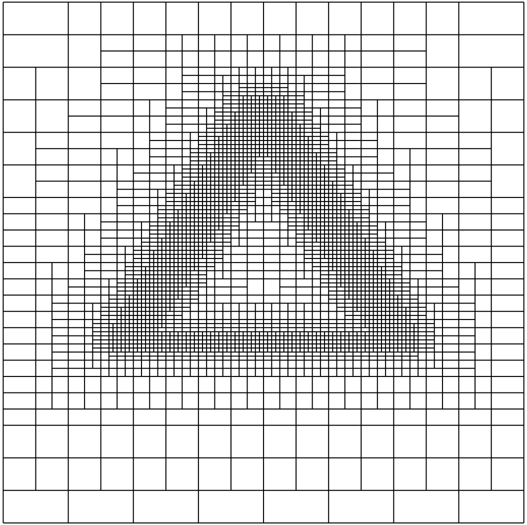
Effective Grading Refinement Strategy

Example: degree $(2, 2)$, Triangle $\rightarrow$ Circle $\rightarrow$ Square



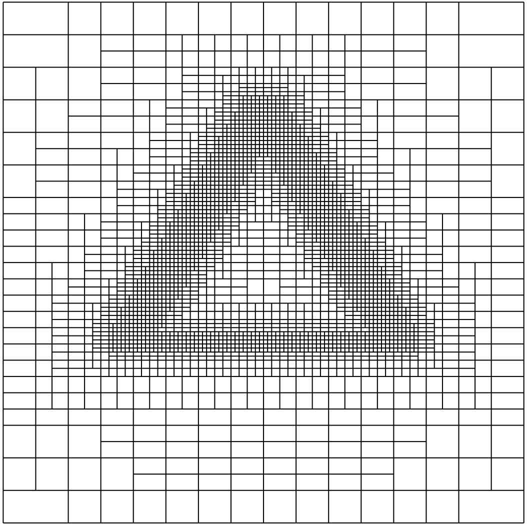
Effective Grading Refinement Strategy

Example: degree $(2, 2)$, Triangle $\rightarrow$ Circle $\rightarrow$ Square



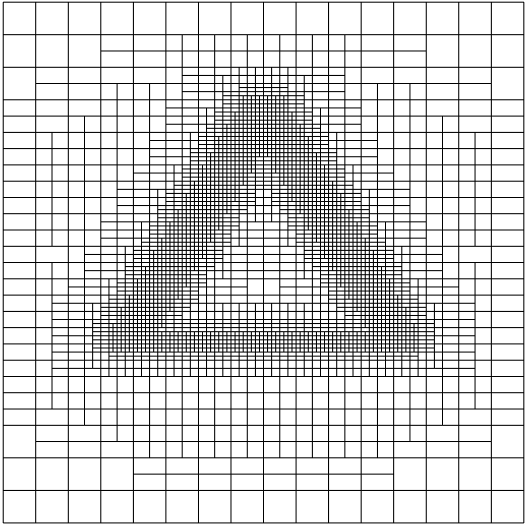
Effective Grading Refinement Strategy

Example: degree $(2, 2)$, Triangle $\rightarrow$ Circle $\rightarrow$ Square



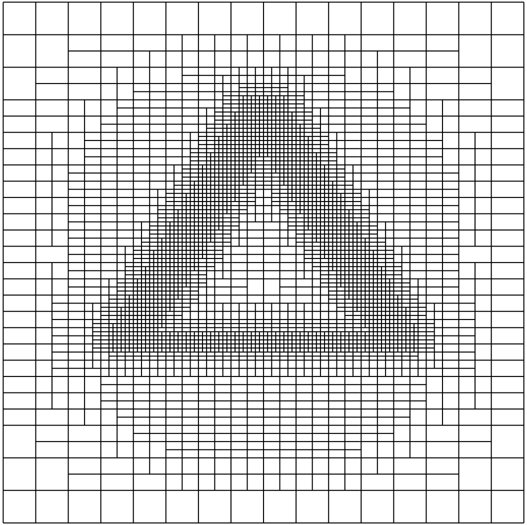
Effective Grading Refinement Strategy

Example: degree $(2,2)$, Triangle $\rightarrow$ Circle $\rightarrow$ Square



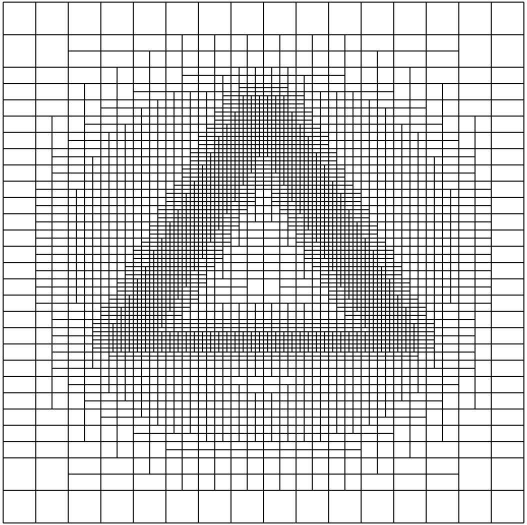
Effective Grading Refinement Strategy

Example: degree $(2,2)$, Triangle $\to$ Circle $\to$ Square



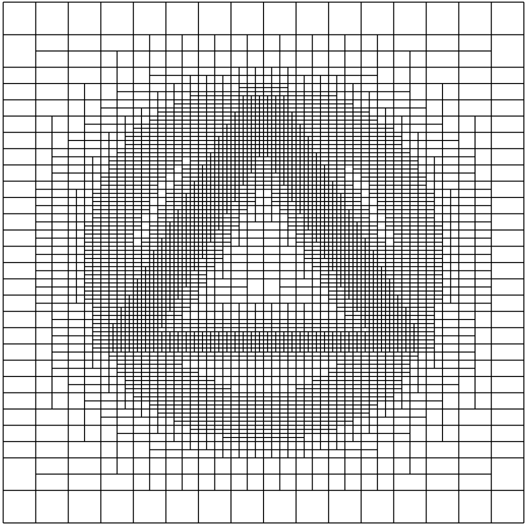
Effective Grading Refinement Strategy

Example: degree $(2,2)$, Triangle $\rightarrow$ Circle $\rightarrow$ Square



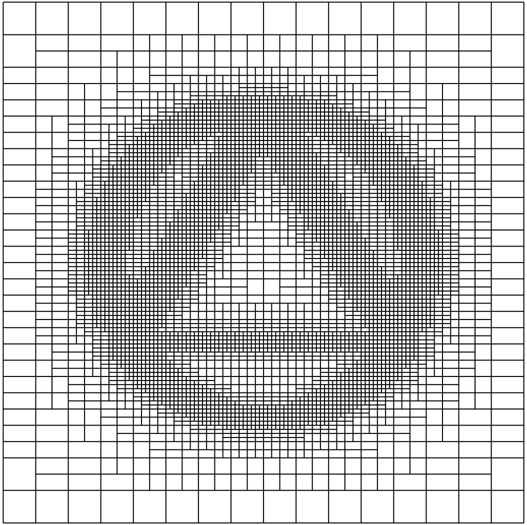
Effective Grading Refinement Strategy

Example: degree $(2, 2)$, Triangle $\rightarrow$ Circle $\rightarrow$ Square



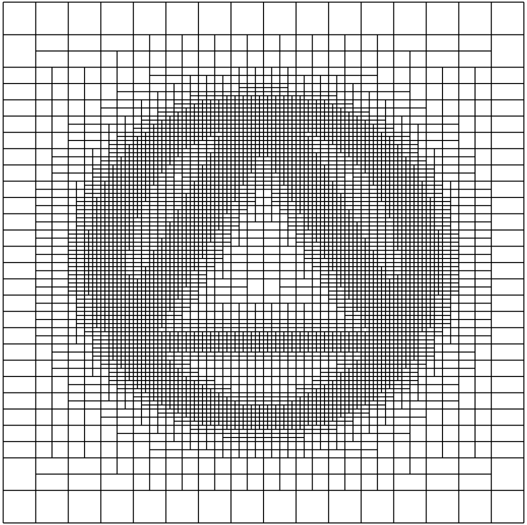
Effective Grading Refinement Strategy

Example: degree $(2, 2)$, Triangle $\rightarrow$ Circle $\rightarrow$ Square



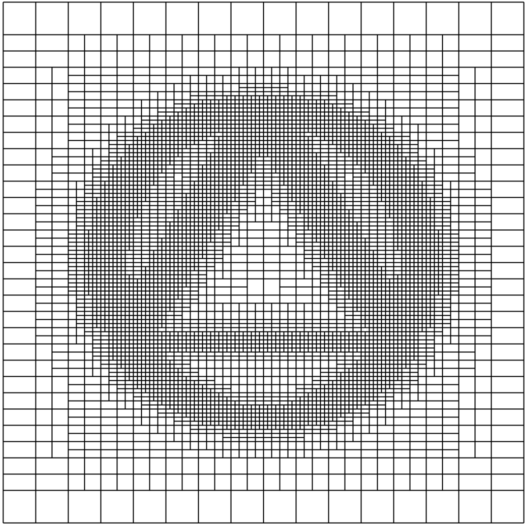
Effective Grading Refinement Strategy

Example: degree $(2,2)$, Triangle $\to$ Circle $\to$ Square



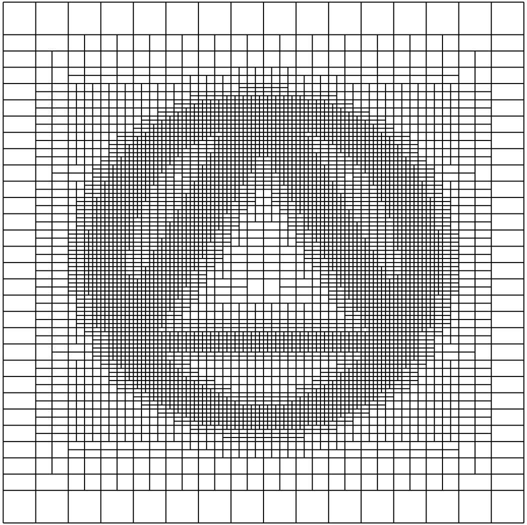
Effective Grading Refinement Strategy

Example: degree $(2, 2)$, Triangle $\to$ Circle $\to$ Square



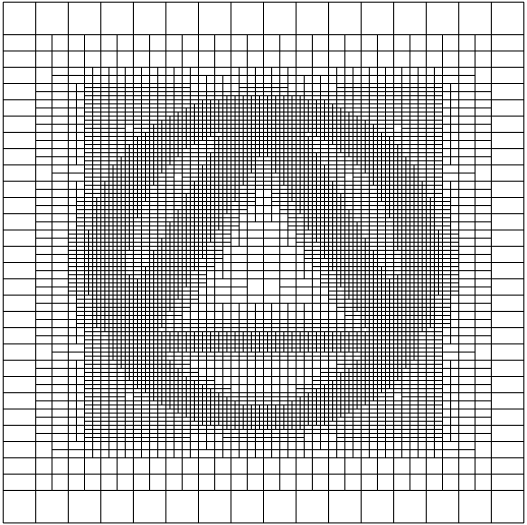
Effective Grading Refinement Strategy

Example: degree $(2, 2)$, Triangle $\rightarrow$ Circle $\rightarrow$ Square



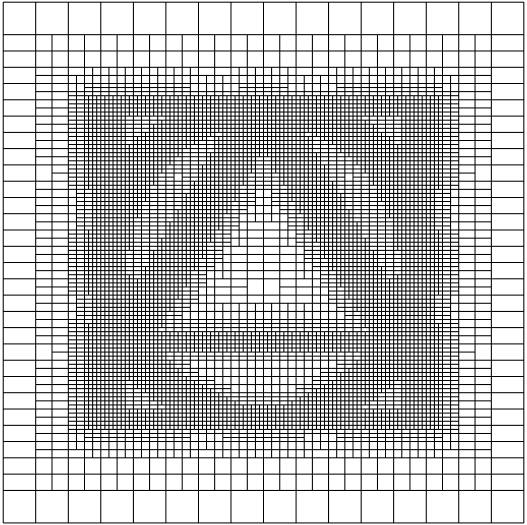
Effective Grading Refinement Strategy

Example: degree $(2,2)$, Triangle $\rightarrow$ Circle $\rightarrow$ Square

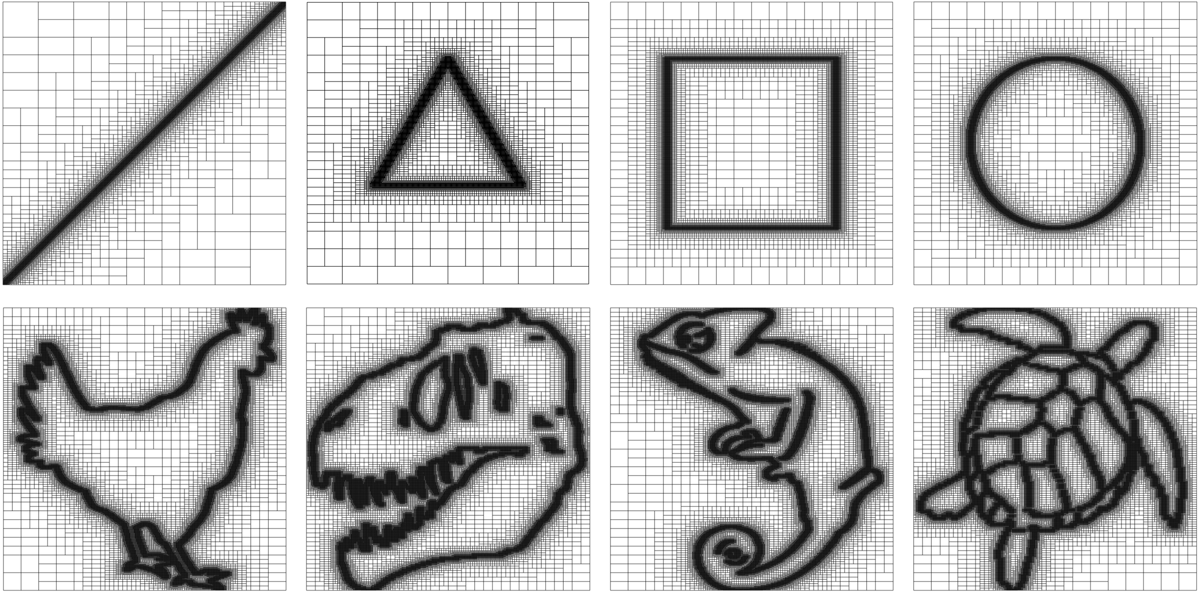


Effective Grading Refinement Strategy

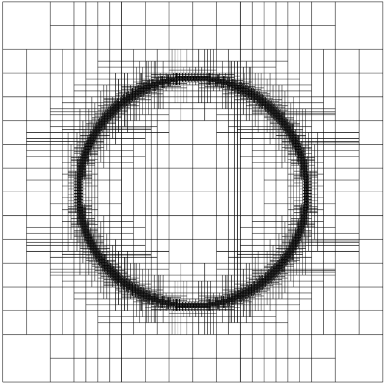
Example: degree $(2,2)$, Triangle $\rightarrow$ Circle $\rightarrow$ Square



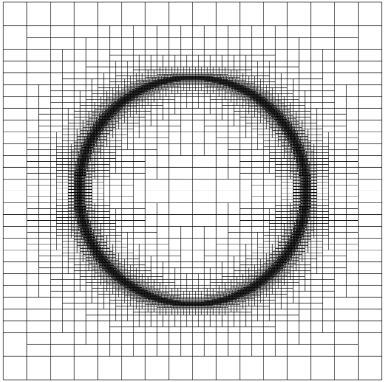
Effective Grading Refinement Strategy



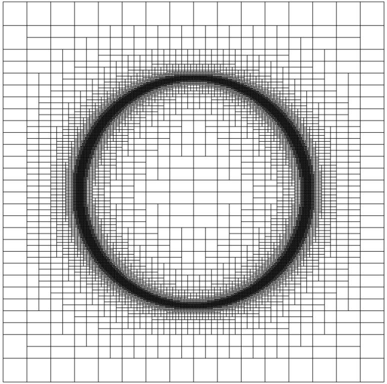
Comparison



N_2S_2 10818 LR B-splines



HLR 12374 LR B-splines



EG 15238 LR B-splines

One word on the spline space

$\mathcal{N}$ mesh with local insertions (not necessarily LR mesh)

$$\mathbb{S}(\mathcal{N}) := \left\{ \begin{aligned} f : \mathbb{R}^2 \to \mathbb{R} \ : & \operatorname{supp} f \subseteq \Omega, \\ & f|_\beta \text{ is a polynomial of bidegree } \boldsymbol{p} \text{ in any } \beta \text{ box of } \mathcal{N}, \\ & f \in C^{p_{3-k}-\mu(\gamma)}\text{-continuous across } \gamma \in \mathcal{N} \text{ in the } k\text{th direction.} \end{aligned} \right\}.$$

One word on the spline space

$\mathcal{N}$ mesh with local insertions (not necessarily LR mesh)

$$\mathbb{S}(\mathcal{N}) := \left\{ \begin{aligned} f : \mathbb{R}^2 \to \mathbb{R} \ : & \operatorname{supp} f \subseteq \Omega, \\ & f|_\beta \text{ is a polynomial of bidegree } \boldsymbol{p} \text{ in any } \beta \text{ box of } \mathcal{N}, \\ & f \in C^{p_{3-k}-\mu(\gamma)}\text{-continuous across } \gamma \in \mathcal{N} \text{ in the } k\text{th direction.} \end{aligned} \right\}.$$

Dimension formula: In general combinartorial part + homological part.

One word on the spline space

$\mathcal{N}$ mesh with local insertions (not necessarily LR mesh)

$$\mathbb{S}(\mathcal{N}) := \left\{ \begin{aligned} f : \mathbb{R}^2 \to \mathbb{R} \ : & \operatorname{supp} f \subseteq \Omega, \\ & f|_\beta \text{ is a polynomial of bidegree } \boldsymbol{p} \text{ in any } \beta \text{ box of } \mathcal{N}, \\ & f \in C^{p_{3-k}-\mu(\gamma)}\text{-continuous across } \gamma \in \mathcal{N} \text{ in the } k\text{th direction.} \end{aligned} \right\}.$$

Dimension formula: In general combinartorial part + homological part. The homological part makes it parametrization-dependent

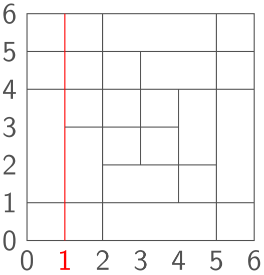
One word on the spline space

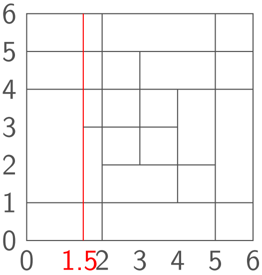
$\mathcal{N}$ mesh with local insertions (not necessarily LR mesh)

$$\mathbb{S}(\mathcal{N}) := \left\{ \begin{aligned} f : \mathbb{R}^2 \to \mathbb{R} \ : & \operatorname{supp} f \subseteq \Omega, \\ & f|_\beta \text{ is a polynomial of bidegree } \boldsymbol{p} \text{ in any } \beta \text{ box of } \mathcal{N}, \\ & f \in C^{p_{3-k}-\mu(\gamma)}\text{-continuous across } \gamma \in \mathcal{N} \text{ in the } k\text{th direction.} \end{aligned} \right\}.$$

Dimension formula: In general combinartorial part + homological part. The homological part makes it parametrization-dependent ⇒ Unstable ☹

$\dim \mathbb{S}(\mathcal{N}) = 36$





$\dim \mathbb{S}(\mathcal{N}) = 36 + 1$

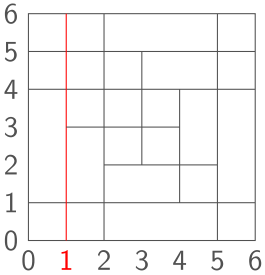
One word on the spline space

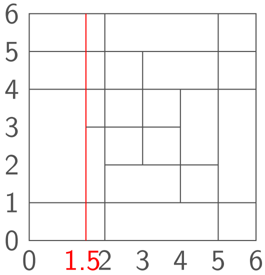
$\mathcal{N}$ mesh with local insertions (not necessarily LR mesh)

$$\mathbb{S}(\mathcal{N}) := \left\{ \begin{array}{ll} f : \mathbb{R}^2 \to \mathbb{R} & : \operatorname{supp} f \subseteq \Omega, \\ & f|_\beta \text{ is a polynomial of bidegree } \boldsymbol{p} \text{ in any } \beta \text{ box of } \mathcal{N}, \\ & f \in C^{p_{3-k}-\mu(\gamma)}\text{-continuous across } \gamma \in \mathcal{N} \text{ in the } k\text{th direction.} \end{array} \right\}.$$

Dimension formula: In general combinartorial part + homological part. The homological part makes it parametrization-dependent $\Rightarrow$ Unstable ☹

$\dim \mathbb{S}(\mathcal{N}) = 36$





$\dim \mathbb{S}(\mathcal{N}) = 36 + 1$

On LR meshes only combinatorial ☺.

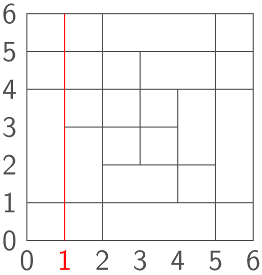
One word on the spline space

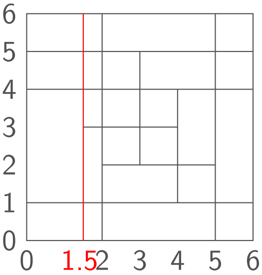
$\mathcal{N}$ mesh with local insertions (not necessarily LR mesh)

$$\mathbb{S}(\mathcal{N}) := \left\{ \begin{array}{l} f : \mathbb{R}^2 \to \mathbb{R} \ : \operatorname{supp} f \subseteq \Omega, \\ \qquad f|_\beta \text{ is a polynomial of bidegree } \boldsymbol{p} \text{ in any } \beta \text{ box of } \mathcal{N}, \\ \qquad f \in C^{p_{3-k}-\mu(\gamma)}\text{-continuous across } \gamma \in \mathcal{N} \text{ in the } k\text{th direction.} \end{array} \right\}.$$

Dimension formula: In general combinartorial part + homological part. The homological part makes it parametrization-dependent $\Rightarrow$ Unstable ☹

$\dim \mathbb{S}(\mathcal{N}) = 36$





$\dim \mathbb{S}(\mathcal{N}) = 36 + 1$

On LR meshes only combinatorial ☺. HB, THB, LR, $\ldots \subseteq \mathbb{S}(\mathcal{N})$.

Comparison

Adaptivity:

Comparison

Adaptivity: Local Refinement

Comparison

Adaptivity: Local Refinement + Change Region at any time

Comparison

Adaptivity: Local Refinement + Change Region at any time

Grading:

Comparison

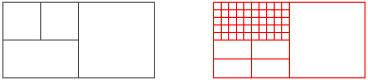
Adaptivity: Local Refinement + Change Region at any time

Grading: Shape Regularity: No skinny elements

Comparison

Adaptivity: Local Refinement + Change Region at any time

Grading: Shape Regularity: No skinny elements + Local Quasi Uniformity:

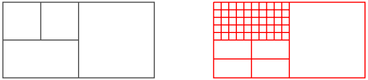


no large boxes side by side small boxes

Comparison

Adaptivity: Local Refinement + Change Region at any time

Grading: Shape Regularity: No skinny elements + Local Quasi Uniformity:



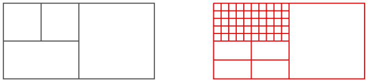
no large boxes side by side small boxes

Completeness:

Comparison

Adaptivity: Local Refinement + Change Region at any time

Grading: Shape Regularity: No skinny elements + Local Quasi Uniformity:



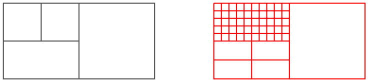
no large boxes side by side small boxes

Completeness: The LR B-splines span the full ambient spline space $\mathbb{S}(\mathcal{N})$.

Comparison

Adaptivity: Local Refinement + Change Region at any time

Grading: Shape Regularity: No skinny elements + Local Quasi Uniformity:



no large boxes side by side small boxes

Completeness: The LR B-splines span the full ambient spline space $\mathbb{S}(\mathcal{N})$.

Marking:

Comparison

Adaptivity: Local Refinement + Change Region at any time

Grading: Shape Regularity: No skinny elements + Local Quasi Uniformity:

no large boxes side by side small boxes

Completeness: The LR B-splines span the full ambient spline space $\mathbb{S}(\mathcal{N})$.

Marking: ▶ function-based: refine those LR B-splines contributing more to the error,

Comparison

Adaptivity: Local Refinement + Change Region at any time

Grading: Shape Regularity: No skinny elements + Local Quasi Uniformity:

no large boxes side by side small boxes

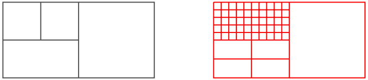
Completeness: The LR B-splines span the full ambient spline space $\mathbb{S}(\mathcal{N})$.

Marking:
- function-based: refine those LR B-splines contributing more to the error,
- box-based: refine those boxes in which a larger error is committed.

Comparison

Adaptivity: Local Refinement + Change Region at any time

Grading: Shape Regularity: No skinny elements + Local Quasi Uniformity:



no large boxes side by side small boxes

Completeness: The LR B-splines span the full ambient spline space $\mathbb{S}(\mathcal{N})$.

Marking:
- function-based: refine those LR B-splines contributing more to the error,
- box-based: refine those boxes in which a larger error is committed.

Comparison

Adaptivity: Local Refinement + Change Region at any time

Grading: Shape Regularity: No skinny elements + Local Quasi Uniformity:

no large boxes side by side small boxes

Completeness: The LR B-splines span the full ambient spline space $\mathbb{S}(\mathcal{N})$.

Marking:
- function-based: refine those LR B-splines contributing more to the error,
- box-based: refine those boxes in which a larger error is committed.

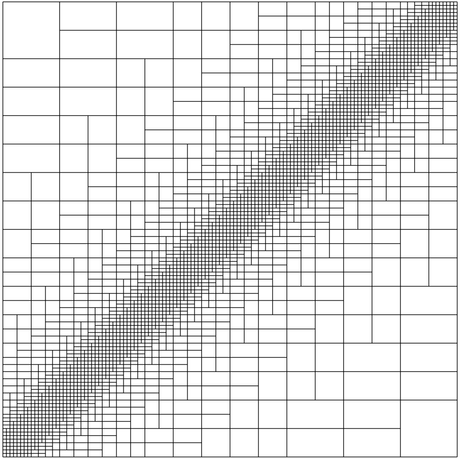
	(Loc.) Lin. Ind.	Adaptivity	Grading	Completeness	Marking
N_2S_2 strategy	✔	✔	✘	?	function-based
Hierarchical	Under Assumptions*	✘	✔	✔	box-based
Effective Grading	✔	✔	✔	✔	box-based

*fix maximal resolution and region of refinement *a priori*

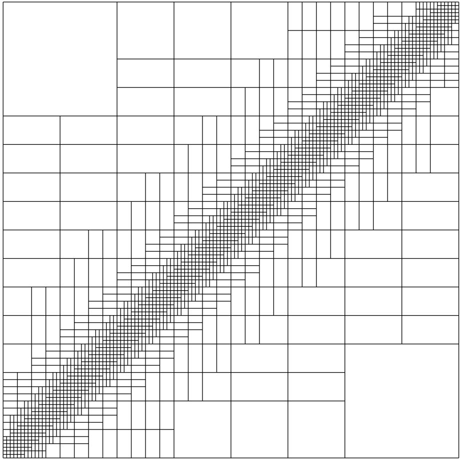
Conjecture: Local Linear Independence $\Rightarrow$ Completeness.

Comparison of Adaptivity with and without Grading

From a refinement localized on a diagonal we switch to the other diagonal to form an "X".



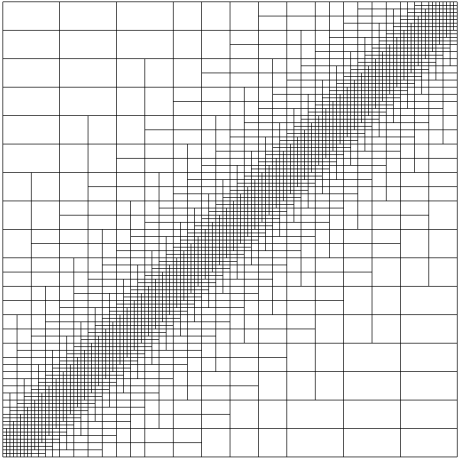
EG: Adaptivity & Grading



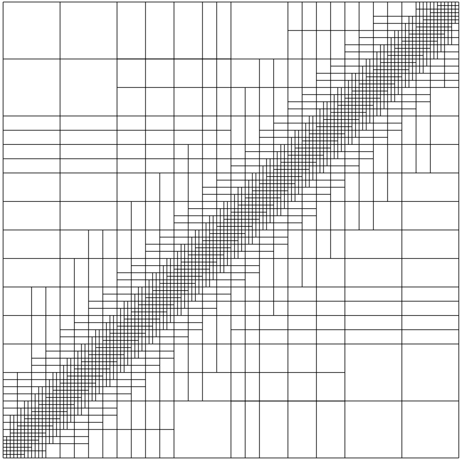
N_2S_2: Adaptivity but no Grading

Comparison of Adaptivity with and without Grading

From a refinement localized on a diagonal we switch to the other diagonal to form an "X".



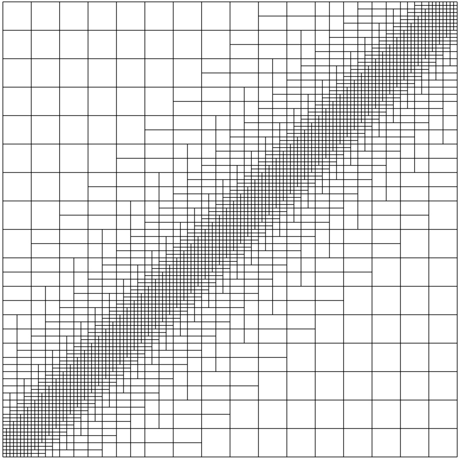
EG: Adaptivity & Grading



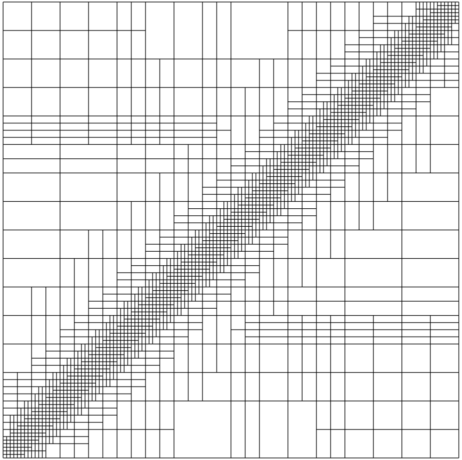
N_2S_2: Adaptivity but no Grading

Comparison of Adaptivity with and without Grading

From a refinement localized on a diagonal we switch to the other diagonal to form an “X”.



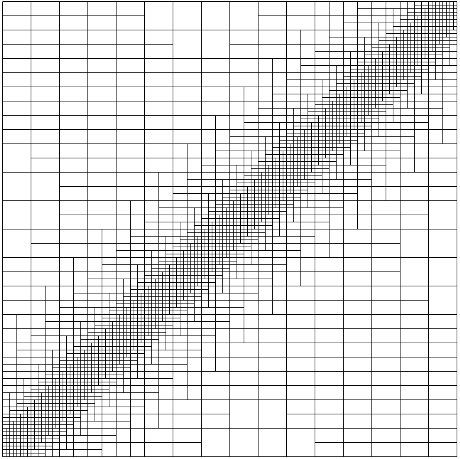
EG: Adaptivity & Grading



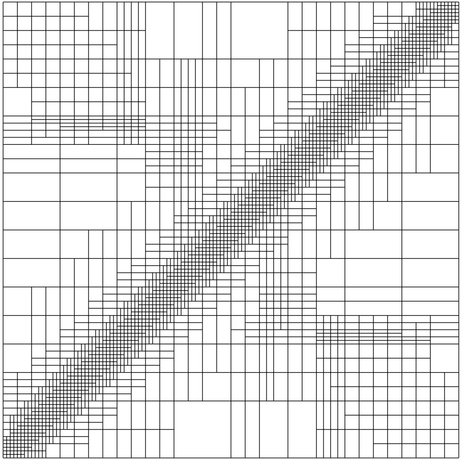
N_2S_2: Adaptivity but no Grading

Comparison of Adaptivity with and without Grading

From a refinement localized on a diagonal we switch to the other diagonal to form an "X".



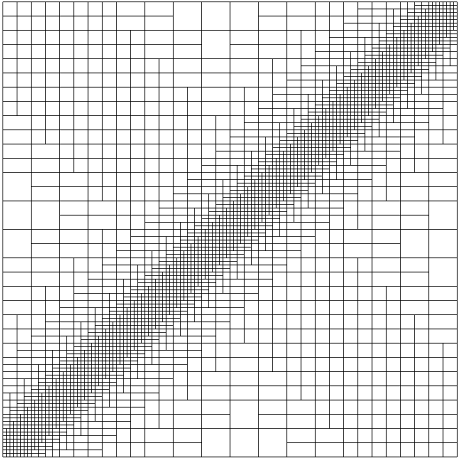
EG: Adaptivity & Grading



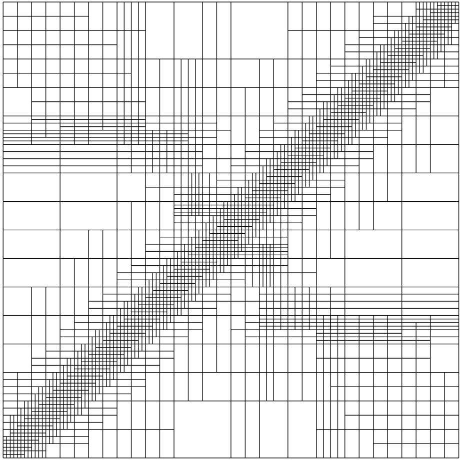
N_2S_2: Adaptivity but no Grading

Comparison of Adaptivity with and without Grading

From a refinement localized on a diagonal we switch to the other diagonal to form an "X".



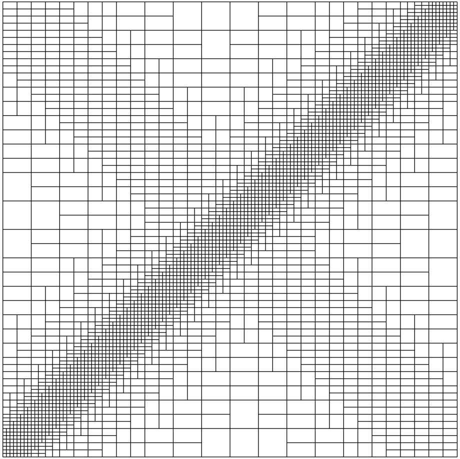
EG: Adaptivity & Grading



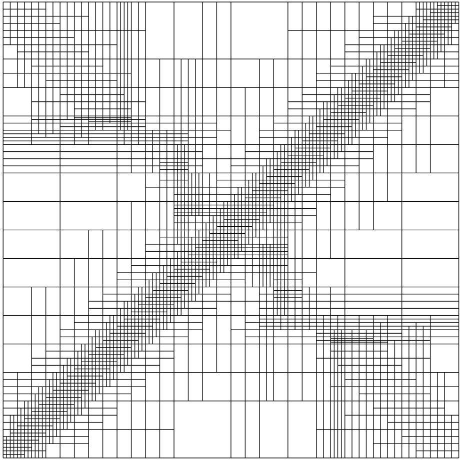
N_2S_2: Adaptivity but no Grading

Comparison of Adaptivity with and without Grading

From a refinement localized on a diagonal we switch to the other diagonal to form an “X”.



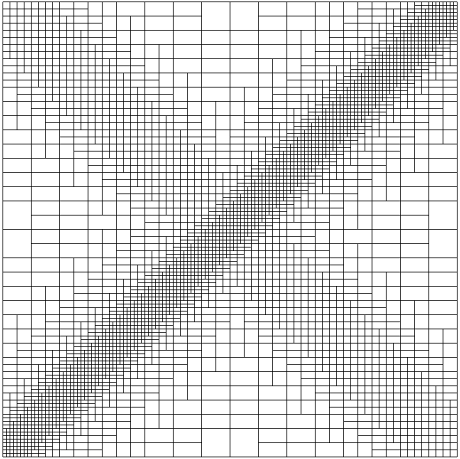
EG: Adaptivity & Grading



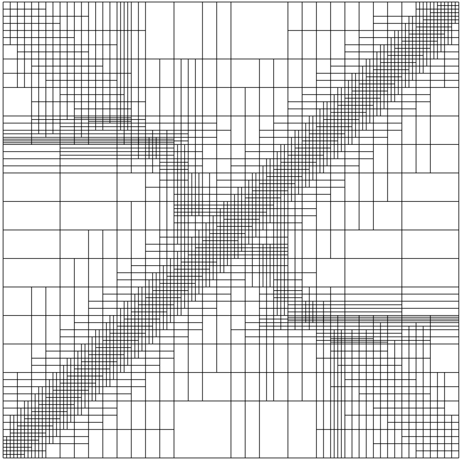
N_2S_2: Adaptivity but no Grading

Comparison of Adaptivity with and without Grading

From a refinement localized on a diagonal we switch to the other diagonal to form an "X".



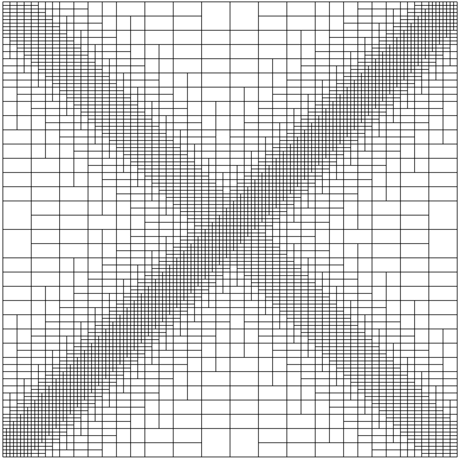
EG: Adaptivity & Grading



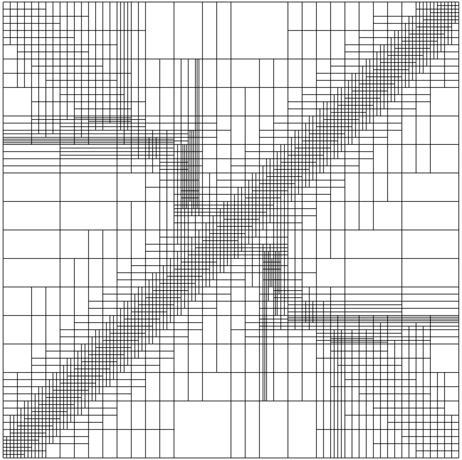
N_2S_2: Adaptivity but no Grading

Comparison of Adaptivity with and without Grading

From a refinement localized on a diagonal we switch to the other diagonal to form an "X".



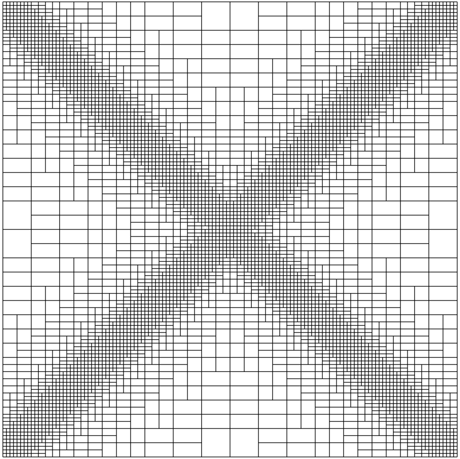
EG: Adaptivity & Grading



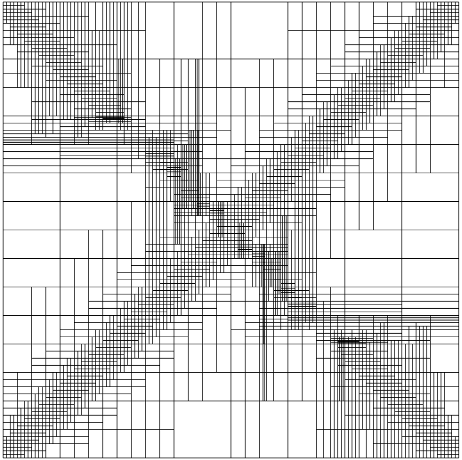
N_2S_2: Adaptivity but no Grading

Comparison of Adaptivity with and without Grading

From a refinement localized on a diagonal we switch to the other diagonal to form an "X".



EG: Adaptivity & Grading



N_2S_2: Adaptivity but no Grading

References

- Bressan, A. (2013). *Some properties of LR-splines*. CAGD, 30(8), 778-794.
- Bressan, A., & Jüttler, B. (2015). *A hierarchical construction of LR meshes in 2D*. CAGD, 37, 9-24.
- Dokken, T., Lyche, T., & Pettersen, K. F. (2013). *Polynomial splines over locally refined box-partitions*. CAGD, 30(3), 331-356.
- Johannessen, K. A., Kvamsdal, T., & Dokken, T. (2014). *Isogeometric analysis using LR B-splines*. CMAME, 269, 471-514.
- Patrizi, F. (2022). *Effective grading refinement for locally linearly independent LR B-splines*. BIT Numerical Mathematics, 1-20.
- Patrizi, F., & Dokken, T. (2020). *Linear dependence of bivariate Minimal Support and Locally Refined B-splines over LR-meshes*. CAGD, 77, 101803.
- Patrizi, F., Manni, C., Pelosi, F., & Speleers, H. (2020). *Adaptive refinement with locally linearly independent LR B-splines: Theory and applications*. CMAME, 369, 113230.