High-order isogeometric methods: Curse or blessing?

Matthias Möller\textsuperscript{a}

Joint work with R. Tielen\textsuperscript{a}, J. Liu\textsuperscript{a,*},
H. Schuttelaars\textsuperscript{a}, K. Vuik\textsuperscript{a}, D. G"{o}ddeke\textsuperscript{b}

\textsuperscript{a} Delft University of Technology, DIAM

\textsuperscript{b} University of Stuttgart, IANS

INdAM: Geometric Challenges in Isogeometric Analysis
Rome, January 31, 2020

\*CSC Scholarship
Outline

• Part 1: Solution **accuracy**
  • interplay of approximation and round-off error
  • towards an a-posteriori $hp$-adaptation strategy

• Part 2: Solver **efficiency**
  • $p$-multigrid method with ILUT smoother
  • discussion of choices and numerical examples

• Conclusion and outlook
Part 1: Solution accuracy
Model problem #1

**Poisson equation** in bounded domain $\Omega$ with Lipschitz continuous boundary $\Gamma$ with $f \in L^2(\Omega)$ and $h \in L^2(\Gamma_N)$:

\[-\Delta u = f \quad \text{in } \Omega\]
\[u = g \quad \text{on } \Gamma_D\]
\[\partial_n u = h \quad \text{on } \Gamma_N\]

If $\Omega$ is convex, $g = 0$, and $\Gamma_N = \emptyset$ then [Nečas 1967]

\[u \in H^2(\Omega) \quad \text{and} \quad \|u\|_{2,\Omega} \leq c(\Omega) \|f\|_{0,\Omega}\]

Otherwise $u \in H^1_{g,D}(\Omega) := \{v \in H^1(\Omega) : v = w + g, w \in H^1_{0,D}(\Omega)\}$
A-priori error analysis

**Weak form:** Find $u \in H^1_{g,D}(\Omega)$ such that

$$(\nabla u, \nabla w) = (f, w) + (h, w)_{\Gamma_N} \quad \forall w \in H^1_{0,D}(\Omega)$$

Optimal approximation property of the FEM

$$\inf_{v_h \in V_h^{(p)}} \| u - v_h \|_{0, \Omega} = O(h^{p+1})$$

$$\inf_{v_h \in V_h^{(p)}} \| \nabla_h (u - v_h) \|_{0, \Omega} = O(h^p)$$
A-priori error analysis

**Weak form:** Find \( u \in H^1_{g,D}(\Omega) \) such that

\[
(\nabla u, \nabla w) = (f, w) + (h, w)_{\Gamma_N} \quad \forall w \in H^1_{0,D}(\Omega)
\]

Optimal approximation property of the FEM

\[
\inf_{v_h \in V_h^{(p)}} \| u - v_h \|_{0,\Omega} = O(h^{p+1})
\]

\[
\inf_{v_h \in V_h^{(p)}} \| \nabla_h (u - v_h) \|_{0,\Omega} = O(h^p)
\]

**A word of caution:** *asymptotic* convergence for \( h \to 0 \) is combated by round-off errors in practical computations w/ finite-precision arithmetic
Interplay of approximation and round-off errors

![Graph showing the interplay of approximation and round-off errors. The graph plots the number of DoFs against absolute error. Key points include: $E_T$, $E_R$, $N_c$, $N_{opt}$, $E_{min}$, $E_c$, $\alpha_T$, $\alpha_R$, $\beta_T$, and $\beta_R$. The graph illustrates the relationship between the number of DoFs and the absolute error, highlighting the optimal number of DoFs for minimizing error.]
Interplay of approximation and round-off errors

Best *computable* solution $u_h$ is obtained for:

$$N_{\text{opt}} = \left( \frac{\alpha_T \beta_T}{\alpha_R \beta_R} \right)^{\frac{1}{\beta_T + \beta_R}}$$

with smallest possible error

$$E_{\text{min}} = \alpha_T \left( \frac{1}{N_{\text{opt}}} \right)^{\beta_T} + \alpha_R \left( \frac{1}{N_{\text{opt}}} \right)^{\beta_R}$$

---

*J. Liu, MM, H. Schuttelaars, arXiv: 1912.08004*
Best *computable* solution $u_h$ is obtained for

$$N_{\text{opt}} = \left( \frac{\alpha T \beta T}{\alpha R \beta R} \right)^{\frac{1}{\beta T + \beta R}}$$

with smallest possible error

$$E_{\text{min}} = \alpha T \left( \frac{1}{N_{\text{opt}}} \right)^{\beta T} + \alpha R \left( \frac{1}{N_{\text{opt}}} \right)^{\beta R}$$

- How sensitive are $\alpha_T, \beta_T, \alpha_R, \beta_R$ to problem parameters?

---

*J. Liu, MM, H. Schuttelaars, arXiv: 1912.08004*
Interplay of approximation and round-off errors

Best *computable* solution $u_h$ is obtained for*

$$N_{\text{opt}} = \left( \frac{\alpha T \beta T}{\alpha R \beta R} \right)^{\frac{1}{\beta T + \beta R}}$$

with smallest possible error

$$E_{\text{min}} = \alpha T \left( \frac{1}{N_{\text{opt}}} \right)^{\beta T} + \alpha R \left( \frac{1}{N_{\text{opt}}} \right)^{\beta R}$$

- How sensitive are $\alpha_T, \beta_T, \alpha_R, \beta_R$ to problem parameters?
- Can we develop an a-posteriori $hp$–adaptation strategy?

---

*J. Liu, MM, H. Schuttelaars, arXiv: 1912.08004*
$P_2$-FEM in 1d: $u(x) = (2\pi c_1)^{-2} \sin(2\pi c_1 x)$, $f(x) = \sin(2\pi c_1 x)$, $\Omega = (0,1)$

solution

first derivative

second derivative

Top row without scaling; bottom row with scaling $f/\|u\|$ and $u_h/\|u\|$
Analysis of further influence factors

- Type of boundary conditions: *no influence*
- Imposition of Dirichlet boundary conditions: *no influence*
- Computer precision: $\alpha_R$ changes, $\beta_R$ remains constant

All results (also using mixed FEM) were produced with deal.II code*

Analysis of further influence factors

- Type of boundary conditions: *no influence*
- Imposition of Dirichlet boundary conditions: *no influence*
- Computer precision: $\alpha_R$ changes, $\beta_R$ remains constant
- Solution strategy: *moderate influence*

All results (also using mixed FEM) were produced with deal.II code*

---

A-posteriori \(hp\)-adaptation strategy

Input: initial geometry with mesh width \(h\) and approximation order \(p\), tolerances for \(E_{\text{min}}\) and maximum mesh refinement steps

1. **Normalization**: compute \(u_h\) on coarse mesh and scale \(f/\|u_h\|\)

2. **Approximation error prediction**: compute \(u_h, u_h/2, \ldots\) on coarse meshes until asymptotic convergence rate is observed \(\rightarrow \alpha_T, \beta_T\)

3. **Round-off error prediction**: use lookup table from previous simulations or use manufactured solution that can be resolved exactly by \(P_p\)-FEM (possibly using lower precision) \(\rightarrow \alpha_R, \beta_R\)

4. **Effective error prediction**: compute \(N_{\text{opt}}\) and \(E_{\text{min}}\)

Output: \(N_{\text{opt}}\) and \(E_{\text{min}}\). If the estimated error satisfies the required tolerance compute \(u_{\text{opt}}\) otherwise repeat procedure with \(p := p + 1\) or switch to mixed FEM formulation
Model problem #2

Helmholtz equation:

\[((0.01 + x)(1.01 - x)u_x)_x - (0.01i)u(x) = 1.0 \text{ in } (0, 1)\]
\[u(0) = 0\]
\[u_x(1) = 0\]
Model problem #2

Helmholtz equation:

\[
((0.01 + x)(1.01 - x)u_x)_x - (0.01i)u(x) = 1.0 \quad \text{in } (0, 1)
\]

\[
u(0) = 0
\]

\[
u_x(1) = 0
\]
Is this of practical relevance?

Yes …

- since high-order methods can improve the 'effective' accuracy of solutions by orders of magnitudes
- since \( h \)-refinement is only effective in a small range of refinements for (very) high-order methods and should therefore be used with care

\( S_p^{p-1} \)-IGA solutions of model problem #1 with \( \Omega = (0, 1) \)
Is this of practical relevance?

Yes ...

- since high-order methods can improve the 'effective' accuracy of solutions by orders of magnitudes
- since $h$-refinement is only effective in a small range of refinements for (very) high-order methods and should therefore be used with care
- since the same phenomenon is observed already for moderately refined meshes in 2d (and 3d)

$S_p^{p-1}$-IGA solutions of model problem #1 with $\Omega = (0, 1)^2$
Part 2: Solver efficiency
Efficient solvers for IGA discretizations

**$h$-multigrid methods** enhanced with
- boundary corrected mass-Richardson smoother [Hofreither 2017]
- hybrid smoother [Sogn 2018]
- multiplicative Schwarz smoother [de la Riva 2018]
- ...

**Preconditioners** based on
- Schwarz methods [Beirão da Veiga 2012]
- Sylvester equation [Sangalli 2016]
- BPX for (T)HB [Bracco et al. 2019]
- ...

14 / 30
Basics of multigrid methods [Strang 2006]

Repeat until converged $u_{fine}$ is reached

1. **Iterate** on $A_{fine}u_{fine} = f_{fine}$ to reach $\tilde{u}_{fine}$
2. **Restrict** the residual $r_{fine} := f_{fine} - A_{fine}\tilde{u}_{fine}$ to the coarse level by applying the restriction operator, i.e. $r_{coarse} = I_{fine}^{coarse}r_{fine}$
3. **Solve** for the coarse level correction $A_{coarse}E_{coarse} = r_{coarse}$
4. **Prolongate** $E_{coarse}$ back to the fine level by $E_{fine} = I_{coarse}^{fine}E_{coarse}$
5. **Add** the correction, i.e. $\hat{u}_{fine} := \tilde{u}_{fine} + E_{fine}$
6. **Iterate** on $A_{fine}\hat{u}_{fine} = f_{fine}$ to reach $u_{fine}$

Step 3 calls the multigrid procedure recursively until a coarse level is reached, where the error equation can be solved 'exactly'.
Motivation for using $p$-multigrid methods

The linear system $A_{h,p} u_{h,p} = f_{h,p}$

- becomes more difficult to solve for increasing $p$
- reduces to $C^0$-FEM for $p = 1$ (where $h$-multigrid works fine)

In contrast to $h$-multigrid methods

- the #DoFs does not reduce significantly on coarser $p$-levels
- the stencil reduces significantly on coarse $p$-levels
- the spaces are not nested, i.e. $(S_{h,p}^{p-1} \not\subset S_{h,p-1}^{p-2} \not\subset \ldots)$
V-cycle $p$-multigrid variants

- ILUT or GS smoothing is applied at each level ($\bullet$)
- LU decomposition is applied as direct coarse level solver
Prolongation and restriction

**Prolongation in \( h \)**

\[ I_{2h,1}^{h,1} \text{ is linear interpolation} \]

**Restriction in \( h \)**

\[ I_{h,1}^{2h,1} = \frac{1}{2} \left( I_{2h,1}^{h,1} \right)^T \]

**Prolongation in \( p \)**

\[ I_{h,p}^{h,p-1} := (M_p^p)^{-1}M_{p-1}^p \]

**Restriction in \( p \)**

\[ I_{h,p}^{h,p-1} := (M_{p-1}^{p-1})^{-1}M_p^{p-1} \]

Let \( \phi_i^q \) denote the \( i^{th} \) basis function from \( S_{h,q}^{q-1} \). Then define

\[ (M_q^r)_{(i,j)} := \int_{\hat{\Omega}_h} \phi_i^q(\xi) \phi_j^r(\xi) c(\xi) \, d\hat{\Omega} \]

Replace \( M_q^q \) by its row-sum lumped counterpart (\( \rightarrow \) diagonal matrix)
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

$$e^{(s)}_{h,p} = (L_{h,p} U_{h,p})^{-1}(f_{h,p} - A_{h,p} u^{(s)}_{h,p})$$

$$u^{(s+1)}_{h,p} = u^{(s)}_{h,p} + e^{(s)}_{h,p}$$
Model problem #1, revisited

Obtaining coarse level operators

- Galerkin projection $\mathbf{A}^G_{h,p-1} = \mathcal{I}^{h,p-1}_h \mathbf{A}_{h,p} \mathcal{I}^{h,p}_h$
- re-discretization of $\mathbf{A}_{h,p}$ on each level

Poisson equation on quarter annulus with radii 1 and 2, $g = 0$, $\Gamma_N = \partial [1, 2]$, $f$ such that

$$u(x,y) = -\left(x^2 + y^2 - 1\right)\left(x^2 + y^2 - 4\right)xy^2$$

$$p = 2\kappa (\mathbf{A}^G_{h,1} - \mathbf{A}^{RD}_{h,1})$$

$$h = 2 - 4\cdot 10^{-0.0078}$$

$$h = 2 - 4\cdot 10^{-0.78}$$

$$h = 2 - 4\cdot 10^{-0.91}$$

$$h = 2 - 5\cdot 10^{-0.56}$$

$$h = 2 - 6\cdot 10^{-0.99}$$

$$h = 2 - 7\cdot 10^{-1.18}$$

$$h = 2 - 8\cdot 10^{-1.07}$$

$$h = 2 - 9\cdot 10^{-1.61}$$

$$h = 2 - 10\cdot 10^{-2.07}$$
Model problem #1, revisited

Obtaining coarse level operators

- Galerkin projection $A_{h,p-1}^G = I_{h,p}^h A_{h,p} I_{h,p-1}^h$
- re-discretization of $A_{h,p}$ on each level

**Poisson equation** on quarter annulus with radii 1 and 2, $g = 0$, $\Gamma_N = \emptyset$, $f$ such that $u(x, y) = -(x^2 + y^2 - 1)(x^2 + y^2 - 4)xy^2$

<table>
<thead>
<tr>
<th>$p = 2$</th>
<th>$\kappa(A_{h,1}^G)$</th>
<th>$\kappa(A_{h,1}^{RD})$</th>
<th>$p = 3$</th>
<th>$\kappa(A_{h,2}^G)$</th>
<th>$\kappa(A_{h,2}^{RD})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h = 2^{-4}$</td>
<td>$6.00 \cdot 10^7$</td>
<td>$9.78 \cdot 10^2$</td>
<td>$h = 2^{-4}$</td>
<td>$7.00 \cdot 10^9$</td>
<td>$1.56 \cdot 10^3$</td>
</tr>
<tr>
<td>$h = 2^{-5}$</td>
<td>$4.79 \cdot 10^9$</td>
<td>$4.19 \cdot 10^3$</td>
<td>$h = 2^{-5}$</td>
<td>$6.15 \cdot 10^{10}$</td>
<td>$6.71 \cdot 10^3$</td>
</tr>
<tr>
<td>$h = 2^{-6}$</td>
<td>$2.94 \cdot 10^{10}$</td>
<td>$1.76 \cdot 10^4$</td>
<td>$h = 2^{-6}$</td>
<td>$4.99 \cdot 10^{11}$</td>
<td>$2.84 \cdot 10^4$</td>
</tr>
<tr>
<td>$h = 2^{-7}$</td>
<td>$5.48 \cdot 10^{10}$</td>
<td>$7.28 \cdot 10^4$</td>
<td>$h = 2^{-7}$</td>
<td>$7.58 \cdot 10^{12}$</td>
<td>$1.18 \cdot 10^5$</td>
</tr>
</tbody>
</table>
V-cycle $p$-multigrid variants, revisited

- **Setup**: Assembly of $A_{h,p}$, $I_{h,p}^{h,p-1}$, $I_{h,p}^{h,p}$ each
  - ILUT factorization of $A_{h,p}$
  - Gauss-Seidel 'setup'
- **V-cycle**: Application of smoother, rest/prol each
  - $O\left(N_{dof}p^3d\right)$ flops
  - $O\left(N_{dof}p^2d\right)$ flops
  - $O\left(N_{dof}p^d\right)$ flops
V-cycle $p$-multigrid variants, revisited

- **Setup**: Assembly of $A_{h,p}$, $I_{h,p}^{h,p-1}$, $I_{h,p}^{h,p}$ each
  - $O(N_{dof}p^{3d})$ flops
- ILUT factorization of $A_{h,p}$
  - $O(N_{dof}p^{2d})$ flops
- Gauss-Seidel ’setup’
  - $O(N_{dof})$ flops
- **V-cycle**: Application of smoother, rest/prol each
  - $O(N_{dof}p^d)$ flops
- Numerical tests show same V-cycle counts for both variants
The final V-cycle $p$-multigrid variant

- ILUT ($p > 1$) / GS smoothing ($p = 1$) is applied at each level (●)
- LU decomposition is applied as direct coarse level solver
Model problem #1: V-cycle counts

V-cycle $p$-multigrid as a solver

<table>
<thead>
<tr>
<th></th>
<th>$p = 2$</th>
<th></th>
<th>$p = 3$</th>
<th></th>
<th>$p = 4$</th>
<th></th>
<th>$p = 5$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ILUT*</td>
<td>GS</td>
<td>ILUT*</td>
<td>GS</td>
<td>ILUT*</td>
<td>GS</td>
<td>ILUT*</td>
</tr>
<tr>
<td>$h = 2^{-6}$</td>
<td>4</td>
<td>30</td>
<td>3</td>
<td>62</td>
<td>3</td>
<td>176</td>
<td>3</td>
</tr>
<tr>
<td>$h = 2^{-7}$</td>
<td>4</td>
<td>29</td>
<td>3</td>
<td>61</td>
<td>3</td>
<td>172</td>
<td>3</td>
</tr>
<tr>
<td>$h = 2^{-8}$</td>
<td>5</td>
<td>30</td>
<td>3</td>
<td>60</td>
<td>3</td>
<td>163</td>
<td>3</td>
</tr>
<tr>
<td>$h = 2^{-9}$</td>
<td>5</td>
<td>32</td>
<td>3</td>
<td>61</td>
<td>3</td>
<td>163</td>
<td>3</td>
</tr>
</tbody>
</table>

V-cycle $h$-multigrid shows similar convergence behavior

*ILUT ($p > 1$), GS ($p = 1$)
Model problem #1: V-cycle counts

V-cycle $p$-multigrid as preconditioner in BiCGStab

<table>
<thead>
<tr>
<th></th>
<th>$p = 2$</th>
<th></th>
<th>$p = 3$</th>
<th></th>
<th>$p = 4$</th>
<th></th>
<th>$p = 5$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ILUT*</td>
<td>GS</td>
<td>ILUT*</td>
<td>GS</td>
<td>ILUT*</td>
<td>GS</td>
<td>ILUT*</td>
</tr>
<tr>
<td>$h = 2^{-6}$</td>
<td>2 13</td>
<td>2 18</td>
<td>2 41</td>
<td>2 78</td>
<td>2 41</td>
<td>2 92</td>
<td>2 43</td>
</tr>
<tr>
<td>$h = 2^{-7}$</td>
<td>2 12</td>
<td>2 20</td>
<td>2 41</td>
<td>2 92</td>
<td>2 43</td>
<td>2 95</td>
<td>2 41</td>
</tr>
<tr>
<td>$h = 2^{-8}$</td>
<td>3 13</td>
<td>2 19</td>
<td>2 43</td>
<td>2 95</td>
<td>2 41</td>
<td>2 95</td>
<td>2 41</td>
</tr>
<tr>
<td>$h = 2^{-9}$</td>
<td>3 13</td>
<td>2 21</td>
<td>2 41</td>
<td>2 95</td>
<td>2 41</td>
<td>2 95</td>
<td>2 41</td>
</tr>
</tbody>
</table>

V-cycle $h$-multigrid shows similar convergence behavior

*ILUT ($p > 1$), GS ($p = 1$)
Model problem #1: CPU times for $h = 2^{-6}$
Model problem #1: CPU times for $h = 2^{-7}$

![Bar chart showing CPU times for different methods and problem sizes.](chart.png)
Model problem #1: CPU times for $h = 2^{-8}$
Model problem #1: CPU times for $h = 2^{-9}$

- **CPU time in seconds**

- **Assembly**, **Factorize**, **Solve**

- **$p = 2$**, **$p = 3$**, **$p = 4$**, **$p = 5$**
Model problem #3

Convection-diffusion-reaction equation in $\Omega = (0,1)^2$

$$-\nabla \cdot \left( \begin{bmatrix} 1.2 & -0.7 \\ -0.4 & 0.9 \end{bmatrix} \nabla u \right) + \begin{bmatrix} 0.4 \\ -0.2 \end{bmatrix} \cdot \nabla u + 0.3 u = f \quad \text{in } \Omega$$

$$u = 0 \quad \text{on } \Gamma$$

with $f$ such that $u(x, y) = \sin(\pi x) \sin(\pi y)$
Model problem #3: V-cycle counts

V-cycle $p$-multigrid as a solver

<table>
<thead>
<tr>
<th></th>
<th>$p = 2$</th>
<th></th>
<th>$p = 3$</th>
<th></th>
<th>$p = 4$</th>
<th></th>
<th>$p = 5$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ILUT</td>
<td>GS</td>
<td>ILUT</td>
<td>GS</td>
<td>ILUT</td>
<td>GS</td>
<td>ILUT</td>
</tr>
<tr>
<td>$h = 2^{-6}$</td>
<td>5</td>
<td>–</td>
<td>3</td>
<td>–</td>
<td>3</td>
<td>–</td>
<td>4</td>
</tr>
<tr>
<td>$h = 2^{-7}$</td>
<td>5</td>
<td>–</td>
<td>3</td>
<td>–</td>
<td>4</td>
<td>–</td>
<td>4</td>
</tr>
<tr>
<td>$h = 2^{-8}$</td>
<td>5</td>
<td>–</td>
<td>3</td>
<td>–</td>
<td>3</td>
<td>–</td>
<td>4</td>
</tr>
<tr>
<td>$h = 2^{-9}$</td>
<td>5</td>
<td>–</td>
<td>4</td>
<td>–</td>
<td>3</td>
<td>–</td>
<td>4</td>
</tr>
</tbody>
</table>

V-cycle $h$-multigrid shows similar convergence behavior
Model problem #3: V-cycle counts

V-cycle \( p \)-multigrid as preconditioner in BiCGStab

<table>
<thead>
<tr>
<th>( h )</th>
<th>( p = 2 )</th>
<th>( p = 3 )</th>
<th>( p = 4 )</th>
<th>( p = 5 )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ILUT</td>
<td>GS</td>
<td>ILUT</td>
<td>GS</td>
</tr>
<tr>
<td>( 2^{-6} )</td>
<td>( 2 )</td>
<td>( 7 )</td>
<td>( 2 )</td>
<td>( 13 )</td>
</tr>
<tr>
<td>( 2^{-7} )</td>
<td>( 2 )</td>
<td>( 8 )</td>
<td>( 2 )</td>
<td>( 13 )</td>
</tr>
<tr>
<td>( 2^{-8} )</td>
<td>( 2 )</td>
<td>( 7 )</td>
<td>( 2 )</td>
<td>( 12 )</td>
</tr>
<tr>
<td>( 2^{-9} )</td>
<td>( 2 )</td>
<td>( 7 )</td>
<td>( 2 )</td>
<td>( 14 )</td>
</tr>
</tbody>
</table>

V-cycle \( h \)-multigrid shows similar convergence behavior
Conclusion and outlook

1. **a-posteriori hp-adaptation strategy** to find \((h, p)\) pair that ensures *computable* approximations with prescribed accuracy

2. **\(p\)-multigrid method with ILUT smoother** as efficient solver
Conclusion and outlook

1. **a-posteriori hp-adaptation strategy** to find \((h, p)\) pair that ensures *computable* approximations with prescribed accuracy
   - integration as fully automated procedure in simulation code
   - further analysis of influence factors, i.e. iterative solvers
   - use of number formats that are less sensitive to round-off errors

2. **\(p\)-multigrid method with ILUT smoother** as efficient solver
Conclusion and outlook

1. **a-posteriori hp-adaptation strategy** to find \((h, p)\) pair that ensures *computable* approximations with prescribed accuracy
   - integration as fully automated procedure in simulation code
   - further analysis of influence factors, i.e. iterative solvers
   - use of number formats that are less sensitive to round-off errors

2. **\(p\)-multigrid method with ILUT smoother** as efficient solver
   - application to biharmonic equation and within NSE solver
   - extension to block-ILUT smoother for multi-patch IGA
   - optimization of assembly procedure in G+Smo
Conclusion and outlook

1. **a-posteriori hp-adaptation strategy** to find \((h, p)\) pair that ensures *computable* approximations with prescribed accuracy
   - integration as fully automated procedure in simulation code
   - further analysis of influence factors, i.e. iterative solvers
   - use of number formats that are less sensitive to round-off errors

2. **p-multigrid method with ILUT smoother** as efficient solver
   - application to biharmonic equation and within NSE solver
   - extension to block-ILUT smoother for multi-patch IGA
   - optimization of assembly procedure in G+Smo

High-order methods, are they a curse or a blessing?
Conclusion and outlook

1. **a-posteriori hp-adaptation strategy** to find \((h, p)\) pair that ensures *computable* approximations with prescribed accuracy
   - integration as fully automated procedure in simulation code
   - further analysis of influence factors, i.e. iterative solvers
   - use of number formats that are less sensitive to round-off errors

2. **\(p\)-multigrid method with ILUT smoother** as efficient solver
   - application to biharmonic equation and within NSE solver
   - extension to block-ILUT smoother for multi-patch IGA
   - optimization of assembly procedure in G+Smo

High-order methods, are they a curse or a blessing? ... a challenge!

Thank you very much!