Locally Refined B-splines and Linear Independence

Tor Dokken, Tom Lyche, Kjell Fredrik Pettersen

Technologies for Local Refined Splines

	T-splines	Truncated Hierar-chical B-splines*	LR B-splines	PHT-splines
Approach	Algorithmic	Spline space	Spline space	Spine space
Generalization of	NonUniform B-splines	Uniform B-splines	NonUniform B-splines	Splines over T-mesh
Minimal refinement	One knotline segment	Split all knot intervals of one B-spline	One knotline segment	One knotline segment
Partition of unity	Rational Scaling	Truncated** B-splines	Scaled B-splines	Tailored basis
Linear independence	When analysis suitable	Yes	When hand-in-hand or successful overload elimination	Tailored basis
Specification of refinement	Insertion of vertex in vertex T-grid	Refinement levels and regions in parameter domain	Knotline segment in box-mesh projected on surface, as T-splines or Hierachi-cal B-splines	Mesh rectangle in T-mesh

*Kraft (1997) Hierarchical B-splines

**Giannelli (2012) – Truncated Hierarchical B-splines

Knotline T-mesh projected on to LR B-spline represented surface

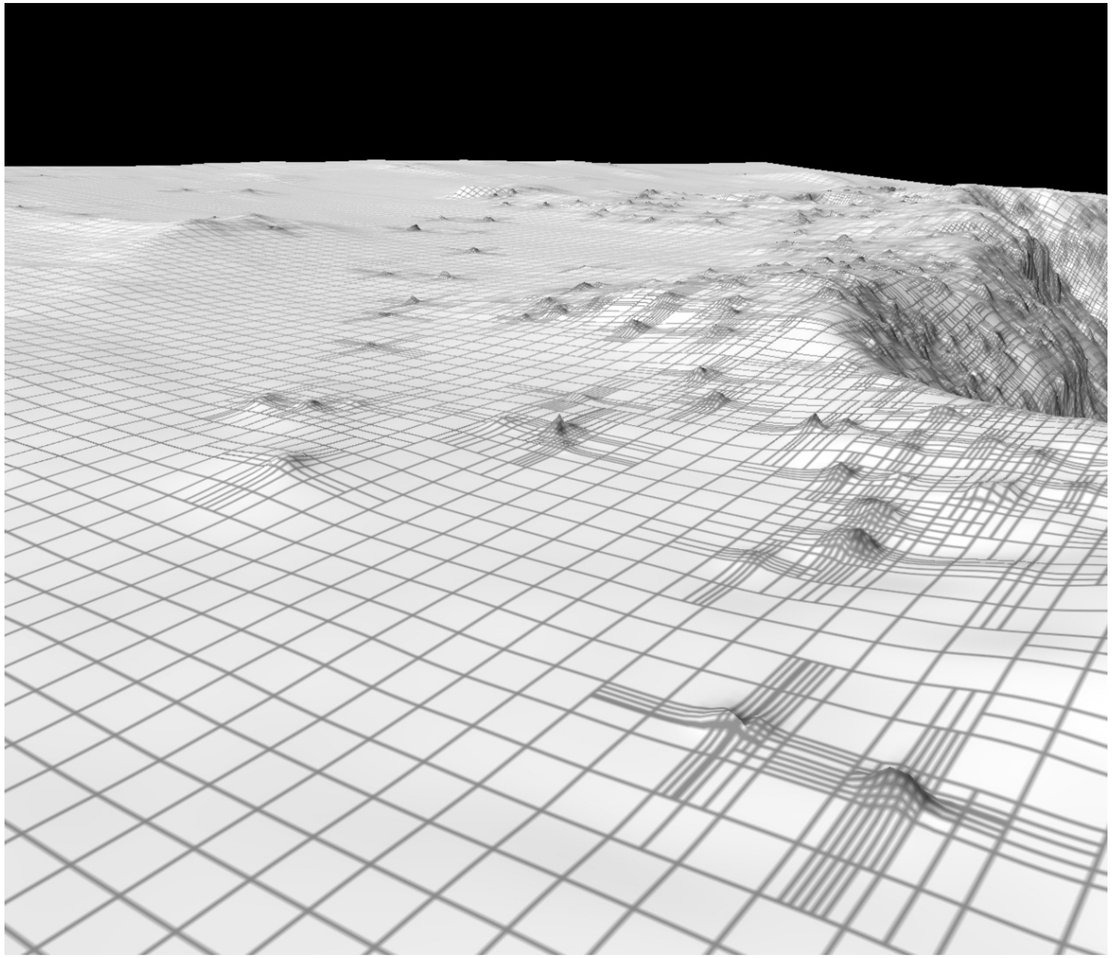
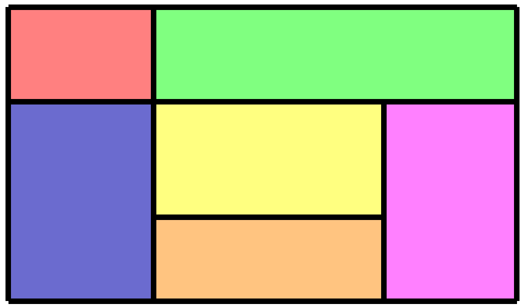


Illustration by:
Odd Andersen,
SINTEF

Box-partitions

- Box-partitions - Rectangular subdivision of regular domain d-box $\mathbb{R}^d$

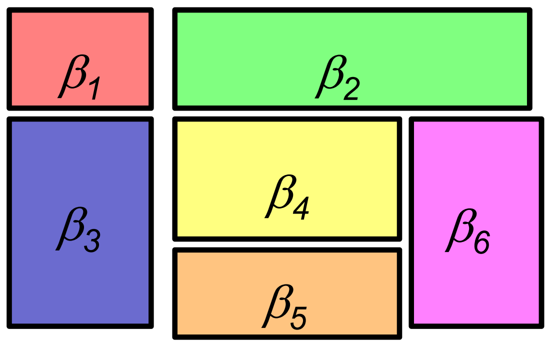
$$\Omega \subseteq \mathbb{R}^d$$
$$\Omega = [a_1, b_1] \times \cdots \times [a_d, b_d]$$
$$a_i < b_i,\ 1 \leq i \leq d$$



$\Omega \subseteq \mathbb{R}^2$

- Subdivision of Ω into smaller d -boxes

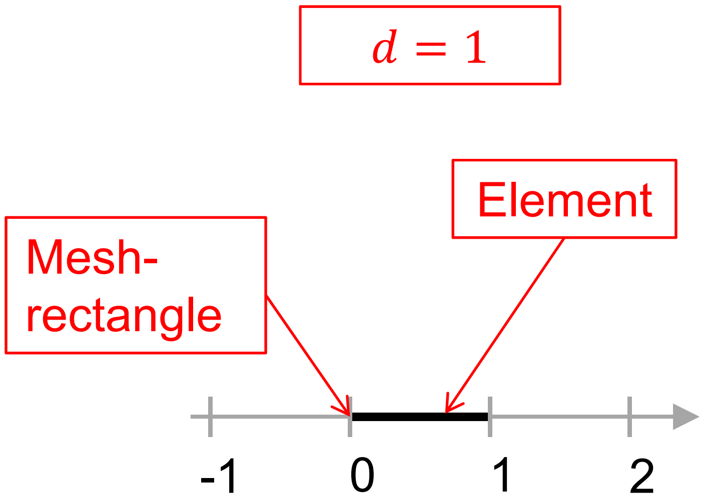
$$\mathcal{E} = \{\beta_1, \ldots, \beta_1\}$$
$$\beta_1 \cup \beta_2 \cup \cdots \cup \beta_n = \Omega$$
$$\beta_i^o \cap \beta_j^o = \emptyset,\ i \neq j$$

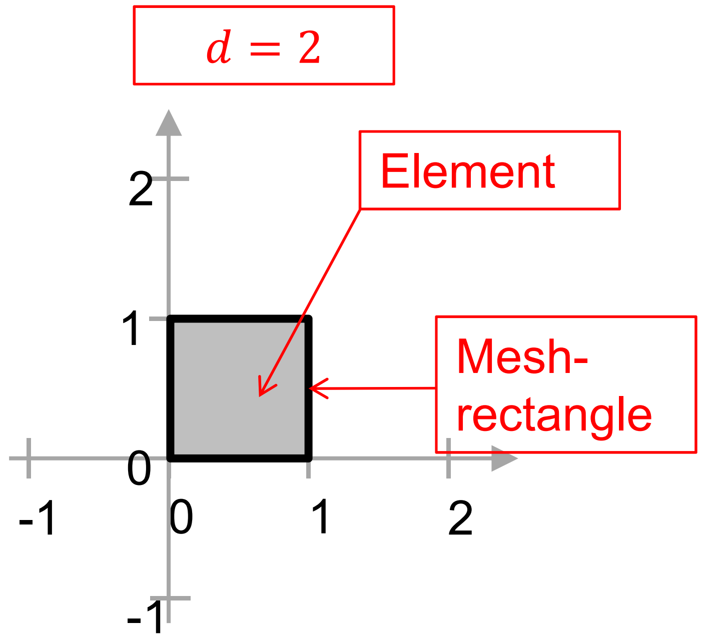


$\mathcal{E} = \{\beta_1, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6\}$

Important boxes

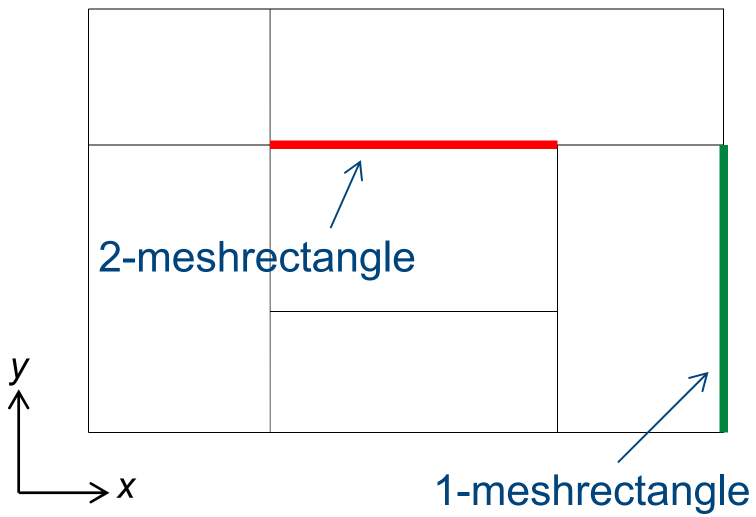
- If $\dim \beta = d$ then β is called an **element**.
- If $\dim \beta = d - 1$ there exists exactly one k such that $J_k = [a]$ is trivial. Then β is called a **mesh-rectangle**, a k-**mesh-rectangle** or a (k, a)-**mesh-rectangle**





Mesh-rectangles

Example, 2D



Example, 3D

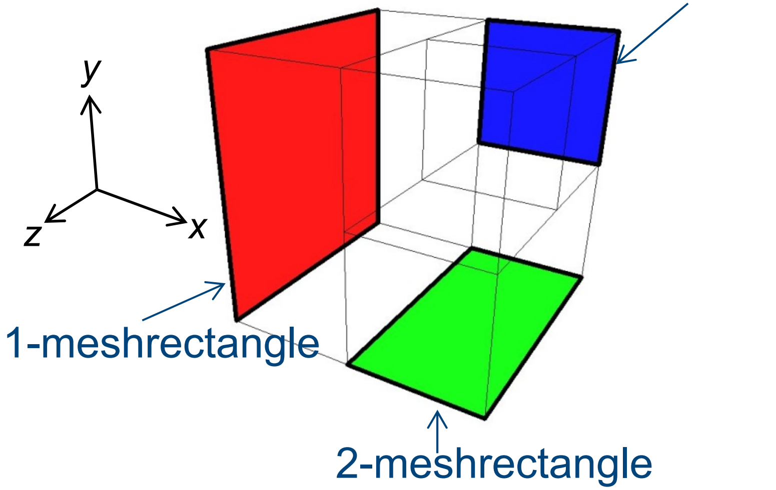
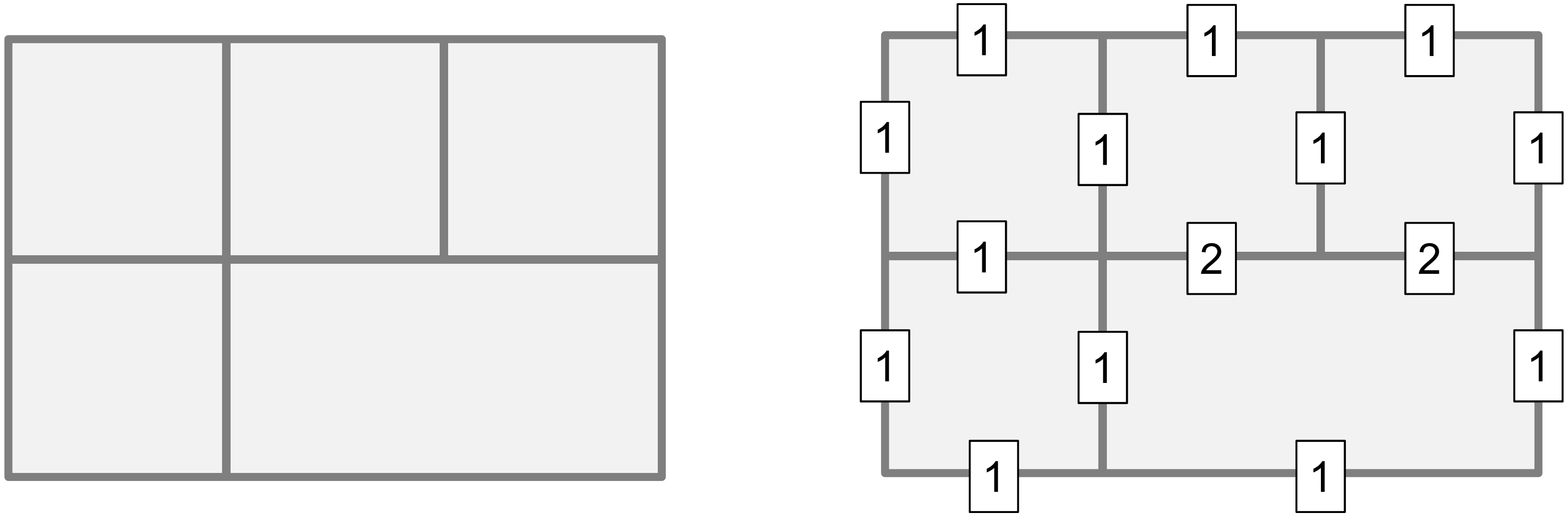


Illustration by: Kjell Fredrik Pettersen, SINTEF

μ-extended box-mesh (adding multiplicities)



- A multiplicity μ is assigned to each mesh-rectangle
- Supports variable knot multiplicity for Locally Refined B-splines, and local lower order continuity across mesh-rectangles.
- Compatible with nonuniform univariate B-splines

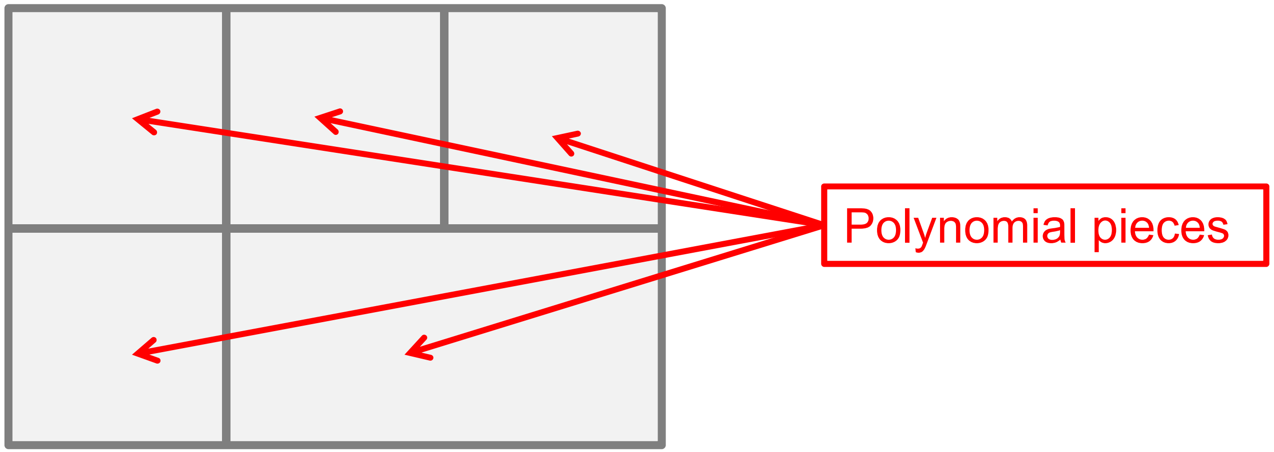
Polynomials of component degree

On each element the spline is a polynomial.

We define polynomials of component degree at most $p_k, k = 1, \dots, d$ by:

$$\Pi_{\boldsymbol{p}}^d = \left\{ f: \mathbb{R}^d \to \mathbb{R}: f(\boldsymbol{x}) = \sum_{\boldsymbol{0} \le \boldsymbol{i} \le \boldsymbol{p}} c_{\boldsymbol{i}} \boldsymbol{x}^{\boldsymbol{i}}, c_{\boldsymbol{i}} \text{ in } \mathbb{R} \text{ for all } \boldsymbol{i} \right\}.$$

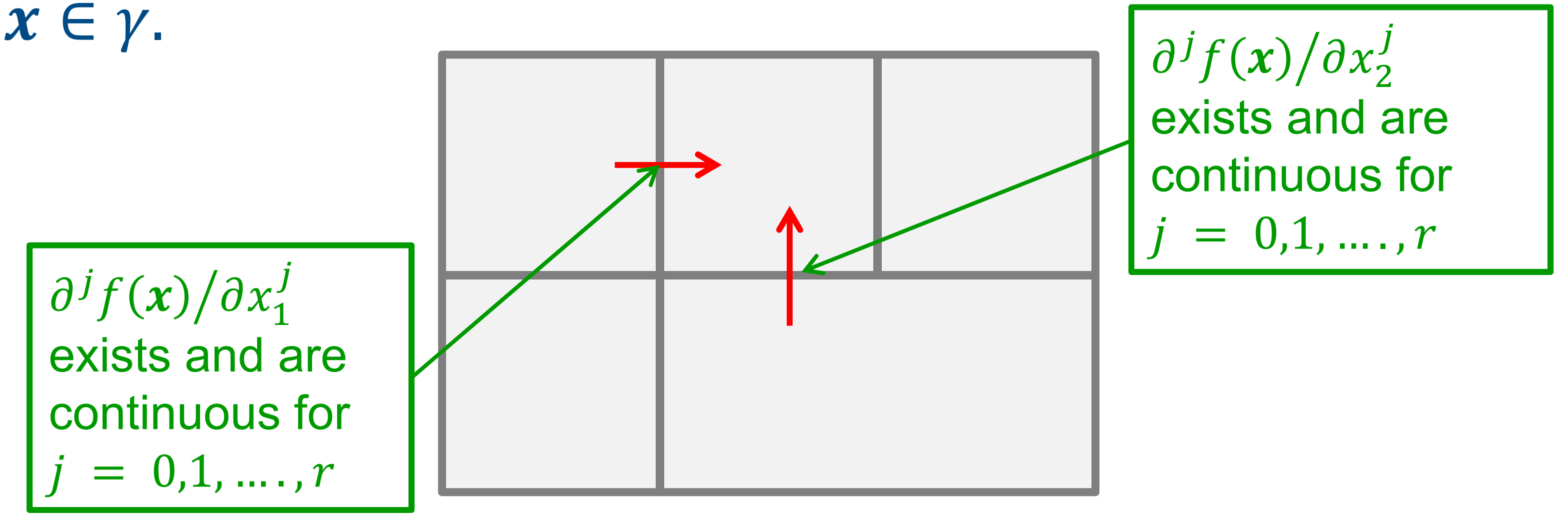
$\boldsymbol{p} = (p_1, \dots, p_d)$
$\boldsymbol{i} = (i_1, \dots, i_d)$



Continuity across mesh-rectangles

Given a function $f: [\boldsymbol{a}, \boldsymbol{b}] \to \mathbb{R}$, and let $\gamma \in \mathcal{F}_{d-1,k}(\mathcal{E})$ be any k-mesh-rectangle in $[\boldsymbol{a}, \boldsymbol{b}]$ for some $1 \leq k \leq d$.

We say that $f \in C^r(\gamma)$ if the partial derivatives $\partial^j f(\boldsymbol{x})/\partial x_k^j$ exists and are continuous for $j = 0,1, \ldots, r$ and all $\boldsymbol{x} \in \gamma$.

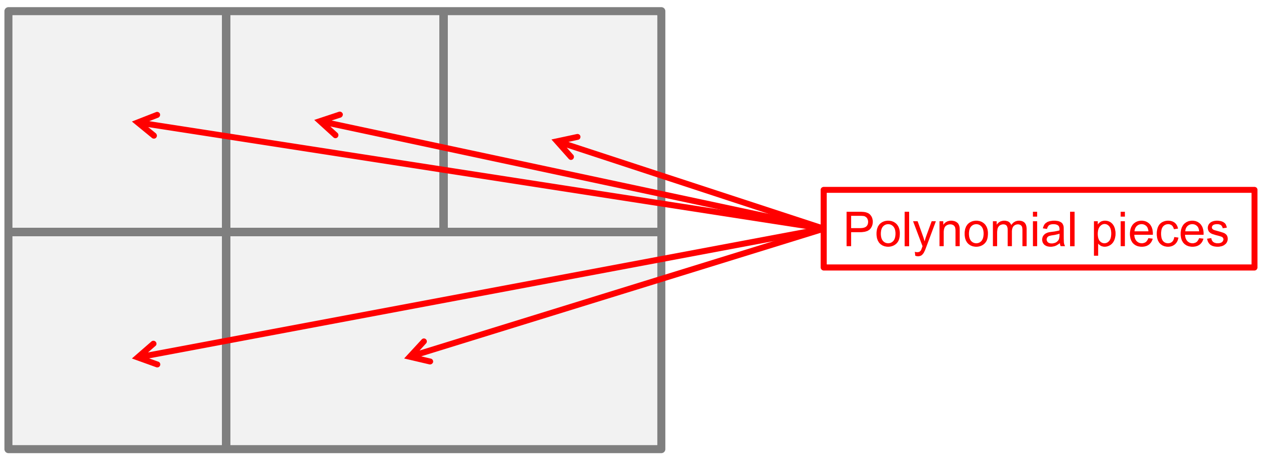


Piecewise polynomial space

We define the piecewise polynomial space

$$\mathbb{P}_{\boldsymbol{p}}(\boldsymbol{\mathcal{E}}) = \{f : [\boldsymbol{a}, \boldsymbol{b}] \rightarrow \mathbb{R} : f|_{\beta} \in \Pi^{d}_{\boldsymbol{p}}, \beta \in \tilde{\mathcal{E}}\},$$

where $\mathcal{E}$ is obtained from $\boldsymbol{\mathcal{E}}$ using half-open intervals as for univate B-splines.



Spline space

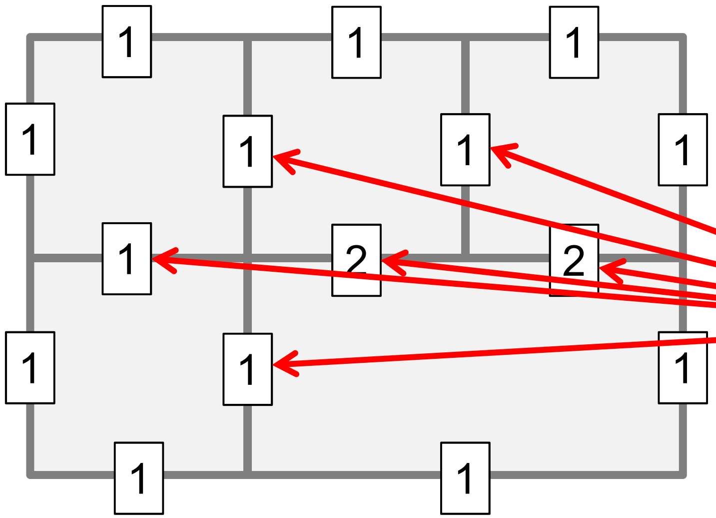
Polynomial degree in direction k

Continuity across k-mesh-rectangle γ

We define the spline space

$$\mathbb{S}_{\boldsymbol{p}}(\mathcal{M},\mu) = \{f \in \Pi_{\boldsymbol{p}}^{d}\left(\mathcal{E}(\mathcal{M})\right): f \in C^{p_k-\mu(\gamma)}(\gamma),$$
$$\forall\gamma \in \mathcal{F}_{d-1,k}\left(\mathcal{E}(\mathcal{M})\right), k = 1, \dots, d\}$$

All k-mesh-rectangles



Specify multiplicity, e.g., continuity across mesh-rectangle

How to measure dimensional of spline space of degree p over a μ-extended box partition $(\mathcal{M}, \mu)$.

- Dimension formula developed (Mourrain, Pettersen)

$$\dim \mathbb{S}_{\boldsymbol{p}}(\mathcal{M}, \boldsymbol{\mu}) = \sum_{\ell=0}^{d} (-1)^{d-\ell} \left(\sum_{\beta \in \mathcal{F}_\ell(\mathcal{M})} \prod_{k=1}^{d} (p_k - \mu_k(\beta) + 1) \right) - \sum_{q=0}^{d-1} (-1)^{d-q} \dim H_q(\widetilde{\mathfrak{S}}(\mathcal{N}))$$

Summing over all l-boxes of all dimensions

Dimension influenced by mesh-rectangle multiplicity

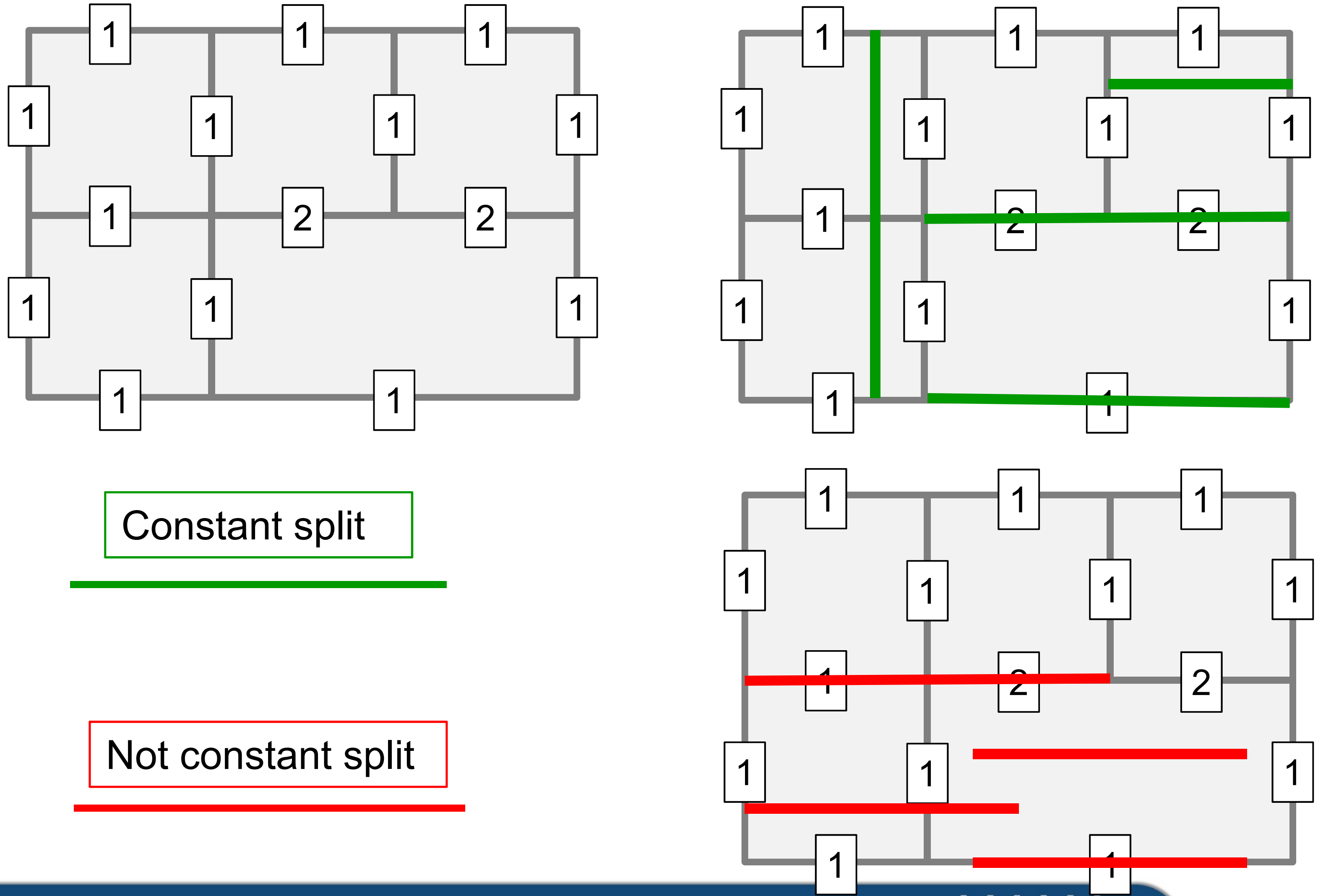
Combinatorial values calculated from topological structure

Homology terms

- In the case of 2-variate LR B-splines always zero

From talk by: Kjell Fredrik Pettersen, SINTEF

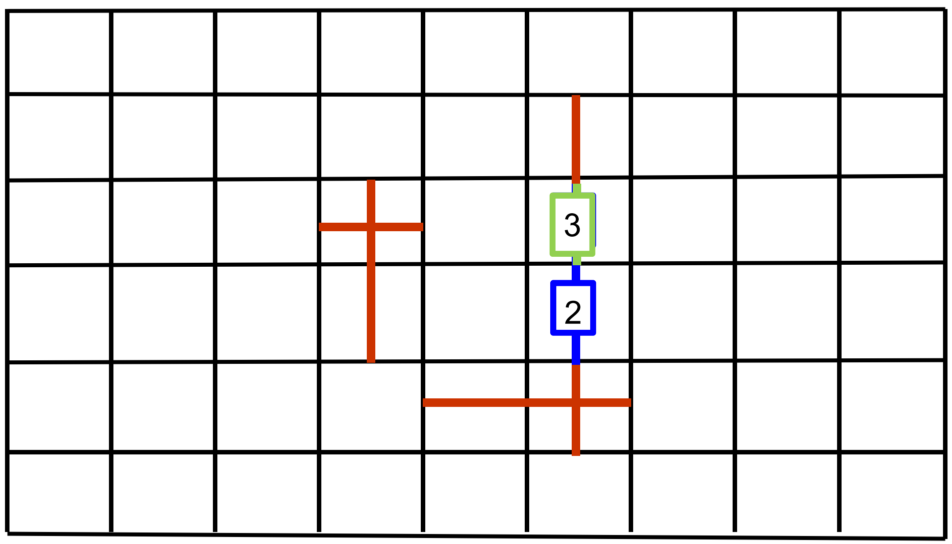
Refinement by inserting mesh-rectangles giving a constant split



μ-extended LR-mesh

A μ -extended LR-mesh is a μ-extended box-mesh $(\mathcal{M},\mu)$ where either

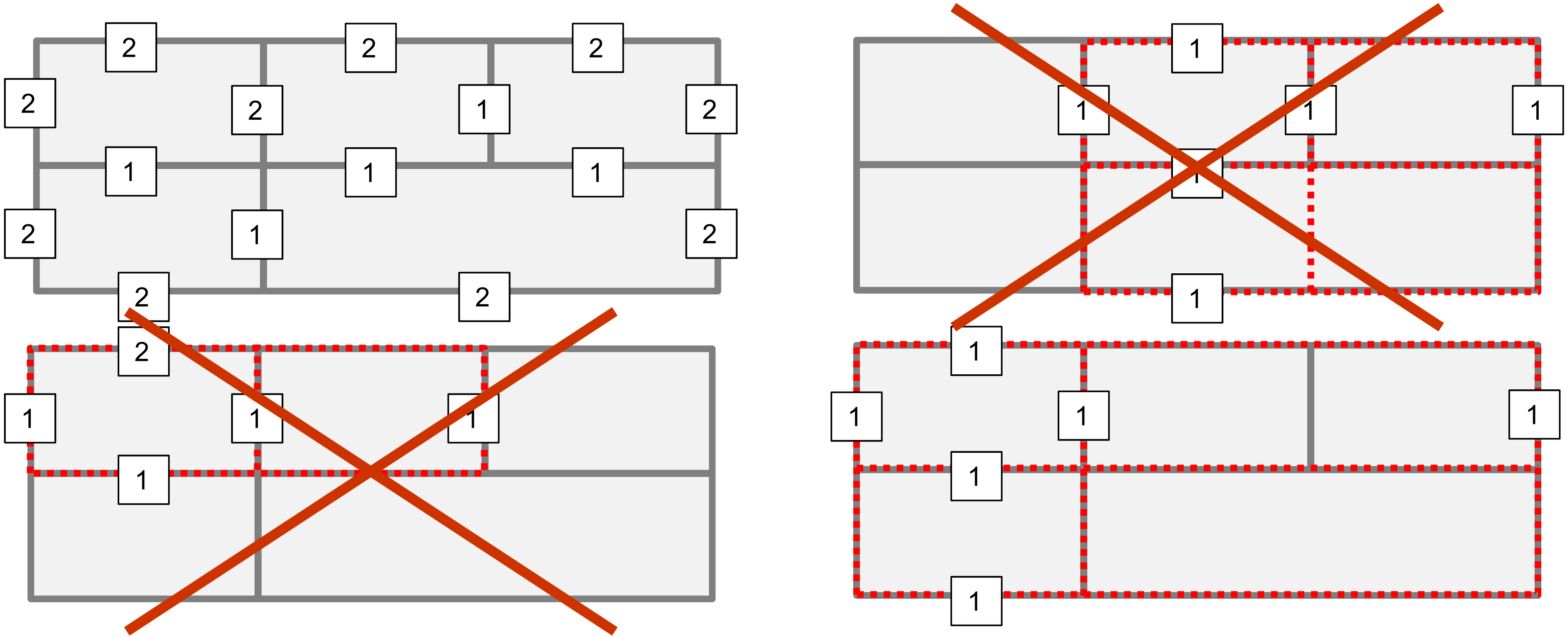
1. $(\mathcal{M},\mu)$ is a tensor-mesh with knot multiplicities or
2. $(\mathcal{M},\mu) = (\widetilde{\mathcal{M}} + \gamma, \tilde{\mu}_\gamma)$ where $(\widetilde{\mathcal{M}}, \tilde{\mu})$ is a μ -extended LR-mesh and γ is a constant split of $(\widetilde{\mathcal{M}}, \tilde{\mu})$.



All multiplicities not shown are 1.

LR B-spline

Let (M, μ) be a μ-extended LR-mesh in $\mathbb{R}^d$. A function $B: \mathbb{R}^d \rightarrow \mathbb{R}$ is called an LR B-spline of degree $\boldsymbol{p}$ on $(\mathcal{M}, \mu)$ if B is a tensor-product B-spline with minimal support in $(\mathcal{M}, \mu)$.



Splines on a μ-extended LR-mesh

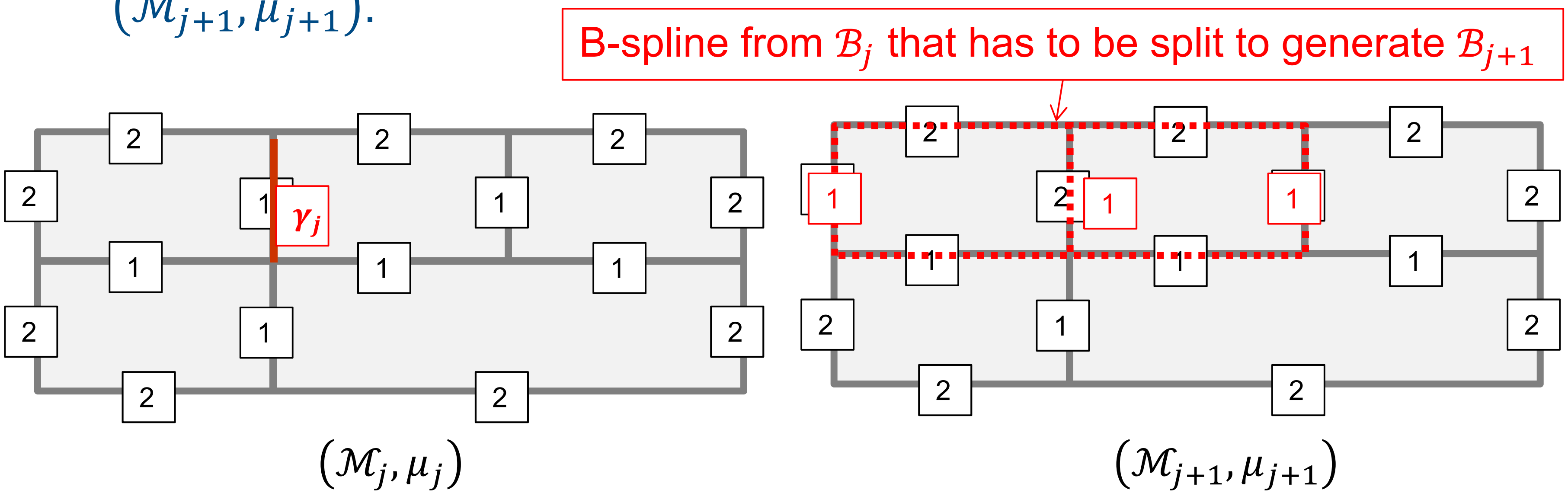
We define as sequence of $\boldsymbol{\mu}$-extended LR-meshes $(\mathcal{M}_1, \mu_1), \ldots, (\mathcal{M}_q, \mu_q)$ with corresponding collections of minimal support B-splines $\mathcal{B}_1, \ldots, \mathcal{B}_q$.

$(\mathcal{M}_1, \mu_1)$,	$(\mathcal{M}_2, \mu_2)$,	…	$(\mathcal{M}_j, \mu_j)$,	$(\mathcal{M}_{j+1}, \mu_{j+1})$	…	$(\mathcal{M}_q, \mu_q)$
$\mathcal{B}_1$,	$\mathcal{B}_2$,	…	$\mathcal{B}_j$,	$\mathcal{B}_{j+1}$,	…	$\mathcal{B}_q$

Creating $(\mathcal{M}_{j+1}, \mu_{j+1})$ from $(\mathcal{M}_j, \mu_j)$

Insert a mesh-rectangles γ_j that increases the number of B-splines.
More specifically:

- γ_j splits $(\mathcal{M}_j, \mu_j)$ in a constant split.
- at least on B-spline in $\mathcal{B}_j$ does not have minimal support in $(\mathcal{M}_{j+1}, \mu_{j+1})$.

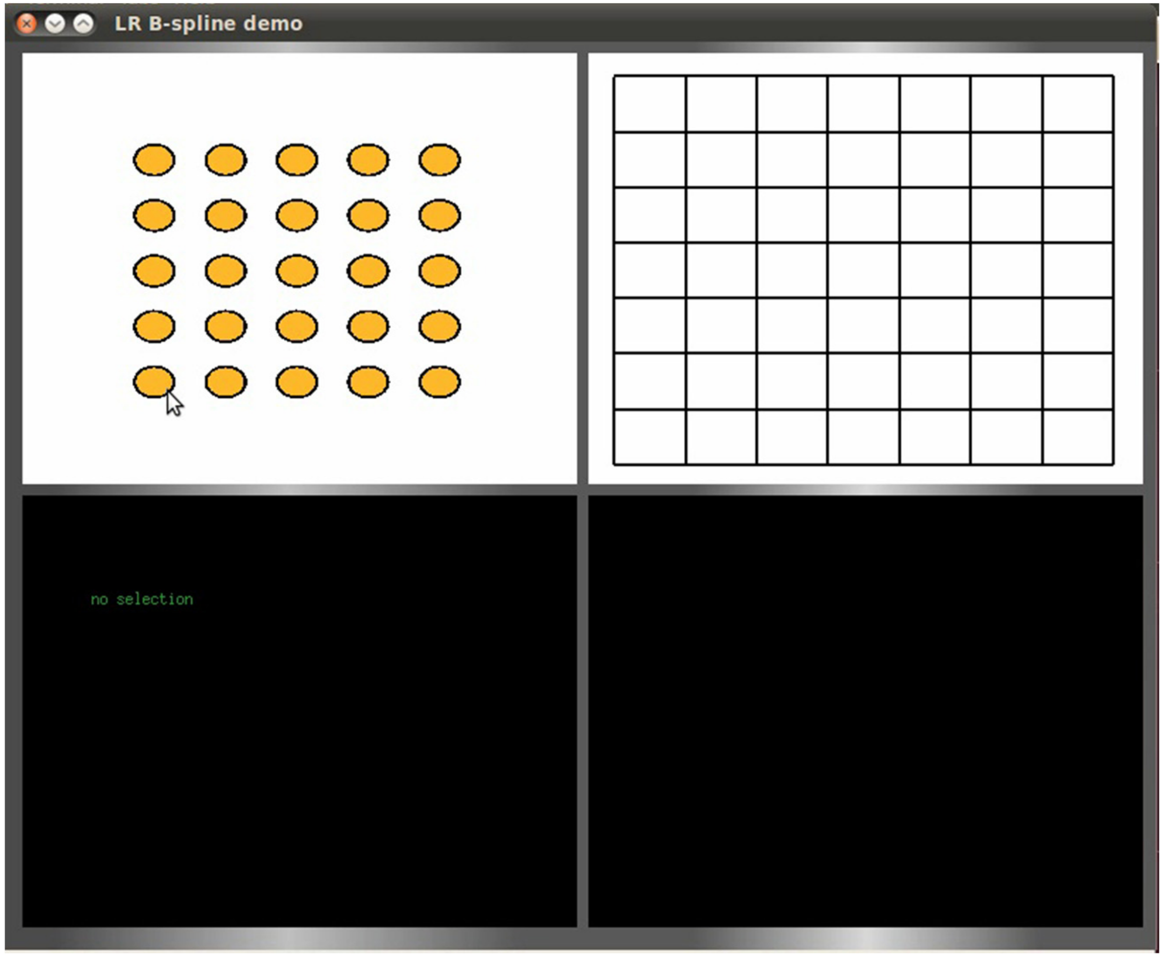


- After inserting γ_j we start a process to generate a collection of minimal support B-splines $\mathcal{B}_{j+1}$ over $(\mathcal{M}_{j+1}, \mu_{j+1})$ from $\mathcal{B}_j$.

LR B-splines and partition of unity

- The LR B-spline refinement starts from a partition of unity tensor product B-spline basis.
- By accumulating the weights α_1 and α_2 as scaling factors for the LR B-splines, partition of unity is maintained throughout the refinement for the scaled collection of tensor product B-splines
- The partition of unity properties gives the coefficients of LR B-splines the same geometric interpretation as B-splines and T-splines.
  - The spatial interrelation of the coefficients is more intricate than for T-splines as the refinements allowed are more general.
  - This is, however, no problem as in general algorithms calculate the coefficients both in FEA and CAD.

Example LR B-spline refinement



Video by Kjetil A. Johannessen, NTNU, Trondheim, Norway.

Frist approach for ensuring linear independence

1. Determine $\dim \mathbb{S}_{\boldsymbol{p}}(\mathcal{M}_{j+1}, \mu_{j+1})$
2. Determine if $\mathcal{B}_{j+1}$ spans $\mathbb{S}_{\boldsymbol{p}}(\mathcal{M}_{j+1}, \mu_{j+1})$
3. Check that $\#\mathcal{B}_{j+1} = \dim \mathbb{S}_{\boldsymbol{p}}(\mathcal{M}_{j+1}, \mu_{j+1})$

The approach is implemented and working well, singles out the linear depended situations efficiently.

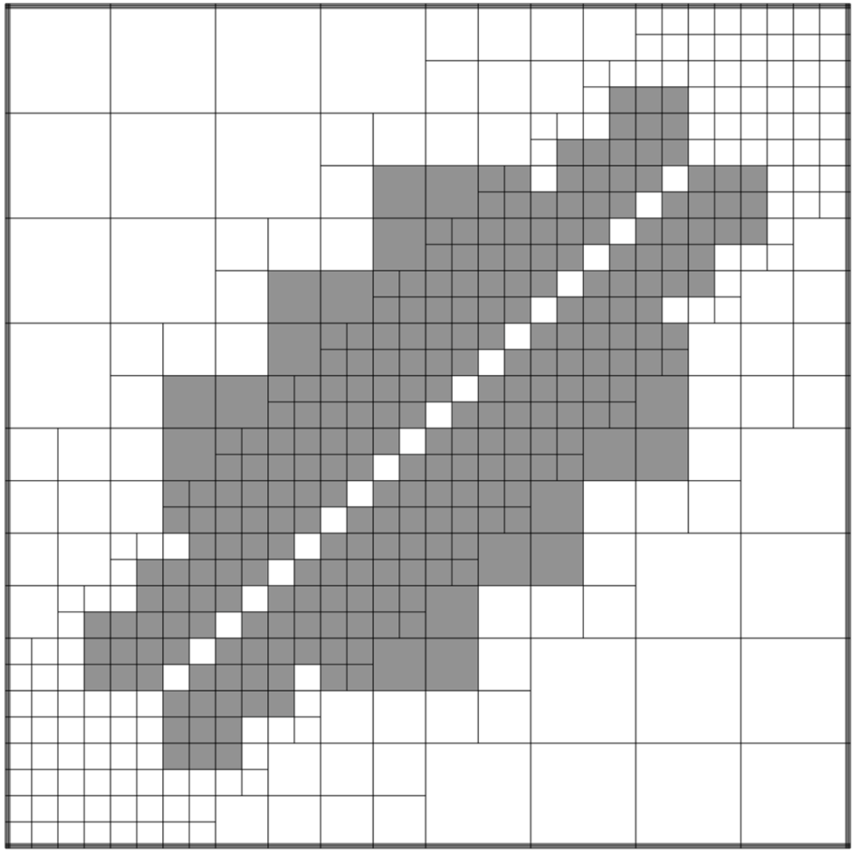
Second Approach for Ensuring Linear Independence

The refinement starts from a tensor product B-spline space with $(p_1 + 1)(p_2 + 1) \dots (p_d + 1)$ B-splines covering an element spanning the polynomial space of degree $(p_1, p_2, \dots, p_d)$ over the element.

- A refinement cannot reduce the polynomial space spanned over an element.
- An extra B-spline in a linear dependency relation can be removed without changing spanning properties over its elements
  - Before the removal of a B-spline there must consequently be more than $(p_1 + 1)(p_2 + 1) \dots (p_d + 1)$ B-splines covering all elements of the removed B-spline.

Overloaded elements and B-splines

- We call an element overloaded if there are more than $(p_1 + 1)(p_2 + 1) \ldots (p_d + 1)$ B-splines covering the element.
- We call a B-spline overloaded if all its elements are overloaded. We denote the collection of overloaded B-splines $\mathcal{B}^{\mathcal{O}}$.



The support of overloaded B-splines colored grey.

Illustration by:
Kjetil A. Johannessen,
SINTEF

Observations

- If there is no overloaded B-spline in the μ-extended mesh then the B-splines are locally (and globally) linearly independent
  - All overloaded elements not part of an overloaded B-spline can be disregarded
- Only overloaded B-splines can occur in linear dependency relations
- A linear dependency relation has to include at least two overloaded B-splines.
  - Elements with only one overloaded B-spline can not be part of linear dependency relation. Thus overloaded B-spline having such an element cannot be part of linear dependency relation.
  - Given a linear dependency relation between the B-splines in a collection of overloaded B-splines $\mathcal{B}^{\mathcal{O}}$. Let $\mathcal{E}^{\mathcal{O}}$ be all the elements of the B-splines in $\mathcal{B}^{\mathcal{O}}$. For every element $e \in \mathcal{E}^{\mathcal{O}}$ there are at least two B-splines from $\mathcal{B}^{\mathcal{O}}$ containing e.

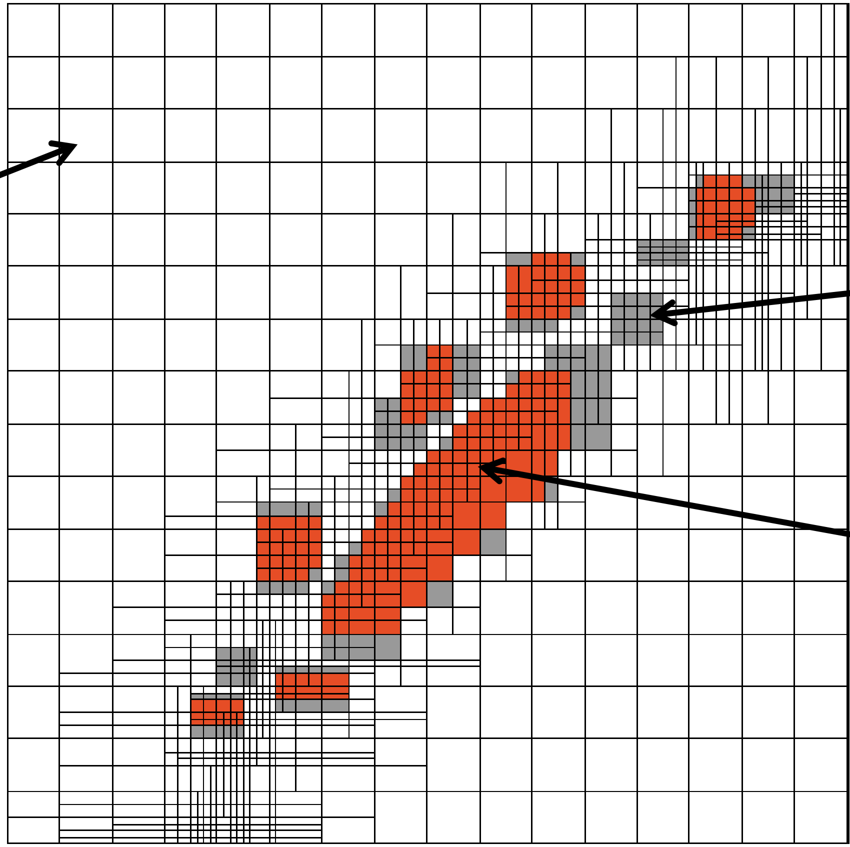
Algorithm

1. From the collection of LR B-splines $\mathcal{B}$ create a collection of overloaded B-splines $\mathcal{B}^{\mathcal{O}}$.
2. Let $\mathcal{E}^{\mathcal{O}}$ be the elements of the B-splines in $\mathcal{B}^{\mathcal{O}}$. For all elements in $\mathcal{E}^{\mathcal{O}}$ identify elements that is covered by only one B-spline from $\mathcal{B}^{\mathcal{O}}$, and collect these B-splines in the collection $\mathcal{B}_1^{\mathcal{O}}$.
3. Remove the B-splines in $\mathcal{B}_1^{\mathcal{O}}$ from $\mathcal{B}^{\mathcal{O}}$: $\mathcal{B}^{\mathcal{O}} := \mathcal{B}^{\mathcal{O}} \backslash \mathcal{B}_1^{\mathcal{O}}$
   - **If** $\mathcal{B}^{\mathcal{O}} \equiv \emptyset$ **then** the B-splines in $\mathcal{B}$ are linearly independent, exit, **else**
   - **If** $\mathcal{B}_1^{\mathcal{O}} \equiv \emptyset$ **then** then the B-splines in $\mathcal{B}^{\mathcal{O}}$ may be part of a linear dependency relation, exit, **else**
   - Try to reduce more, go to 2.

Note: Can both be run in "index space" and with mesh-rectangles with multiplicity.

Example reduction algorithm for overloaded B-splines.

White: Element covered by no B-spline in $\mathcal{B}^{\mathcal{O}}$



Grey: Element covered by only one B-spline in $\mathcal{B}^{\mathcal{O}}$

Red: Element covered by more than one B-spline in $\mathcal{B}^{\mathcal{O}}$

Illustration by:
Kjetil A. Johannessen, NTNU

Example, Continued.

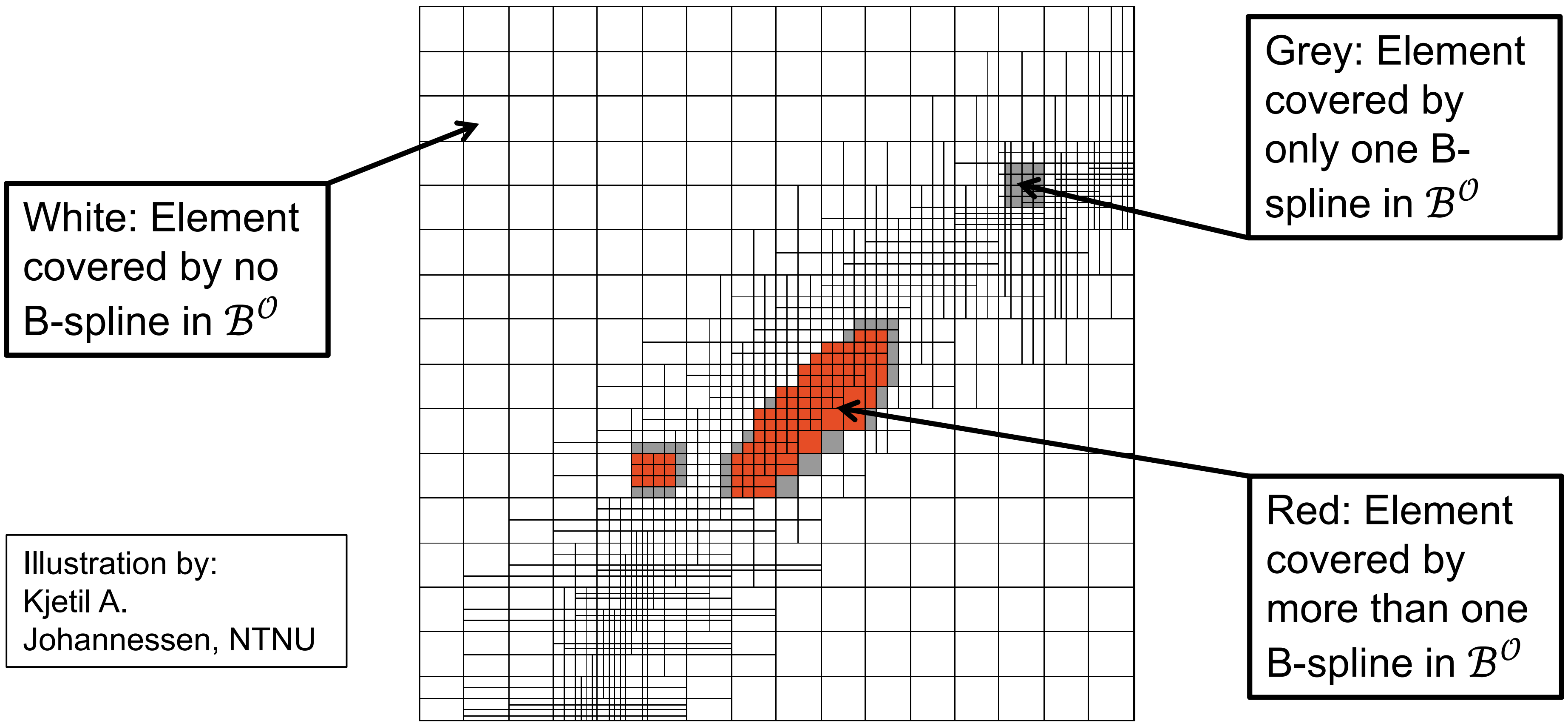
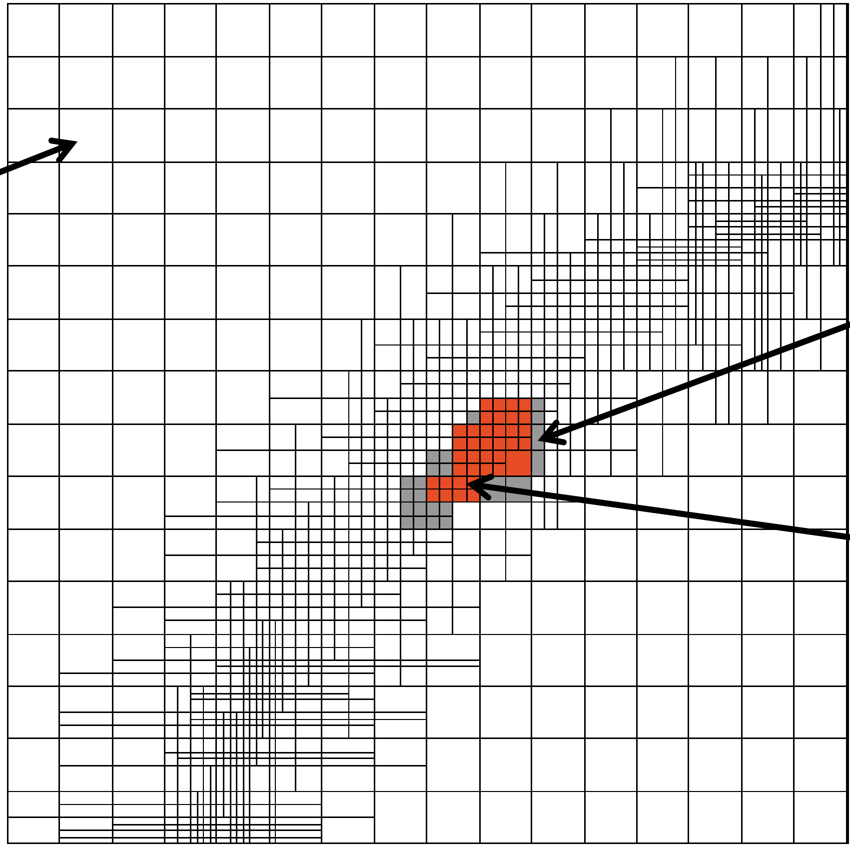


Illustration by:
Kjetil A.
Johannessen, NTNU

Example, Continued.

White: Element covered by no B-spline in $\mathcal{B}^{\mathcal{O}}$



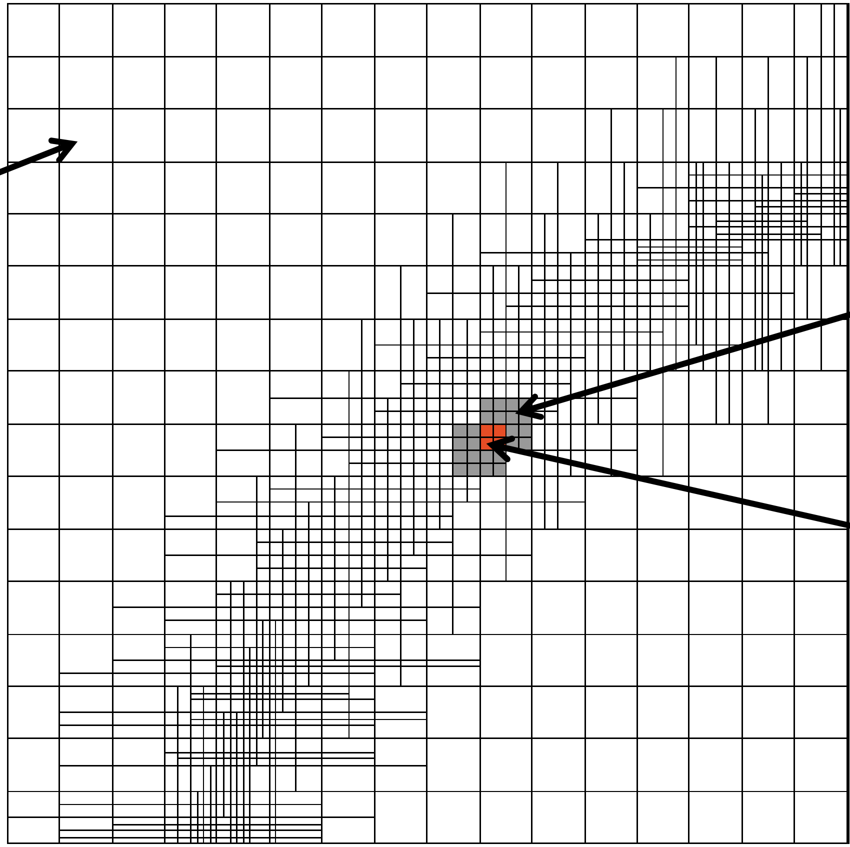
Grey: Element covered by only one B-spline in $\mathcal{B}^{\mathcal{O}}$

Red: Element covered by more than one B-spline in $\mathcal{B}^{\mathcal{O}}$

Illustration by:
Kjetil A. Johannessen, NTNU

All B-splines remaining some elements that are overloaded only once. Linear dependency not possible.

White: Element covered by no B-spline in $\mathcal{B}^{\mathcal{O}}$



Grey: Element covered by only one B-spline in $\mathcal{B}^{\mathcal{O}}$

Red: Element covered by more than one B-spline in $\mathcal{B}^{\mathcal{O}}$

Illustration by:
Kjetil A.
Johannessen, NTNU

No overloaded B-splines remaining. Linear dependency not possible.

White: Element covered by no B-spline in $\mathcal{B}^{\mathcal{O}}$

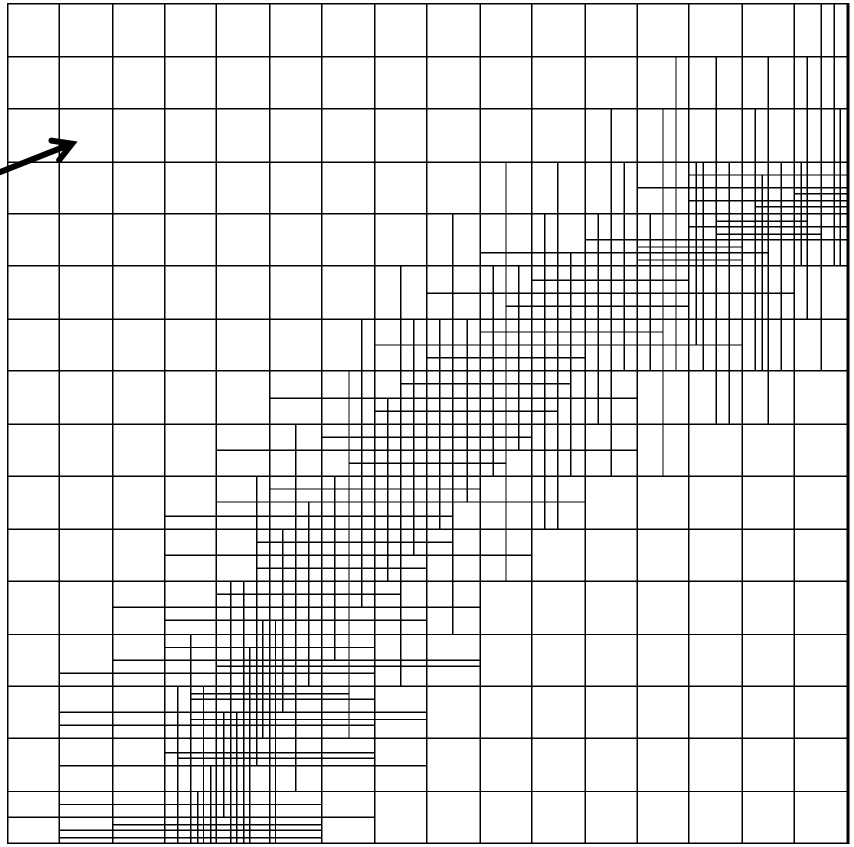


Illustration by: Kjetil A. Johannessen, NTNU

Example, Reduction not successful. Possible linear dependence.

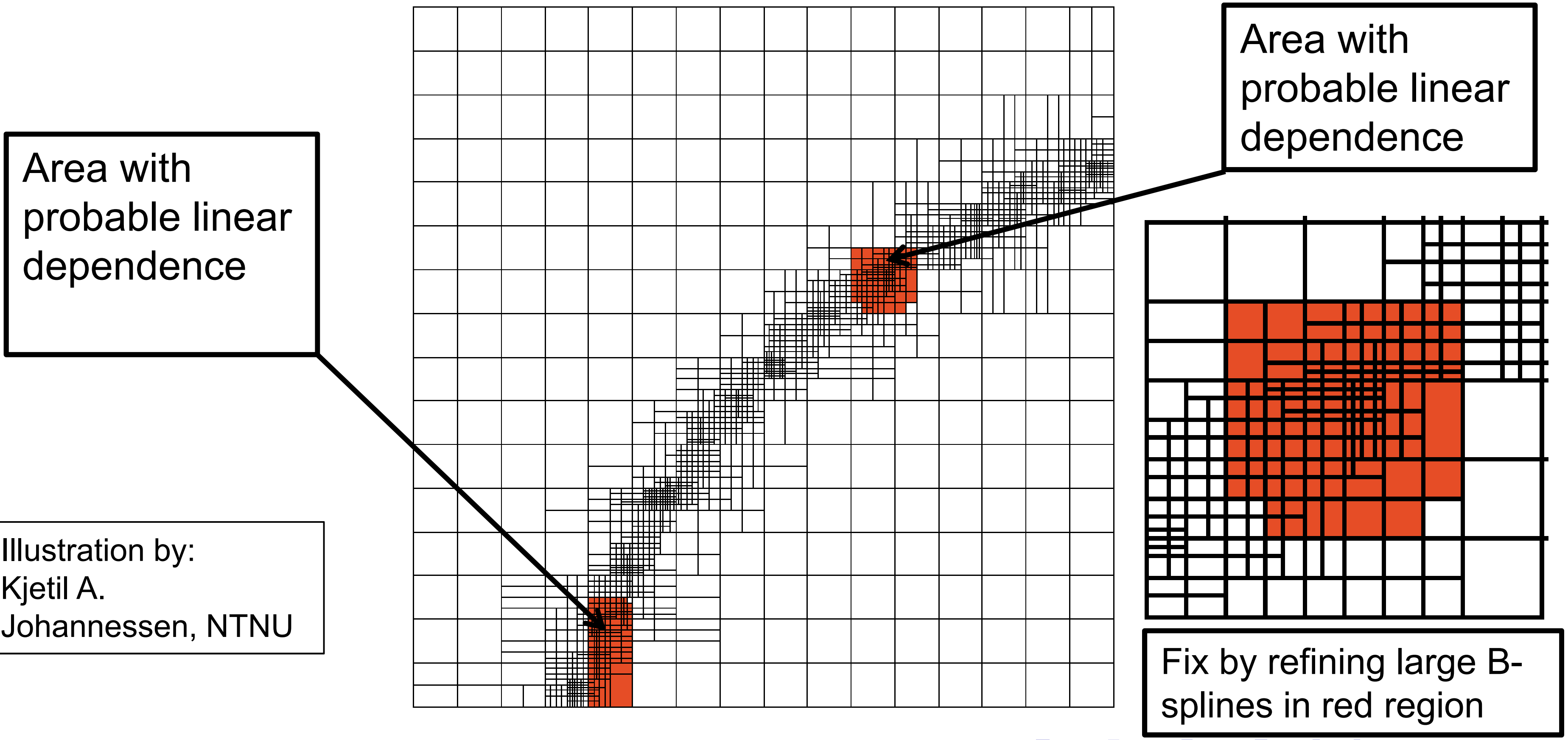


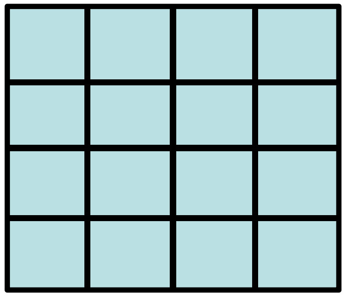
Illustration by:
Kjetil A. Johannessen, NTNU

Alternative uses of the reduction algorithm

- Use overload information to direct refinements
- Prohibit the refinements if $\mathcal{B}^{\mathcal{O}} \neq \emptyset$ after reduction algorithm.
- Check incremental refinement, and perform corrective refinements. In general we expect that the number of remaining overload B-splines is small.
- Check a B-spline collection over an μ-extended mesh for possible linear dependency relations (and perform corrective refinements).

Concluding remarks

- T-splines and LR B-splines are two related approaches to Local Refinement of B-splines
  - LR B-splines refine directly in the μ-extended box-partition
  - T-splines specifies the refinement though the T-spline vertex mesh thus limiting refinement opportunities
- Hierarchical B-splines can as far as we understand be regarded as a special instance of LR B-splines
  - The sets $\Omega_l, l = 1, \ldots$ can be regarded as the sum of the supports of the B-splines being subdivided at a given level
  - The scaled B-splines of LR B-splines can be used as an alternative approach for achieving partition of unity for hierarchical B-splines.



LR B-splines refinement to represent the spline space of truncated hierarchical B-splines
- Will always produce linearly independent B-splines