Robust multigrid methods in isogeometric analysis

Stefan Takacs

Johann Radon Institute for Computational and Applied Mathematics (RICAM)
Austrian Academy of Sciences (ÖAW)
Linz, Austria

AANMPDE 2019

Outline

Model problem: the Poisson problem

Given: domain $\Omega \subset \mathbb{R}^d$ and function $f \in L^2(\Omega)$

Find solution $u \in H^1(\Omega)$ such that

$$\begin{aligned} -\Delta u &= f && \text{in } \Omega \\ u &= 0 && \text{on } \partial\Omega \end{aligned}$$

Galerkin discretization: Find solution $u \in S_{p,h}(\Omega)$ such that

$$(\nabla u, \nabla v) = (f, v) \qquad \text{for all } v \in S_{p,h}(\Omega)$$

Matrix-vector formulation: Find solution $\underline{u}_h$ such that

$$A_h \, \underline{u}_h = \underline{f}_h$$

Model problem: the Poisson problem

Given: domain $\Omega \subset \mathbb{R}^d$ and function $f \in L^2(\Omega)$

Find solution $u \in H^1(\Omega)$ such that

$$\begin{aligned} -\Delta u &= f && \text{in } \Omega \\ u &= 0 && \text{on } \partial\Omega \end{aligned}$$

Galerkin discretization: Find solution $u \in S_{p,h}(\Omega)$ such that

$$(\nabla u, \nabla v) = (f, v) \qquad \text{for all } v \in S_{p,h}(\Omega)$$

Matrix-vector formulation: Find solution $\underline{u}_h$ such that

$$A_h \, \underline{u}_h = \underline{f}_h$$

Single-patch Isogeometric Analysis

- **Spline based FEM with global geometry function**
- Univariate splines $S_{p,k,h}(0,1)$
  - degree p
  - smoothness k ($S_{p,k,h}(0,1) = \{u|_{[ih,(i+1)h)} \in \mathbb{P}_p\} \cap C^k(0,1)$)
  - grid size h
- $S_{p,h} := S_{p,p-1,h}$ are splines of maximum smoothness.
- Tensor-product splines on $\widehat{\Omega} := (0,1)^d$
- Global geometry function $\mathbf{G}$:

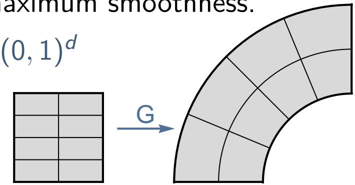
  $\widehat{\Omega} \to \Omega = \mathbf{G}(\widehat{\Omega})$
- Pull-back principle:

  $S_{p,h}(\Omega) = S_{p,h}(\widehat{\Omega}) \circ \mathbf{G}^{-1} = \{u : u \circ \mathbf{G} \in S_{p,h}(\widehat{\Omega})\}$

Single-patch Isogeometric Analysis

- **Spline based FEM with global geometry function**
- Univariate splines $S_{p,k,h}(0,1)$
  - degree p
  - smoothness k ($S_{p,k,h}(0,1) = \{u|_{[ih,(i+1)h)} \in \mathbb{P}_p\} \cap C^k(0,1)$)
  - grid size h
- $S_{p,h} := S_{p,p-1,h}$ are splines of maximum smoothness.
- Tensor-product splines on $\widehat{\Omega} := (0,1)^d$
- Global geometry function $\mathbf{G}$:

  $\widehat{\Omega} \to \Omega = \mathbf{G}(\widehat{\Omega})$



- Pull-back principle:

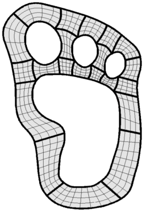
  $S_{p,h}(\Omega) = S_{p,h}(\widehat{\Omega}) \circ \mathbf{G}^{-1} = \{u \,:\, u \circ \mathbf{G} \in S_{p,h}(\widehat{\Omega})\}$

Multi-patch Isogeometric Analysis

- **Spline based FEM with global geometry function**
- Univariate splines $S_{p,k,h}(0,1)$
  - degree p
  - smoothness k ($S_{p,k,h}(0,1) = \{u|_{[ih,(i+1)h)} \in \mathbb{P}_p\} \cap C^k(0,1)$)
  - grid size h
- $S_{p,h} := S_{p,p-1,h}$ are splines of maximum smoothness.
- Tensor-product splines on $\widehat{\Omega} := (0,1)^d$
- **Multi-patch domains:**

  Per-patch geometry functions $\mathbf{G}_k$:

  $\overline{\Omega} = \bigcup_{k=1}^K \overline{\mathbf{G}_k(\widehat{\Omega})}$



- Pull-back principle:

  $S_{p,h}(\Omega) = \{u \,:\, u \circ \mathbf{G}_k \in S_{p,h}(\widehat{\Omega})\ \forall_{k=1,\dots,K}\} \cap C^0(\Omega)$

Model problem: the Poisson problem

Given: domain $\Omega \subset \mathbb{R}^d$ and function $f \in L^2(\Omega)$

Find solution $u \in H^1(\Omega)$ such that

$$\begin{aligned} -\Delta u &= f && \text{in } \Omega \\ u &= 0 && \text{on } \partial\Omega \end{aligned}$$

Galerkin discretization: Find solution $u \in S_{p,h}(\Omega)$ such that

$$(\nabla u, \nabla v) = (f, v) \qquad \text{for all } v \in S_{p,h}(\Omega)$$

Matrix-vector formulation: Find solution $\underline{u}_h$ such that

$$A_h \, \underline{u}_h = \underline{f}_h$$

Why to use IgA?

- **IgA has approximation power of a high-order method:**

$$\inf_{u_h \in S_{p,h}} \|u - u_h\|_{L^2} \lesssim h^{p+1} |u|_{H^{p+1}}$$

- **IgA has problem size of a low-order method:**

$$N := \dim S_{p,h} \eqsim (n+p)^d$$

Problem size of standard high-order FEM: $\dim S_{p,0,h} \eqsim (np)^d$.

- Number of non-zero entries of M_h and A_h grows like $\mathcal{O}(p^d N)$.

Hughes, Cottrell and Bazilevs
Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement.
CMAME, 2005.

Why to use IgA?

- **IgA has approximation power of a high-order method:**

$$\inf_{u_h \in S_{p,h}} \|u - u_h\|_{L^2} \lesssim h^{p+1} |u|_{H^{p+1}}$$

- **IgA has problem size of a low-order method:**

$$N := \dim S_{p,h} \eqsim (n+p)^d$$

Problem size of standard high-order FEM: $\dim S_{p,0,h} \eqsim (np)^d$.

- Number of non-zero entries of M_h and A_h grows like $\mathcal{O}(p^d N)$.

Hughes, Cottrell and Bazilevs
Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement.
CMAME, 2005.

Why to use IgA?

- **IgA has approximation power of a high-order method:**

$$\inf_{u_h \in S_{p,h}} \|u - u_h\|_{L^2} \lesssim h^{p+1} |u|_{H^{p+1}}$$

- **IgA has problem size of a low-order method:**

$$N := \dim S_{p,h} \eqsim (n+p)^d$$

  Problem size of standard high-order FEM: $\dim S_{p,0,h} \eqsim (np)^d$.

- Number of non-zero entries of M_h and A_h grows like $\mathcal{O}(p^d N)$.

Hughes, Cottrell and Bazilevs
Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement.
CMAME, 2005.

Linear solvers

How to solve $\quad A_h \underline{u}_h = \underline{f}_h \quad$?

Note:

$$\kappa(M_h) = \mathcal{O}(2^{pd}), \qquad \kappa(A_h) = \mathcal{O}(h^{-2} 2^{pd}).$$

Multigrid solvers

- **Robustness** in grid size h
- **Robustness** in spline degree p
- **Robustness** in geometry

Multigrid solvers

- **Robustness** in grid size h
- **Robustness** in spline degree p
- **Robustness** in geometry

Multigrid solvers

- **Robustness** in grid size h
- **Robustness** in spline degree p
- **Robustness** in geometry

Multigrid framework

One step of the multigrid method applied to iterate $\underline{u}_h^{(0,0)} = \underline{u}_h^{(0)}$ and right-hand-side $\underline{f}_h$ to obtain $\underline{u}_h^{(1)}$ is given by:

- Apply ν_1 **pre-smoothing steps**

$$\underline{u}_h^{(0,m)} = \underline{u}_h^{(0,m-1)} + \tau L_h^{-1}(\underline{f}_h - A_h \underline{u}_h^{(0,m-1)})$$

  for $m = 1, \ldots, \nu_1$.

- Apply **coarse-grid correction**
  - Compute defect and restrict to coarser grid
  - Solve problem on coarser grid (grid size $H := 2h$)
  - Prolongate and add result

  If realized exactly (two-grid method):

$$\underline{u}_h^{(1)} = \underline{u}_h^{(0,\nu)} + P_H A_H^{-1} P_H^\top (\underline{f}_h - A_h \underline{u}_h^{(0,\nu)})$$

- Apply ν_2 **post-smoothing steps**

Multigrid framework

One step of the multigrid method applied to iterate $\underline{u}_h^{(0,0)} = \underline{u}_h^{(0)}$ and right-hand-side $\underline{f}_h$ to obtain $\underline{u}_h^{(1)}$ is given by:

- Apply ν_1 **pre-smoothing steps**

$$\underline{u}_h^{(0,m)} = \underline{u}_h^{(0,m-1)} + \tau L_h^{-1}(\underline{f}_h - A_h \underline{u}_h^{(0,m-1)})$$

  for $m = 1, \ldots, \nu_1$.
- Apply **coarse-grid correction**
  - Compute defect and restrict to coarser grid
  - Solve problem on coarser grid (grid size $H := 2h$)
  - Prolongate and add result

  If realized exactly (two-grid method):

$$\underline{u}_h^{(1)} = \underline{u}_h^{(0,\nu)} + P_H A_H^{-1} P_H^\top (\underline{f}_h - A_h \underline{u}_h^{(0,\nu)})$$

- Apply ν_2 **post-smoothing steps**

Multigrid framework

One step of the multigrid method applied to iterate $\underline{u}_h^{(0,0)} = \underline{u}_h^{(0)}$ and right-hand-side $\underline{f}_h$ to obtain $\underline{u}_h^{(1)}$ is given by:

- Apply ν_1 **pre-smoothing steps**

$$\underline{u}_h^{(0,m)} = \underline{u}_h^{(0,m-1)} + \tau L_h^{-1}(\underline{f}_h - A_h \underline{u}_h^{(0,m-1)})$$

  for $m = 1, \ldots, \nu_1$.
- Apply **coarse-grid correction**
  - Compute defect and restrict to coarser grid
  - Solve problem on coarser grid (grid size $H := 2h$)
  - Prolongate and add result

  If realized exactly (two-grid method):

$$\underline{u}_h^{(1)} = \underline{u}_h^{(0,\nu)} + P_H A_H^{-1} P_H^\top (\underline{f}_h - A_h \underline{u}_h^{(0,\nu)})$$

- Apply ν_2 **post-smoothing steps**

Multigrid framework

One step of the multigrid method applied to iterate $\underline{u}_h^{(0,0)} = \underline{u}_h^{(0)}$ and right-hand-side $\underline{f}_h$ to obtain $\underline{u}_h^{(1)}$ is given by:

- Apply ν_1 **pre-smoothing steps**

$$\underline{u}_h^{(0,m)} = \underline{u}_h^{(0,m-1)} + \tau L_h^{-1}(\underline{f}_h - A_h \underline{u}_h^{(0,m-1)})$$

  for $m = 1, \ldots, \nu_1$.
- Apply **coarse-grid correction**
  - Compute defect and restrict to coarser grid
  - Solve problem on coarser grid (grid size $H := 2h$)
  - Prolongate and add result

  If realized exactly (two-grid method):

$$\underline{u}_h^{(1)} = \underline{u}_h^{(0,\nu)} + P_H A_H^{-1} P_H^\top (\underline{f}_h - A_h \underline{u}_h^{(0,\nu)})$$

- Apply ν_2 **post-smoothing steps**

Multigrid smoothers

Gauss-Seidel

Gauss-Seidel smoother

- Works well in standard (low-order) finite elements
- Robust convergence (W-cycle) in grid size h:
  - Gahalaut, Kraus, and Tomar
    Multigrid methods for isogeometric discretization.
    CMAME, 2013.
- Not robust in the spline degree p
- Rather robust in geometry

Gauss-Seidel smoother

- Works well in standard (low-order) finite elements
- Robust convergence (W-cycle) in grid size h:

Gahalaut, Kraus, and Tomar
Multigrid methods for isogeometric discretization.
CMAME, 2013.

- Not robust in the spline degree p
- Rather robust in geometry

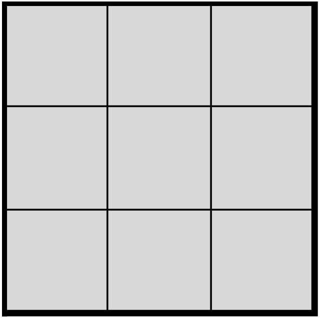
Gauss-Seidel smoother

- Works well in standard (low-order) finite elements
- Robust convergence (W-cycle) in grid size h:
  - Gahalaut, Kraus, and Tomar
    Multigrid methods for isogeometric discretization.
    CMAME, 2013.
- Not robust in the spline degree p
- Rather robust in geometry

Gauss-Seidel smoother

- Works well in standard (low-order) finite elements
- Robust convergence (W-cycle) in grid size h:
  - Gahalaut, Kraus, and Tomar
    Multigrid methods for isogeometric discretization.
    CMAME, 2013.
- Not robust in the spline degree p
- Rather robust in geometry

Unit square

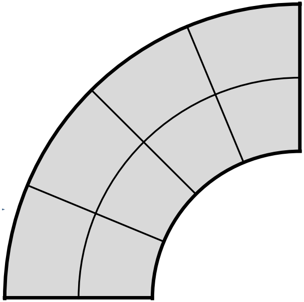


Iteration counts

$\ell \diagdown p$	1	2	3	4	5	6	7
3	8	9	25	53	66	$>$100	$>$100
4	8	9	24	75	$>$100	$>$100	$>$100
5	8	9	23	73	$>$100	$>$100	$>$100
6	8	9	24	73	$>$100	$>$100	$>$100
7	8	9	24	70	$>$100	$>$100	$>$100

V-cycle, $\epsilon = 10^{-8}$

Quarter annulus

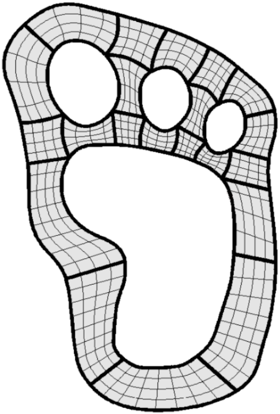


Iteration counts

$\ell \diagdown p$	1	2	3	4	5	6	7
3	12	10	26	48	>100	>100	>100
4	14	11	24	75	>100	>100	>100
5	16	13	23	61	>100	>100	>100
6	18	14	23	63	>100	>100	>100
7	19	15	24	68	>100	>100	>100

V-cycle, $\epsilon = 10^{-8}$

Yeti footprint



Iteration counts

$\ell \diagdown p$	1	2	3	4	5	6	7
2	12	11	26	82	$>$100	$>$100	$>$100
3	15	13	25	75	$>$100	$>$100	$>$100
4	16	14	25	74	$>$100	$>$100	$>$100
5	18	15	25	74	$>$100	$>$100	$>$100

V-cycle, $\epsilon = 10^{-8}$

Computational complexity

- The cost for applying the smoother is linear in the number of non-zeros of A_h, thus each smoothing step costs

$$\mathcal{O}(p^d N) \quad \text{flops.}$$

- Computational costs for one multigrid cycle are asymptotically the same.

Multigrid smoothers

Subspace corrected mass smoother

Subspace corrected mass smoother

T. and Takacs.
Approximation error estimates and inverse inequalities for B-splines of maximum smoothness. *M^3AS*, 2016.

The space
$V_0 := \{u \in S_{p,h}(0,1) : u^{(i)}(0) = u^{(i)}(1) = 0 \ \forall_{i=1,3,\ldots,2\lfloor p/2\rfloor-1}\}$
satisfies both

- a **robust inverse estimate**

$$|u_0|_{H^1(0,1)} \le 2\sqrt{3}h^{-1}\|u_0\|_{L_2(0,1)} \quad \text{for} \quad u_0 \in V_0$$

- a **robust approximation error estimate**

$$\inf_{u_0 \in V_0} \|u - u_0\|_{L_2(0,1)} \le \sqrt{2}h|u|_{H^1(0,1)}$$

Subspace corrected mass smoother

Hofreither and T.
Robust Multigrid for Isogeometric Analysis using Subspace Correction. *SINUM. 55 (4). p. 2004 - 2024*, 2017.

The L_2-orthogonal splitting of $V := S_{p,h}$ into V_0 and its complement V_1 is H^1-stable

Tensor-product structure (for unit square):

$$A_h = K \otimes M + M \otimes K$$
$$\approx \sum_{(\alpha,\beta)\in\{0,1\}^2} (\Pi_\alpha \otimes \Pi_\beta)(K_\alpha \otimes M_\beta + M_\alpha \otimes K_\beta)(\Pi_\alpha \otimes \Pi_\beta)^\top$$

Π_α is L_2-projection $V \to V_\alpha$

Subspace corrected mass smoother

Hofreither and T.
Robust Multigrid for Isogeometric Analysis using Subspace Correction. *SINUM. 55 (4). p. 2004 - 2024*, 2017.

The L_2-orthogonal splitting of $V := S_{p,h}$ into V_0 and its complement V_1 is H^1-stable

Tensor-product structure (for unit square):

$$A_h^{-1} \eqsim \sum_{(\alpha,\beta)\in\{0,1\}^2} (P_\alpha \otimes P_\beta)(K_\alpha \otimes M_\beta + M_\alpha \otimes K_\beta)^{-1}(P_\alpha \otimes P_\beta)^\top$$

P_α is embedding $V_\alpha \to V$

Subspace corrected mass smoother

Hofreither and T.
Robust Multigrid for Isogeometric Analysis using Subspace Correction. *SINUM. 55 (4). p. 2004 - 2024*, 2017.

The L_2-orthogonal splitting of $V := S_{p,h}$ into V_0 and its complement V_1 is H^1-stable

Tensor-product structure (for unit square):

$$\begin{aligned} A_h^{-1} \gtrsim\, & (P_0 \otimes P_0)(h^{-2} M_0^{-1} \otimes M_0^{-1})(P_0 \otimes P_0)^\top \\ & + (P_1 \otimes P_0)((K_1 + h^{-2} M_1)^{-1} \otimes M_0^{-1})(P_1 \otimes P_0)^\top \\ & + (P_0 \otimes P_1)(M_0^{-1} \otimes (K_1 + h^{-2} M_1)^{-1})(P_0 \otimes P_1)^\top \\ & + (P_1 \otimes P_1)(K_1 \otimes M_1 + M_1 \otimes K_1)^{-1}(P_1 \otimes P_1)^\top =: L_h^{-1} \end{aligned}$$

using $K_0 \lesssim h^{-2} M_0$

Convergence theory

Can show

$$L_h \approx A_h + h^{-2} M_h$$

Theorem

If sufficiently many smoothing steps are applied (independent of grid size and spline degree), the W-cycle multigrid solver converges robustly.

Hofreither and T.
Robust Multigrid for Isogeometric Analysis using Subspace Correction. *SINUM. 55 (4). p. 2004 - 2024*, 2017.

Computational complexity

- The setup of the smoother costs

$$\mathcal{O}(pN + p^{3d}) \quad \text{flops}$$

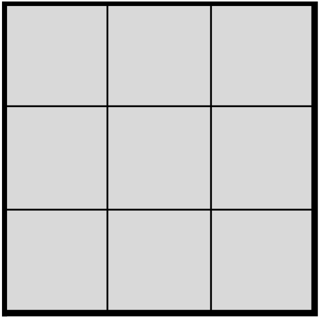
and for applying the smoother costs

$$\mathcal{O}(pN + p^{2d}) \quad \text{flops}$$

per smoothing step.
- The computation of the residual costs $\mathcal{O}(\text{nnz } A_h) \approx \mathcal{O}(p^d N)$ flops.
- The overall cost for one multigrid cycle is

$$\mathcal{O}(p^d N + p^{2d} \log N) \quad \text{flops.}$$

Unit square



Iteration counts

$\ell \setminus p$	1	2	3	4	5	6	7
3	23	19	16	12	10	8	6
4	26	26	23	20	19	16	14
5	26	29	28	26	25	23	22
6	27	30	29	28	27	26	26
7	27	31	30	28	28	27	27

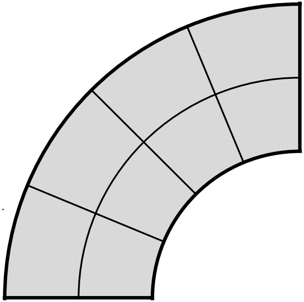
V-cycle, $2+2$ smoothing steps, $\epsilon = 10^{-8}$

Iteration counts

$\ell \setminus p$	1	2	3	4	5	6	7
3	14	12	10	8	7	7	6
4	15	15	14	13	12	11	10
5	16	16	16	15	14	14	13
6	16	17	16	16	15	15	15
7	16	17	17	16	16	16	15

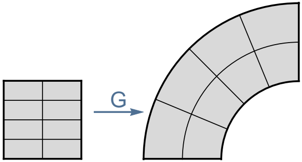
V-cycle, PCG, $\epsilon = 10^{-8}$

Quarter annulus



Quarter annulus

Remember **pull-back principle:**



Substitution rule yields

$$A_h \eqsim \widehat{A}_h,$$

which is **robust** in grid size h and spline degree p, but **heavily depending** on geometry function G.

Iteration counts

$\ell \diagdown p$	1	2	3	4	5	6	7
3	21	18	16	15	18	23	32
4	26	26	23	22	24	47	47
5	29	30	28	27	30	47	47
6	31	32	31	30	36	47	47
7	32	34	33	32	41	47	47

V-cycle, PCG, $\epsilon = 10^{-8}$

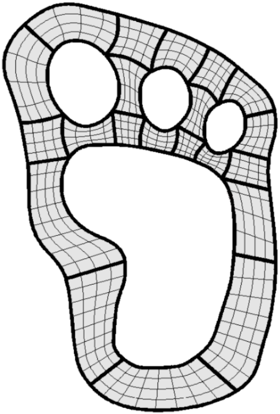
Convergence theory

Theorem

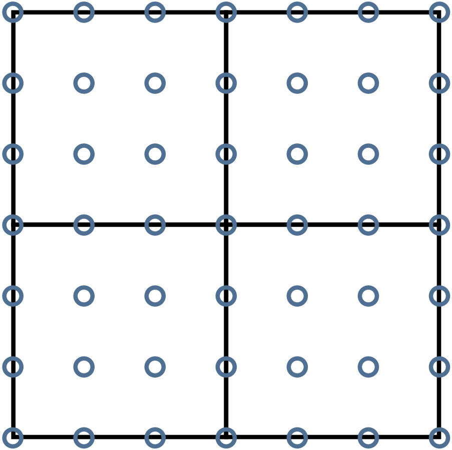
If sufficiently many smoothing steps are applied (independent of grid size and spline degree ***but depending on the geometry function****), the W-cycle multigrid solver converges robustly.*

Hofreither and T.
Robust Multigrid for Isogeometric Analysis using Subspace Correction. *SINUM. 55 (4). p. 2004 - 2024*, 2017.

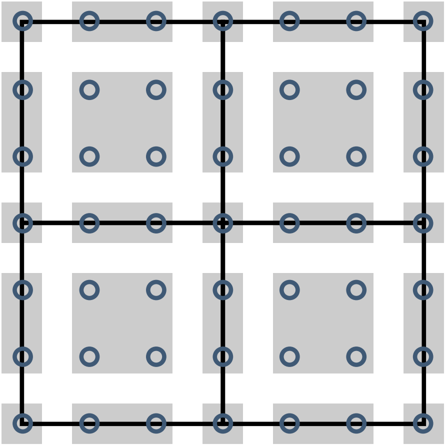
Yeti footprint



Decomposition of the degrees of freedom



Decomposition of the degrees of freedom



Extension to multi-patch case

- On the patch-interior, have tensor-product structure:
  subspace corrected mass smoother
- The problems on edges, vertices are small: can use a direct solver
- Additive Schwarz type combination

Extension to multi-patch case

- On the patch-interior, have tensor-product structure:
  subspace corrected mass smoother
- The problems on edges, vertices are small: can use a direct solver
- Additive Schwarz type combination

Extension to multi-patch case

- On the patch-interior, have tensor-product structure: **subspace corrected mass smoother**
- The problems on edges, vertices are small: can use a direct solver
- Additive Schwarz type combination

Convergence theory

The splitting between the subspaces is almost stable:

$$A_h + h^{-2} M_h \lesssim \sum_T P_T (A_T + h^{-2} M_T) P_T^\top \lesssim p (A_h + h^{-2} M_h)$$

Theorem

If $\mathcal{O}(p)$ smoothing steps are applied (independent of grid size ***but depending on the geometry function****), the W-cycle multigrid solver converges robustly.*

T.
Robust approximation error estimates and multigrid solvers for isogeometric multi-patch discretizations. *M^3AS*, 2018.

Computational complexity

- Applying the smoother costs

  $$\mathcal{O}(pN + p^{2d}) \quad \text{flops}$$

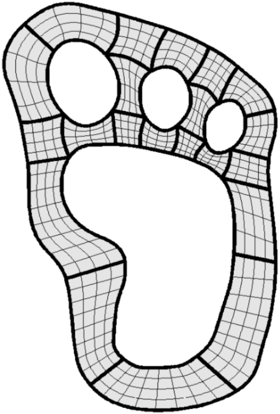
  per smoothing step.
- The computation of the residual is $\mathcal{O}(\text{nnz } A_h) \approx \mathcal{O}(p^d N)$ flops.
- The overall cost for one multigrid cycle is

  $$\mathcal{O}(p^d N + p^{2d} \log N) \quad \text{flops}$$

  or, if $\mathcal{O}(p)$ smoothing steps are applied,

  $$\mathcal{O}(p^{d+1} N + p^{2d+1} \log N) \quad \text{flops.}$$

Yeti footprint



Iteration counts

$\ell \setminus p$	1	2	3	4	5	6	7
2	30	31	29	27	25	24	22
3	36	36	35	34	32	31	30
4	38	39	38	37	35	35	33
5	40	42	40	39	38	37	36

V-cycle, PCG, $\epsilon = 10^{-8}$

Multigrid smoothers

Macro-element Gauss-Seidel

Macro-element Gauss-Seidel smoother

Gauss-Seidel:

$$\underline{u}_h^{(new)} = \underline{u}_h - P_i A_i^{-1} P_i^\top (A_h \underline{u}_h - \underline{f}_h),$$

where $A_i := P_i^\top A P_i$ and $P_i = (\underbrace{0, \ldots, 0}_{i-1}, 1, \underbrace{0, \ldots, 0}_{N-1-i})^\top$.

Macro-element Gauss-Seidel: Include $p-1$ neighbors in each direction

Beirão da Veiga, Cho, Pavarino, and Scacchi
Overlapping Schwarz methods for Isogeometric Analysis.
SINUM, 2012.

Macro-element Gauss-Seidel smoother

Gauss-Seidel:

$$\underline{u}_h^{(new)} = \underline{u}_h - P_i A_i^{-1} P_i^\top (A_h \underline{u}_h - \underline{f}_h),$$

where $A_i := P_i^\top A P_i$ and $P_i = (\underbrace{0,\ldots,0}_{i-1},1,\underbrace{0,\ldots,0}_{N-1-i})^\top$.

Macro-element Gauss-Seidel: Include $p-1$ neighbors in each direction

Beirão da Veiga, Cho, Pavarino, and Scacchi
Overlapping Schwarz methods for Isogeometric Analysis.
SINUM, 2012.

Macro-element Gauss-Seidel smoother

Gauss-Seidel:

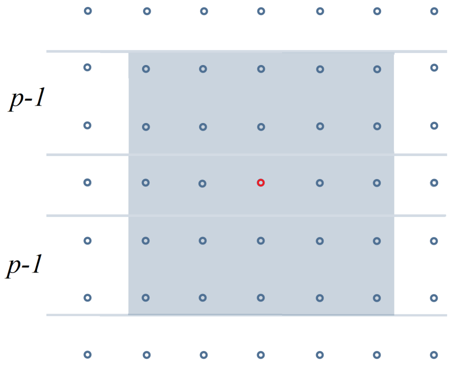
$$\underline{u}_h^{(new)} = \underline{u}_h - P_i A_i^{-1} P_i^\top (A_h \underline{u}_h - \underline{f}_h),$$

where $A_i := P_i^\top A P_i$ and $P_i = (\underbrace{0,\ldots,0}_{i-1},1,\underbrace{0,\ldots,0}_{N-1-i})^\top$.

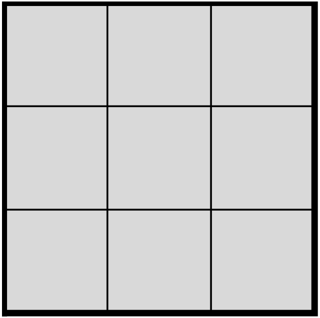
Macro-element Gauss-Seidel: Include $p-1$ neighbors in each direction

Beirão da Veiga, Cho, Pavarino, and Scacchi
Overlapping Schwarz methods for Isogeometric Analysis.
SINUM, 2012.

Macro-element Gauss-Seidel smoother



Unit square

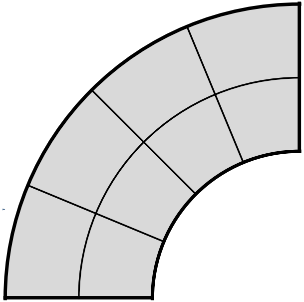


Iteration counts

$\ell \setminus p$	1	2	3	4	5	6	7
3	8	3	3	3	2	2	1
4	8	4	3	3	2	2	2
5	8	4	3	3	3	2	2
6	8	4	3	3	3	3	2
7	8	4	4	3	3	3	3

V-cycle, $\epsilon = 10^{-8}$

Quarter annulus

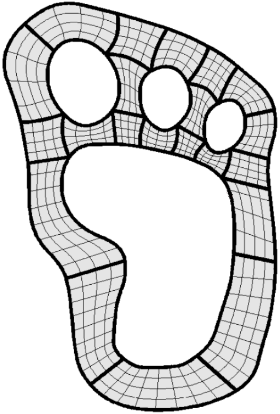


Iteration counts

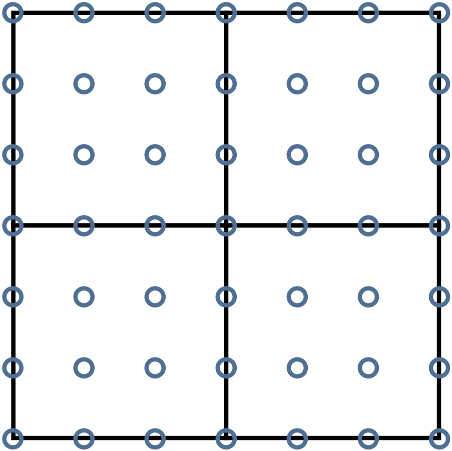
$\ell \diagdown p$	1	2	3	4	5	6	7
3	12	4	3	2	2	2	1
4	14	5	3	3	3	2	2
5	16	5	4	3	3	3	3
6	18	5	4	3	3	3	3
7	19	5	4	3	3	3	3

V-cycle, $\epsilon = 10^{-8}$

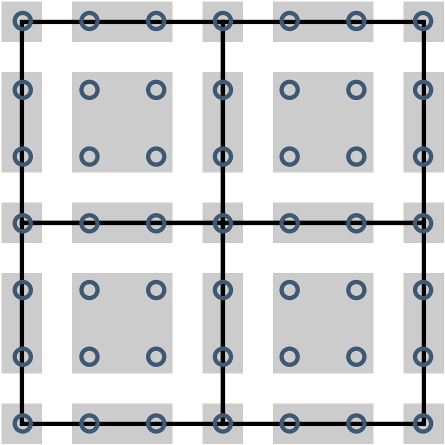
Yeti footprint



Decomposition of the degrees of freedom



Decomposition of the degrees of freedom



Iteration counts

$\ell \setminus p$	1	2	3	4	5	6	7
2	12	9	10	11	11	11	11
3	16	10	12	13	15	18	17
4	16	11	14	16	18	21	20

V-cycle, $\epsilon = 10^{-8}$

Macro-element Gauss-Seidel smoother

- So far, no complete convergence analysis (showing robustness)

- Robust convergence in grid size h, cf.

Gahalaut, Kraus, and Tomar
Multigrid methods for isogeometric discretization.
CMAME, 2013.

Macro-element Gauss-Seidel smoother

- So far, no complete convergence analysis (showing robustness)

- Robust convergence in grid size h, cf.

Gahalaut, Kraus, and Tomar
Multigrid methods for isogeometric discretization.
CMAME, 2013.

Computational complexity

- Each macro-element has $(2p-1)^d$ degrees of freedom
- Setup of patch-local solver costs $\mathcal{O}(p^{3d})$ flops
- Application of patch-local solver costs $\mathcal{O}(p^{2d})$ flops
- Update of residual costs $\mathcal{O}(p^{2d})$ flops
- Total costs: $\mathcal{O}(p^{3d}N)$ (application: $\mathcal{O}(p^{2d}N)$)
- Can we improve?

Computational complexity

- Each macro-element has $(2p-1)^d$ degrees of freedom
- Setup of patch-local solver costs $\mathcal{O}(p^{3d})$ flops
- Application of patch-local solver costs $\mathcal{O}(p^{2d})$ flops
- Update of residual costs $\mathcal{O}(p^{2d})$ flops
- Total costs: $\mathcal{O}(p^{3d}N)$ (application: $\mathcal{O}(p^{2d}N)$)
- Can we improve?

Computational complexity

- Each macro-element has $(2p-1)^d$ degrees of freedom
- Setup of patch-local solver costs $\mathcal{O}(p^{3d})$ flops
- Application of patch-local solver costs $\mathcal{O}(p^{2d})$ flops
- Update of residual costs $\mathcal{O}(p^{2d})$ flops
- Total costs: $\mathcal{O}(p^{3d}N)$ (application: $\mathcal{O}(p^{2d}N)$)
- Can we improve?

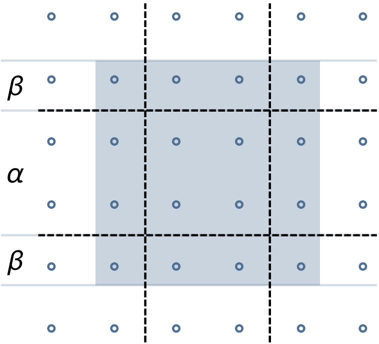
Computational complexity

- Each macro-element has $(2p-1)^d$ degrees of freedom
- Setup of patch-local solver costs $\mathcal{O}(p^{3d})$ flops
- Application of patch-local solver costs $\mathcal{O}(p^{2d})$ flops
- Update of residual costs $\mathcal{O}(p^{2d})$ flops
- Total costs: $\mathcal{O}(p^{3d}N)$ (application: $\mathcal{O}(p^{2d}N)$)
- Can we improve?

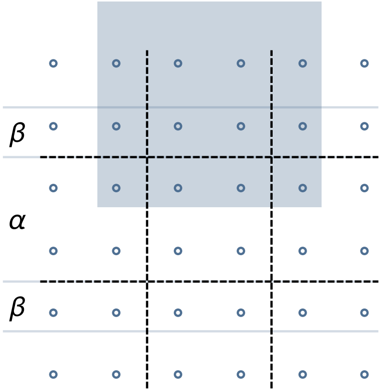
Computational complexity

- Each macro-element has $(2p-1)^d$ degrees of freedom
- Setup of patch-local solver costs $\mathcal{O}(p^{3d})$ flops
- Application of patch-local solver costs $\mathcal{O}(p^{2d})$ flops
- Update of residual costs $\mathcal{O}(p^{2d})$ flops
- Total costs: $\mathcal{O}(p^{3d}N)$ (application: $\mathcal{O}(p^{2d}N)$)
- Can we improve?

Macro-element Gauss-Seidel smoother



Macro-element Gauss-Seidel smoother



Computational complexity

- Each macro-element has $(\alpha + 2\beta)^d$ degrees of freedom
- Setup of patch-local solver costs $\mathcal{O}((\alpha + 2\beta)^{3d})$ flops
- Application of patch-local solver costs $\mathcal{O}((\alpha + 2\beta)^{2d})$ flops
- Update of residual costs $\mathcal{O}((\alpha + 2\beta)^d p^d)$ flops
- Number of macro-elements is $\eqsim N/\alpha^d$
- Total costs: $\mathcal{O}((1 + \alpha^{-1}\beta)^d((\alpha + \beta)^{2d} + p^d)N)$
  For $\beta \eqsim p$: $\mathcal{O}((1 + \alpha^{-1}p)^d(\alpha + p)^{2d}N)$
  For $\alpha, \beta \eqsim p$: $\mathcal{O}(p^{2d}N)$ (application: $\mathcal{O}(p^dN)$)

Computational complexity

- Each macro-element has $(\alpha + 2\beta)^d$ degrees of freedom
- Setup of patch-local solver costs $\mathcal{O}((\alpha + 2\beta)^{3d})$ flops
- Application of patch-local solver costs $\mathcal{O}((\alpha + 2\beta)^{2d})$ flops
- Update of residual costs $\mathcal{O}((\alpha + 2\beta)^d p^d)$ flops
- Number of macro-elements is $\eqsim N/\alpha^d$
- Total costs: $\mathcal{O}((1 + \alpha^{-1}\beta)^d((\alpha + \beta)^{2d} + p^d)N)$
  For $\beta \eqsim p$: $\mathcal{O}((1 + \alpha^{-1}p)^d(\alpha + p)^{2d}N)$
  For $\alpha, \beta \eqsim p$: $\mathcal{O}(p^{2d}N)$ (application: $\mathcal{O}(p^d N)$)

Computational complexity

- Each macro-element has $(\alpha + 2\beta)^d$ degrees of freedom
- Setup of patch-local solver costs $\mathcal{O}((\alpha + 2\beta)^{3d})$ flops
- Application of patch-local solver costs $\mathcal{O}((\alpha + 2\beta)^{2d})$ flops
- Update of residual costs $\mathcal{O}((\alpha + 2\beta)^d p^d)$ flops
- Number of macro-elements is $\eqsim N/\alpha^d$
- Total costs: $\mathcal{O}((1 + \alpha^{-1}\beta)^d((\alpha + \beta)^{2d} + p^d)N)$
  For $\beta \eqsim p$: $\mathcal{O}((1 + \alpha^{-1}p)^d(\alpha + p)^{2d}N)$
  For $\alpha, \beta \eqsim p$: $\mathcal{O}(p^{2d}N)$ (application: $\mathcal{O}(p^d N)$)

Computational complexity

- Each macro-element has $(\alpha + 2\beta)^d$ degrees of freedom
- Setup of patch-local solver costs $\mathcal{O}((\alpha + 2\beta)^{3d})$ flops
- Application of patch-local solver costs $\mathcal{O}((\alpha + 2\beta)^{2d})$ flops
- Update of residual costs $\mathcal{O}((\alpha + 2\beta)^d p^d)$ flops
- Number of macro-elements is $\eqsim N/\alpha^d$
- Total costs: $\mathcal{O}((1 + \alpha^{-1}\beta)^d((\alpha + \beta)^{2d} + p^d)N)$
  For $\beta \eqsim p$: $\mathcal{O}((1 + \alpha^{-1}p)^d(\alpha + p)^{2d}N)$
  For $\alpha, \beta \eqsim p$: $\mathcal{O}(p^{2d}N)$ (application: $\mathcal{O}(p^d N)$)

Computational complexity

- Each macro-element has $(\alpha + 2\beta)^d$ degrees of freedom
- Setup of patch-local solver costs $\mathcal{O}((\alpha + 2\beta)^{3d})$ flops
- Application of patch-local solver costs $\mathcal{O}((\alpha + 2\beta)^{2d})$ flops
- Update of residual costs $\mathcal{O}((\alpha + 2\beta)^d p^d)$ flops
- Number of macro-elements is $\eqsim N/\alpha^d$
- Total costs: $\mathcal{O}((1 + \alpha^{-1}\beta)^d((\alpha + \beta)^{2d} + p^d)N)$
  For $\beta \eqsim p$: $\mathcal{O}((1 + \alpha^{-1}p)^d(\alpha + p)^{2d}N)$
  For $\alpha, \beta \eqsim p$: $\mathcal{O}(p^{2d}N)$ (application: $\mathcal{O}(p^d N)$)

Computational complexity

- Each macro-element has $(\alpha + 2\beta)^d$ degrees of freedom
- Setup of patch-local solver costs $\mathcal{O}((\alpha + 2\beta)^{3d})$ flops
- Application of patch-local solver costs $\mathcal{O}((\alpha + 2\beta)^{2d})$ flops
- Update of residual costs $\mathcal{O}((\alpha + 2\beta)^d p^d)$ flops
- Number of macro-elements is $\eqsim N/\alpha^d$
- Total costs: $\mathcal{O}((1 + \alpha^{-1}\beta)^d((\alpha + \beta)^{2d} + p^d)N)$
  For $\beta \eqsim p$: $\mathcal{O}((1 + \alpha^{-1}p)^d(\alpha + p)^{2d}N)$
  For $\alpha, \beta \eqsim p$: $\mathcal{O}(p^{2d}N)$ (application: $\mathcal{O}(p^d N)$)

Computational complexity

- Each macro-element has $(\alpha + 2\beta)^d$ degrees of freedom
- Setup of patch-local solver costs $\mathcal{O}((\alpha + 2\beta)^{3d})$ flops
- Application of patch-local solver costs $\mathcal{O}((\alpha + 2\beta)^{2d})$ flops
- Update of residual costs $\mathcal{O}((\alpha + 2\beta)^d p^d)$ flops
- Number of macro-elements is $\eqsim N/\alpha^d$
- Total costs: $\mathcal{O}((1 + \alpha^{-1}\beta)^d((\alpha + \beta)^{2d} + p^d)N)$
  For $\beta \eqsim p$: $\mathcal{O}((1 + \alpha^{-1}p)^d(\alpha + p)^{2d}N)$
  For $\alpha, \beta \eqsim p$: $\mathcal{O}(p^{2d}N)$ (application: $\mathcal{O}(p^d N)$)

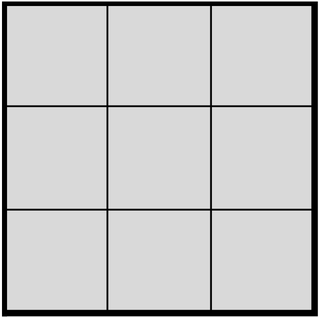
Computational complexity

- Each macro-element has $(\alpha + 2\beta)^d$ degrees of freedom
- Setup of patch-local solver costs $\mathcal{O}((\alpha + 2\beta)^{3d})$ flops
- Application of patch-local solver costs $\mathcal{O}((\alpha + 2\beta)^{2d})$ flops
- Update of residual costs $\mathcal{O}((\alpha + 2\beta)^d p^d)$ flops
- Number of macro-elements is $\eqsim N/\alpha^d$
- Total costs: $\mathcal{O}((1 + \alpha^{-1}\beta)^d((\alpha + \beta)^{2d} + p^d)N)$
  For $\beta \eqsim p$: $\mathcal{O}((1 + \alpha^{-1}p)^d(\alpha + p)^{2d}N)$
  For $\alpha, \beta \eqsim p$: $\mathcal{O}(p^{2d}N)$ (application: $\mathcal{O}(p^d N)$)

Macro-element Gauss-Seidel

$$\alpha := p, \qquad \beta := p - 1$$

Unit square

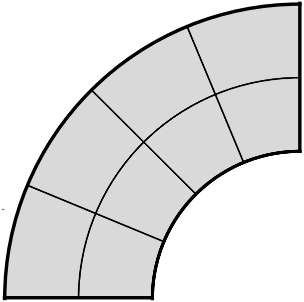


Iteration counts

$\ell \diagdown p$	1	2	3	4	5	6	7
3	8	4	3	3	2	2	1
4	8	4	3	3	3	3	2
5	8	4	3	3	3	3	3
6	8	4	3	3	3	3	3
7	8	4	4	3	3	3	3

V-cycle, $\epsilon = 10^{-8}$

Quarter annulus

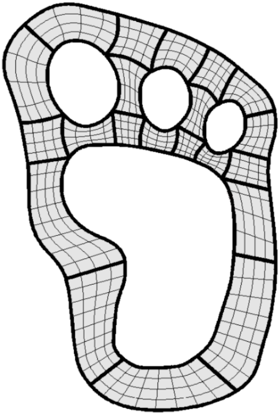


Iteration counts

$\ell \diagdown p$	1	2	3	4	5	6	7
3	12	4	3	3	2	2	1
4	14	5	4	3	3	3	2
5	16	5	4	3	3	3	3
6	18	5	4	3	3	3	3
7	19	5	4	3	3	3	3

V-cycle, $\epsilon = 10^{-8}$

Yeti footprint



Iteration counts

$\ell \diagdown p$	1	2	3	4	5	6	7
2	12	9	10	11	11	11	11
3	16	10	12	13	15	18	17
4	16	11	14	16	18	21	20

V-cycle, $\epsilon = 10^{-8}$

Conclusions

- Multigrid solvers can be fast in the IgA context.
- Thay are robust in the grid size.
- They can be provable robust in the spline degree (but maybe those are not the fastest ones).
- Simple Gauss-Seidel like constructions allow to solve for non-trivial problems more easily, although they might not have optimal complexity.

Conclusions

- Multigrid solvers can be fast in the IgA context.
- Thay are robust in the grid size.
- They can be provable robust in the spline degree (but maybe those are not the fastest ones).
- Simple Gauss-Seidel like constructions allow to solve for non-trivial problems more easily, although they might not have optimal complexity.

Conclusions

- Multigrid solvers can be fast in the IgA context.
- Thay are robust in the grid size.
- They can be provable robust in the spline degree (but maybe those are not the fastest ones).
- Simple Gauss-Seidel like constructions allow to solve for non-trivial problems more easily, although they might not have optimal complexity.

Conclusions

- Multigrid solvers can be fast in the IgA context.
- Thay are robust in the grid size.
- They can be provable robust in the spline degree (but maybe those are not the fastest ones).
- Simple Gauss-Seidel like constructions allow to solve for non-trivial problems more easily, although they might not have optimal complexity.

Conclusions

- Multigrid solvers can be fast in the IgA context.
- Thay are robust in the grid size.
- They can be provable robust in the spline degree (but maybe those are not the fastest ones).
- Simple Gauss-Seidel like constructions allow to solve for non-trivial problems more easily, although they might not have optimal complexity.

Thanks for your attention!

References

T. and Takacs.
Approximation error estimates and inverse inequalities for B-splines of maximum smoothness. *M^3AS*, 2016.

Hofreither and T.
Robust multigrid for isogeometric analysis using subspace correction. *SINUM*, 2017.

T.
Robust multigrid methods for isogeometric discretizations of the Stokes equation. *In Bjorstad et al (eds.): Domain Decomposition Methods in Science and Engineering XXIV*, 2019.

T.
Robust approximation error estimates and multigrid solvers for isogeometric multi-patch discretizations. *M^3AS*, 2018

Hofer and T.
A parallel multigrid solver for multi-patch Isogeometric Analysis. *To appear in Apel, Langer, Meyer, Steinbach (eds.): Advanced Finite Element Methods with Applications*, 2018.

T.
A quasi-robust discretization error estimate for discontinuous Galerkin Isogeometric Analysis. *Submitted*, 2019.

T.
Fast multigrid solvers for conforming and non-conforming multi-patch Isogeometric Analysis. *Submitted*, 2019.

Bressan and T.
Sum-factorization techniques in Isogeometric Analysis. *CMAME*, 2019 (to appear).