(Block) ILUT smoothers for *p*-multigrid methods in Isogeometric Analysis

R. Tielen, M. Möller and **C. Vuik**

Delft University of Technology
Delft Institute of Applied Mathematics

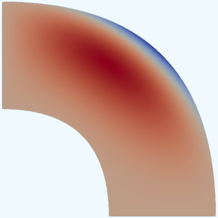
20$^{\mathrm{th}}$ Copper Mountain Conference On Multigrid Methods

March 29 - April 2, 2021

TUDelft

p-Multigrid methods

- Poisson's equation on a quarter annulus
- ILUT and Gauss-Seidel as a smoother

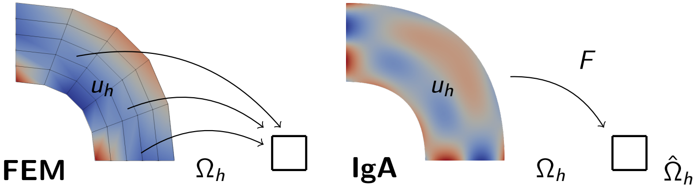


	ILUT	GS
$h = 2^{-6}$	3	491
$h = 2^{-7}$	3	499
$h = 2^{-8}$	3	473
$h = 2^{-9}$	3	452

p-Multigrid cycles for different mesh widths

Isogeometric Analysis (IgA)

- Extension of the Finite Element Method (FEM)
- Geometry Ω and solution u are approximated by same basis functions (**B-Spline basis functions**)
- Global mapping from Ω_h to parametric domain $\hat{\Omega}_h$
- Description of the geometry that is highly accurate (‘$\Omega = \Omega_h$’) throughout all computation steps



TU Delft

Model problem (CDR-equation)

Consider

$$-\nabla \cdot (\mathrm{D}\nabla u) + \mathrm{v} \cdot \nabla u + Ru = f, \quad \text{on } \Omega \quad (1)$$

$$u = g, \quad \text{on } \partial\Omega \quad (2)$$

where D denotes the diffusion tensor, v a divergence-free velocity field and R a source term. Here $\Omega \subset \mathbb{R}^2$ is a connected, Lipschitz domain and $f \in L^2(\Omega)$.

Let $\mathcal{V} = H_0^1(\Omega)$ denote the space of functions in the Sobolev space $H^1(\Omega)$ that vanish at $\partial\Omega$.

TU Delft

Variational formulation

Multiplication of Equation (1) with an arbitrary test function $v \in \mathcal{V}$ and application of integration by parts leads to the variational form:

$$a(u, v) = (f, v) \quad \forall v \in \mathcal{V}, \tag{3}$$

where

$$a(u, v) = \int_\Omega (\mathrm{D}\nabla u) \cdot \nabla v + (\mathrm{v} \cdot \nabla u)v + Ruv \; \mathrm{d}\Omega$$

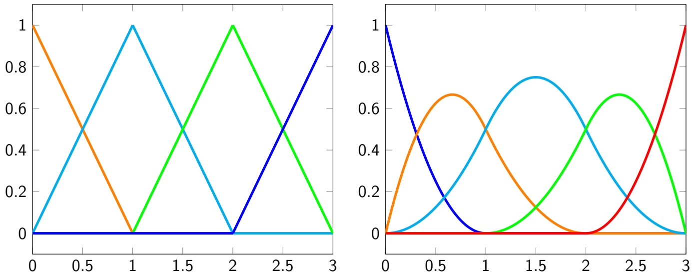
and

$$(f, v) = \int_\Omega fv \; \mathrm{d}\Omega.$$

TU Delft

B-spline basis functions

Isogeometric Analysis adopts **B-spline basis functions** to discretize the variational formulation

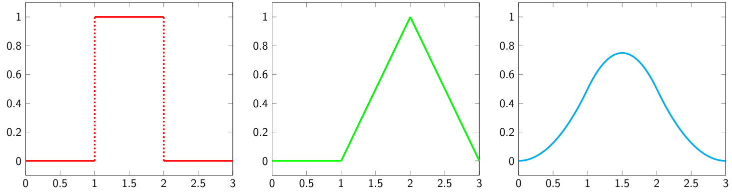


Linear ($p = 1$) and quadratic ($p = 2$) B-spline basis functions

B-spline basis functions

Properties of B-spline basis functions

- Compact support $\Rightarrow$ Sparse system matrices
- Strictly positive $\Rightarrow$ Mass matrix positive
- Partition of unity $\Rightarrow$ Direct mass lumping



Galerkin formulation

Given the spline space $\mathcal{V}_{h,p}$, the Galerkin formulation of (3) becomes:

Find $u_{h,p} \in \mathcal{V}_{h,p}$ such that

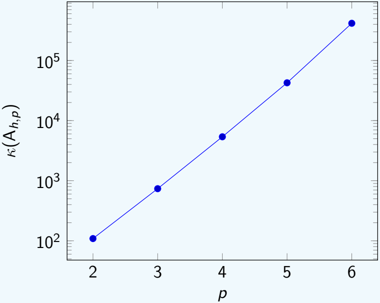
$$a(u_{h,p}, v_{h,p}) = (f, v_{h,p}) \qquad \forall v_{h,p} \in \mathcal{V}_{h,p},$$

where p is the approximation order of the B-splines and h the mesh width. The discretized problem can be written as a linear system

$$\mathrm{A}_{h,p}\mathrm{u}_{h,p} = \mathrm{f}_{h,p}. \tag{4}$$

Need for efficient solvers

For a fixed mesh width h, the condition number $\kappa(\mathrm{A}_{h,p})$ scales exponentially with the approximation order p.



Standard (iterative) solvers become less efficient for higher values of p!

Need for efficient solvers

Enhanced *h*-multigrid methods

- Subspace corrected mass smoother [Takacs, 2017]
- Hybrid smoother [Sogn, 2018]
- Multiplicative Schwarz smoother [de la Riva, 2018]

Preconditioners

- Schwarz methods [Beirão da Veiga, 2012]
- Sylvester equation [Sangalli, 2016]

Our solution strategy:

p-multigrid methods [Tielen et al, 2018 & 2020]

p-multigrid method

Motivation

The linear system $\mathsf{A}_{h,p}\mathsf{u}_{h,p} = \mathsf{f}_{h,p}$

1. becomes more difficult to solve for increasing p
2. reduces to standard C^0-FEM for $p = 1$
   (where h-multigrid is an established solution technique)

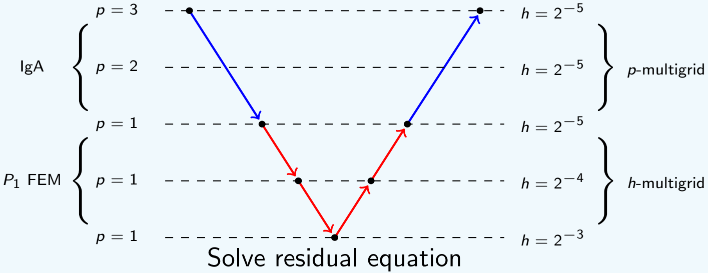
In contrast to h-multigrid methods (in IgA)

- the #DoFs remains similar on coarser p-levels
- the stencil reduces significantly on coarse p-levels
- the spaces are not nested ($\mathcal{V}_{h,p} \not\supset \mathcal{V}_{h,p-1} \not\supset \dots$)

TUDelft

p-multigrid method

- Hierarchy of discretizations with different h and p
- Low-order error is used to update high-order solution
- Smoothing step ($\nu = 1$) is applied at each level (•)
- Gauss-Seidel as smoother on low-order level



Prolongation and restriction

Prolongation in h

$\mathcal{I}_{2h,1}^{h,1}$ is linear interpolation

Restriction in h

$$\mathcal{I}_{h,1}^{2h,1} = \frac{1}{2}\left(\mathcal{I}_{2h,1}^{h,1}\right)^{\mathsf{T}}$$

Prolongation in p

$$\mathcal{I}_{h,1}^{h,p} := (\mathrm{M}_p^p)^{-1}\mathrm{M}_1^p$$

Restriction in p

$$\mathcal{I}_{h,p}^{h,1} := (\mathrm{M}_1^1)^{-1}\mathrm{M}_p^1$$

Let ϕ_i^q denote the i^{th} basis function from $\mathcal{V}_{h,q}$. Then define

$$(\mathrm{M}_q^r)_{(i,j)} := \int_{\hat{\Omega}_h} \phi_i^q(\boldsymbol{\xi})\ \phi_j^r(\boldsymbol{\xi})\ c(\boldsymbol{\xi})\ \mathrm{d}\hat{\Omega}$$

Replace M_q^q by its row-sum lumped counterpart!

Smoother at high order level: Gauss-Seidel?

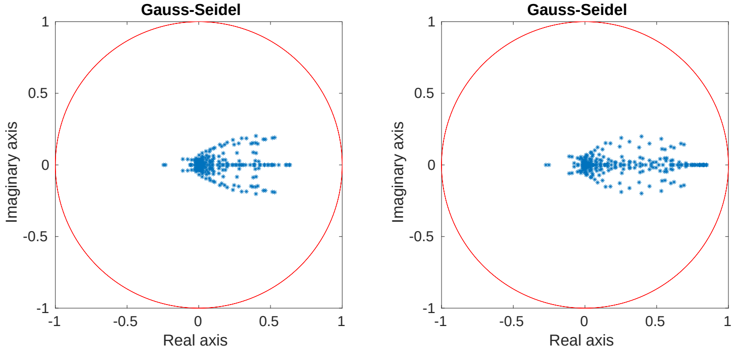


Figure: Spectrum iteration matrix p-multigrid for $p = 2$ (left) and $p = 3$ (right) for Poisson's equation on a quarter annulus.

Alternative: ILUT smoother [Saad 1994]

Setup: Incomplete LU factorization of $A_{h,p} \approx L_{h,p}U_{h,p}$ thereby

1. dropping all elements lower than tolerance $\tau = 10^{-13}$
2. keeping only the N ($=$ average number of non-zero entries in each row of $A_{h,p}$) largest elements in each row

Application: perform $s = 1, \ldots, \nu$ smoothing steps

$$
\begin{aligned}
e_{h,p}^{(s)} &= (L_{h,p}U_{h,p})^{-1}(f_{h,p} - A_{h,p}u_{h,p}^{(s)}) \\
u_{h,p}^{(s+1)} &= u_{h,p}^{(s)} + e_{h,p}^{(s)}
\end{aligned}
$$

TUDelft

Spectrum iteration matrix (p=2)

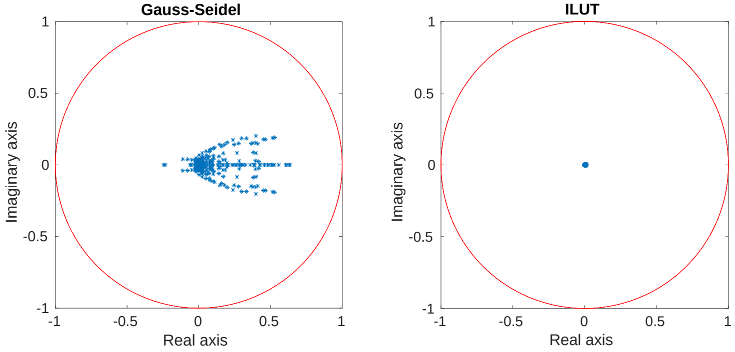


Figure: Spectrum iteration matrix for Gauss-Seidel (left) and ILUT (right) for Poisson's equation on a quarter annulus.

Spectrum iteration matrix (p=3)

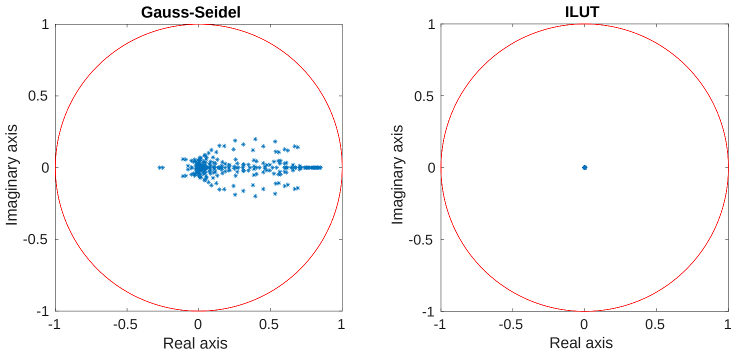


Figure: Spectrum iteration matrix for Gauss-Seidel (left) and ILUT (right) for Poisson's equation on a quarter annulus.

Spectrum iteration matrix (p=4)

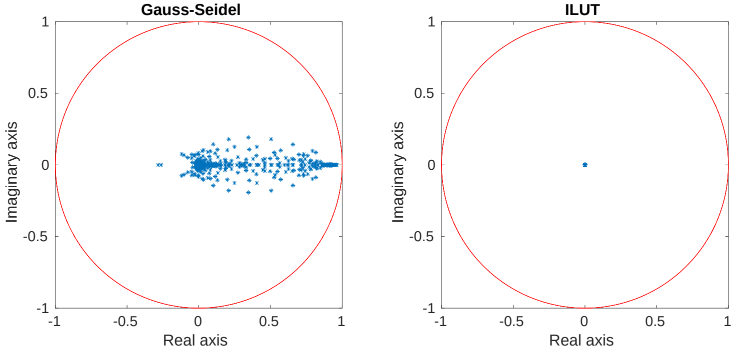


Figure: Spectrum iteration matrix for Gauss-Seidel (left) and ILUT (right) for Poisson's equation on a quarter annulus.

Benchmark #1

- Poisson's equation:

$$\mathrm{D} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad \mathrm{v} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \qquad R = 0.$$

- Ω is a quarter annulus with radia 1 and 2.

	$p=2$		$p=3$		$p=4$		$p=5$	
	ILUT	GS	ILUT	GS	ILUT	GS	ILUT	GS
$h=2^{-6}$	4	30	3	62	3	176	3	491
$h=2^{-7}$	4	29	3	61	3	172	3	499
$h=2^{-8}$	5	30	3	60	3	163	3	473
$h=2^{-9}$	5	32	3	61	3	163	3	452

Benchmark #2

- CDR-equation:

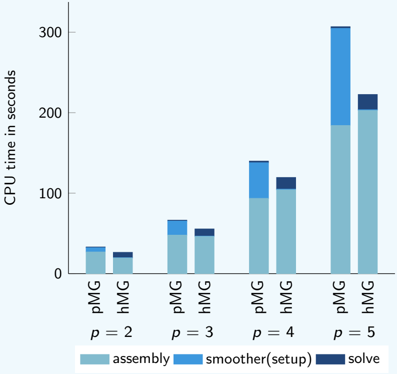
$$D = \begin{bmatrix} 1.2 & -0.7 \\ -0.4 & 0.9 \end{bmatrix}, \quad v = \begin{bmatrix} 0.4 \\ -0.2 \end{bmatrix}, \quad R = 0.3.$$

- Ω is the unit square, i.e. $\Omega = [0,1]^2$.

	$p=2$		$p=3$		$p=4$		$p=5$	
	ILUT	GS	ILUT	GS	ILUT	GS	ILUT	GS
$h = 2^{-6}$	5	–	3	–	3	–	4	–
$h = 2^{-7}$	5	–	3	–	4	–	4	–
$h = 2^{-8}$	5	–	3	–	3	–	4	–
$h = 2^{-9}$	5	–	4	–	3	–	4	–

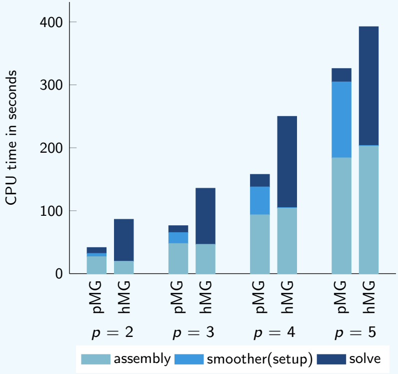
CPU timings (single solve)

- Comparison with *h*-multigrid method [Takacs,2017]
- Higher setup costs with *p*-multigrid, but fast solves!



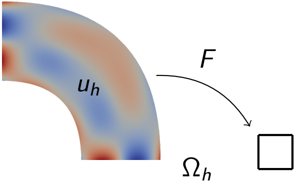
CPU timings (Multiple solves)

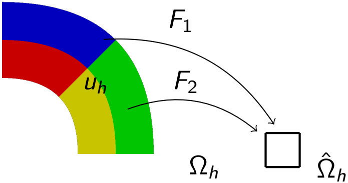
- Typical example: $Au_i = f_i$, $i \in \{1, \ldots, 10\}$
- Increasing influence of solving costs



Multipatch geometries

- Geometry Ω_h can **not always** be described by a single mapping to parametric domain $\hat{\Omega}_h$!
- Represent Ω by non-overlapping subdomains (**patches**), each with their own mapping
- Resulting operator $A_{h,p}$ has a block structure, where each patch leads to a single block





TUDelft

Multipatch geometries

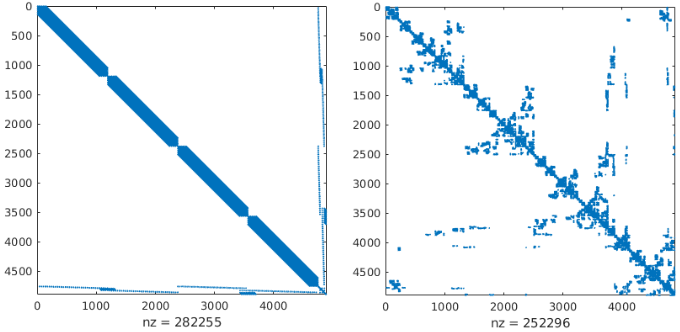


Figure: Sparsity pattern of the system matrix (left) and global ILUT factorization (right) for $p = 3$ and $h = 2^{-5}$ for Poisson's equation on a quarter annulus.

Block ILUT

Observation

We can write $A = LU$ as:

$$\begin{bmatrix} A_{11} & & 0 & A_{\Gamma 1} \\ & \ddots & & \vdots \\ 0 & & A_{KK} & A_{\Gamma K} \\ A_{1\Gamma} & \cdots & A_{K\Gamma} & A_{\Gamma\Gamma} \end{bmatrix} = \begin{bmatrix} L_1 & & & \\ & \ddots & & \\ & & L_K & \\ B_1 & \cdots & B_K & I \end{bmatrix} \begin{bmatrix} U_1 & & & C_1 \\ & \ddots & & \vdots \\ & & U_K & C_K \\ & & & S \end{bmatrix},$$

where

1. $A_{ii} = L_i U_i$
2. $B_i = A_{i\Gamma} U_i^{-1}$
3. $C_i = L_i^{-1} A_{\Gamma i}$
4. $S = A_{\Gamma\Gamma} - \sum_{i=1}^{K} B_i C_i$

Block ILUT

Key Idea (Nievinski 2018)

Replace L and U by their ILUT factorizations:

$$\begin{bmatrix} A_{11} & & & A_{\Gamma 1} \\ & \ddots & & \vdots \\ & & A_{KK} & A_{\Gamma K} \\ A_{1\Gamma} & \cdots & A_{K\Gamma} & A_{\Gamma\Gamma} \end{bmatrix} \approx \begin{bmatrix} \tilde{L}_1 & & & \\ & \ddots & & \\ & & \tilde{L}_K & \\ \tilde{B}_1 & \cdots & \tilde{B}_K & I \end{bmatrix} \begin{bmatrix} \tilde{U}_1 & & & \tilde{C}_1 \\ & \ddots & & \vdots \\ & & \tilde{U}_K & \tilde{C}_K \\ & & & \tilde{S} \end{bmatrix},$$

where

1. $A_{ii} \overset{(!)}{\approx} \tilde{L}_i \tilde{U}_i$
2. $\tilde{B}_i = A_{i\Gamma} \tilde{U}_i^{-1}$
3. $\tilde{C}_i = \tilde{L}_i^{-1} A_{\Gamma i}$
4. $\tilde{S} = A_{\Gamma\Gamma} - \sum_{i=1}^{K} \tilde{B}_i \tilde{C}_i$

Block ILUT

- $\tilde{L}_i, \tilde{U}_i$ can be determined in parallel
- Inversion of $\tilde{L}_i, \tilde{U}_i$ is avoided by solving:

$$\tilde{U}_i^\top \tilde{B}_i^\top = A_{i\Gamma}^\top, \qquad \tilde{L}_i \tilde{C}_i = A_{\Gamma i},$$

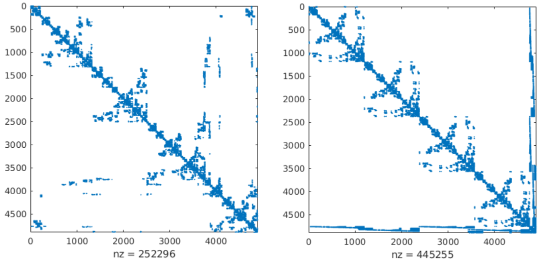
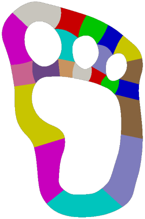


Figure: Global and block ILUT factorization for Poisson's equation

Block ILUT vs. ILUT

Poisson on Yeti Footprint

	$p=2$		$p=3$		$p=4$		$p=5$	
	Global	Block	Global	Block	Global	Block	Global	Block
$h=2^{-3}$	5	4	4	2	4	2	4	2
$h=2^{-4}$	8	4	5	3	5	3	4	2
$h=2^{-5}$	8	4	6	3	5	3	5	3



Conclusions

p-multigrid methods

1. are efficient and robust solvers for IgA
2. enhanced with ILUT as a smoother they are
   - robust in the order p and mesh width h
   - competitive to state-of-the-art *h*-multigrid methods
3. adopting block ILUT has potential (for parallelization) in case of multipatch geometries.

Further reading:
R.Tielen, M. Möller, D. Göddeke and C.Vuik: *p-multigrid methods and their comparison to h-multigrid methods within Isogeometric Analysis*, Comput. Methods Appl. Mech. Engrg., Vol 372 (2020)

TU Delft

Outlook

- Theoretical insight into effectiveness of ILUT
- Further exploration of block ILUT smoother
- Exploit parallelism of block ILUT smoother

G+Smo (Geometry plus Simulation modules),
http://github.com/gismo

Thank you for your attention! Questions?