# Bridging the gap between isogeometric analysis and deep operator learning

Matthias Möller

Department of Applied Mathematics, TU Delft, The Netherlands

Seminar at Dipartimento di Matematica @ Università di Pavia, February 6, 2024

Joint work with Deepesh Toshniwal, Frank van Ruiten (TU Delft), Ye Ji, Mengyun Wang (TU Delft, Dalian), Casper van Leeuwen, Paul Melis (SURF), and Jaewook Lee (TU Vienna)

# Finite element analysis

**The mathematician**: Given a *geometry* $\mathbf{G}$, create a computational *mesh* $M_h$, define *basis functions* $\{b_i\}_i$, and determine the *coefficients* $\{u_i\}_i$ such that $u_{h,p} = \sum_i u_i b_i$ is a 'good' approximation to the exact solution $u$ of the (initial) boundary value problem at hand.

$$\mathbf{G} \quad \Rightarrow \quad M_h \quad \Rightarrow \quad \mathbb{S}_{h,p} = \text{span}\{b_i\}_i \quad \Rightarrow \quad u_{h,p} \in \mathbb{S}_{h,p} \quad \Rightarrow \quad \|u - u_{h,p}\| \leq c h^{p+1}$$

# Finite element analysis

**The mathematician**: Given a *geometry* $\mathbf{G}$, create a computational *mesh* $M_h$, define *basis functions* $\{b_i\}_i$, and determine the *coefficients* $\{u_i\}_i$ such that $u_{h,p} = \sum_i u_i b_i$ is a 'good' approximation to the exact solution $u$ of the (initial) boundary value problem at hand.

$$\mathbf{G} \quad \Rightarrow \quad M_h \quad \Rightarrow \quad \mathbb{S}_{h,p} = \mathsf{span}\{b_i\}_i \quad \Rightarrow \quad u_{h,p} \in \mathbb{S}_{h,p} \quad \Rightarrow \quad \|u - u_{h,p}\| \leq ch^{p+1}$$

**The engineer**: Given a use case, find an *optimal design* $\mathbf{D}$ (geometry, materials, etc.) that maximizes/minimizes one or more *key performance indicators* (weight, drag/lift, etc.)

$$\mathbf{D} \quad \Rightarrow \quad M_h \quad \Rightarrow \quad \mathbb{S}_{h,p} = \mathsf{span}\{b_i\}_i \quad \Rightarrow \quad u_{h,p} \in \mathbb{S}_{h,p} \quad \Rightarrow \quad \alpha, \beta, \gamma = \mathsf{KPI}(u_{h,p})$$

# Finite element analysis

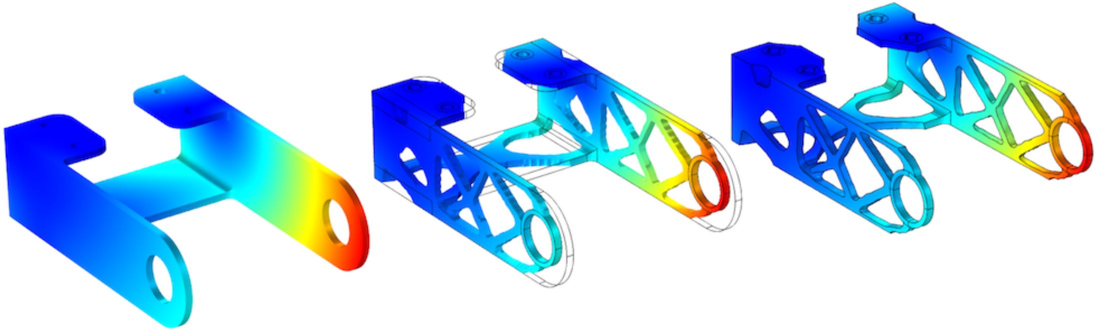
**The mathematician**: Given a *geometry* $\mathbf{G}$, create a computational *mesh* $M_h$, define *basis functions* $\{b_i\}_i$, and determine the *coefficients* $\{u_i\}_i$ such that $u_{h,p} = \sum_i u_i b_i$ is a 'good' approximation to the exact solution $u$ of the (initial) boundary value problem at hand.

$$\mathbf{G} \quad \Rightarrow \quad M_h \quad \Rightarrow \quad \mathbb{S}_{h,p} = \mathsf{span}\{b_i\}_i \quad \Rightarrow \quad u_{h,p} \in \mathbb{S}_{h,p} \quad \Rightarrow \quad \|u - u_{h,p}\| \leq ch^{p+1}$$

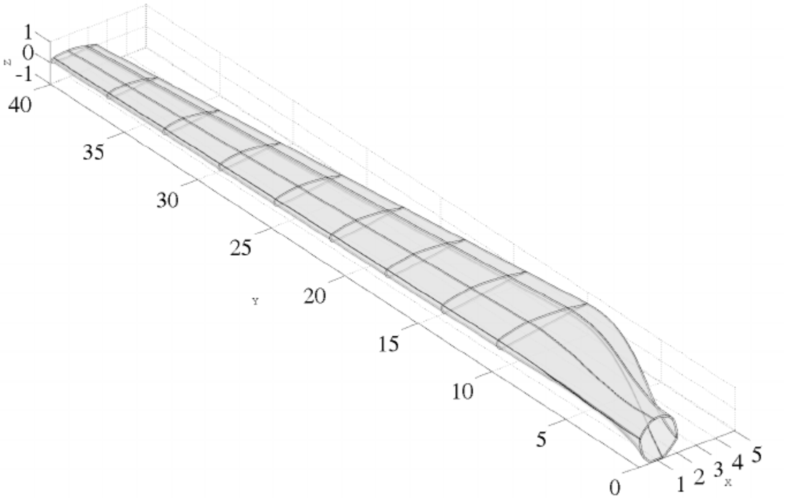
**The engineer**: Given a use case, find an *optimal design* $\mathbf{D}$ (geometry, materials, etc.) that maximizes/minimizes one or more *key performance indicators* (weight, drag/lift, etc.)

$$\mathbf{D} \quad \overset{?}{\Leftarrow} \quad M_h \quad \Leftarrow \quad \mathbb{S}_{h,p} = \mathsf{span}\{b_i\}_i \quad \Leftarrow \quad u_{h,p} \in \mathbb{S}_{h,p} \quad \Leftarrow \quad \alpha, \beta, \gamma = \mathsf{KPI}(u_{h,p})$$

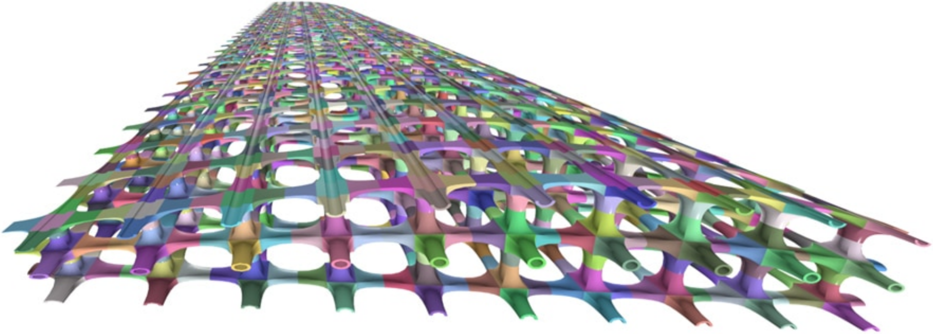
# Design optimization: topology



# Design optimization: shape

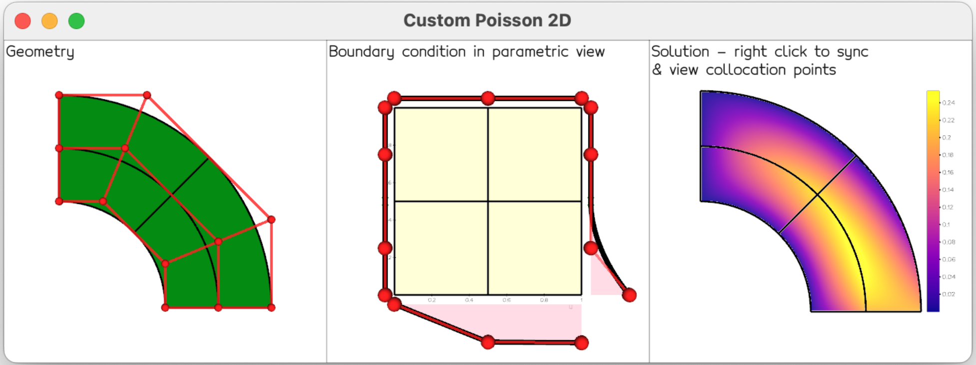


# Design optimization: meso-/micro-structure and materials



# Isogeometric analysis demo applet by J. Lee, TU Vienna

*pip install feigen; python3 -c "import feigen; feigen.CustomPoisson2D().start()"*




$$\mathbf{x}_h(\boldsymbol{\xi}) = \sum_{i \in \mathcal{I}_{\text{all}}} \mathbf{x}_i b_i(\boldsymbol{\xi}) \qquad u_D(\boldsymbol{\xi}) = \sum_{i \in \mathcal{I}_{\text{bdr}}} u_i b_i(\boldsymbol{\xi}) \qquad u_h(\boldsymbol{\xi}) = u_D(\boldsymbol{\xi}) + \sum_{i \in \mathcal{I}_{\text{in}}} u_i b_i(\boldsymbol{\xi})$$

# Outline

1. Introduction into basis splines
2. Introduction to isogeometric analysis
3. Some technicalities
4. Parametrization techniques

# Univariate B-splines

## Knot vector

$$\Xi = \{\xi_1, \xi_2, \dots, \xi_{n+d+1}\}, \qquad \xi_i \leq \xi_{i+1}, \quad \forall i = 1, \dots, n+d$$

with $\xi_i$ being a *knot*, $n$ the *number* and $d$ the *degree* of the B-spline basis functions

## Recurrence formula [de Boor, 1971]

$$b^0_{i;\Xi}(\xi) = \begin{cases} 1 & \text{if } \xi_i \leq \xi < \xi_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

$$b^d_{i;\Xi}(\xi) = \frac{\xi - \xi_i}{\xi_{i+d} - \xi_i} b^{d-1}_{i;\Xi}(\xi) + \frac{\xi_{i+d+1} - \xi}{\xi_{i+d+1} - \xi_{i+1}} b^{d-1}_{i+1;\Xi}(\xi) \qquad \text{“}\frac{0}{0}\text{”} := 0$$

# Univariate B-spline properties

## Local support and non-negativity

$$b^d_{i;\Xi}(\xi) \begin{cases} > 0 & \forall \xi \in \text{supp}\left(b^d_{i;\Xi}\right) := [\xi_i, \xi_{i+d+1}), \\ = 0 & \text{otherwise} \end{cases}$$

## Partition of unity

$$\sum_{i=1}^{n} b^d_{i;\Xi}(\xi) \equiv 1, \quad \forall \xi \in \hat{I}_\Xi := [\xi_1, \ldots \xi_{n+d+1})$$

# Knot vectors

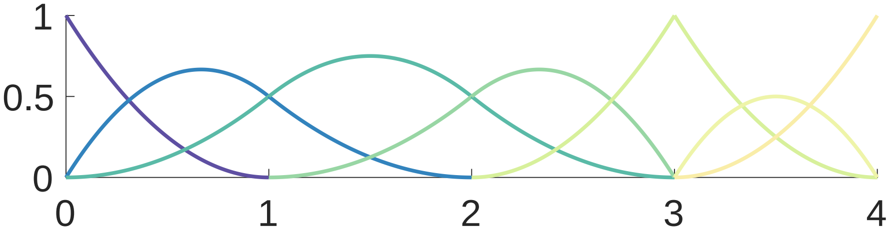
Open knot vector (i.e. $d+1$ repetition of the first and last knot)

$$\Xi = [\xi_1 = \cdots = \xi_{d+1}, \quad \xi_{d+2}, \ldots, \quad \xi_{n+1} = \cdots = \xi_{n+d+1}]$$

First and last basis functions are *interpolatory* at the left and right endpoint, respectively.

Repeated knots reduce the continuity of the basis functions that are non-zero at the respective knot from $C^{d-1}$ to $C^{d-m_i}$ *locally* with $m_i$ being the multiplicity of the $i$-th knot.

# The power of knot repetition




$$\Xi = \{0, 0, 0, 1, 2, 3, 3, 4, 4, 4\}, \quad \hat{I}_\Xi = (0, 4), \quad n = 7, \quad d = 2$$

# The spline space $\mathbb{S}_{\Xi}^{d,s}$

$$
\begin{aligned}
\mathbb{S}_{\Xi}^{d,s} &= \text{span}\left\{b_{1;\Xi}^{d}, \ldots, b_{n;\Xi}^{d}\right\} \\
&= \left\{\sum_{i=1}^{n} b_{i;\Xi}^{d}(\xi)\, c_i \;:\; c_i \in \mathbb{R}^s, \text{ for } 1 \leq i \leq n,\, \xi \in \hat{I}_{\Xi}\right\}
\end{aligned}
$$

Define spline function $f \in \mathbb{S}_{\Xi}^{d,s}$, i.e. mapping from $\hat{I}_{\Xi}$ to $\mathbb{R}^s$ through

$$
f(\xi) = \begin{bmatrix} b_{1;\Xi}^{d}(\xi) & \ldots & b_{n;\Xi}^{d}(\xi) \end{bmatrix} \cdot \begin{bmatrix} c_1 & \ldots & c_n \end{bmatrix} = \mathbf{b} \cdot \mathbf{c}
$$

and fix the B-spline coefficients $c_i \in \mathbb{R}^s$ relative to the given B-spline basis

# Greville abscissae and control polygon

Greville abscissae (i.e. parameter values where basis functions attain maximum values)

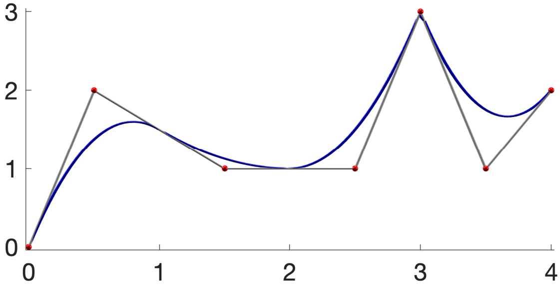
$$\bar{\xi}_i = \frac{\xi_{i+1} + \cdots + \xi_{i+d}}{d}$$

have a special geometric interpretation; the pairs $(\bar{\xi}_i, c_i)$ form the control polygon

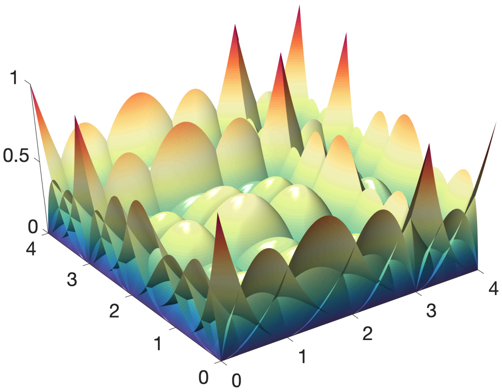
Example: $n = 7, \quad d = 2$

$$\Xi = \{0, 0, 0, 1, 2, 3, 3, 4, 4, 4\}$$

$$\mathbf{c} = \begin{bmatrix} 0 & 2 & 1 & 1 & 3 & 1 & 2 \end{bmatrix}$$



# Multi-variate B-splines




Tensor-product basis functions

$$b^{\mathbf{d}}_{\mathbf{i};\boldsymbol{\Xi}}(\boldsymbol{\xi}) = \prod_{k=1}^{p} b^{d_k}_{i_k;\Xi_k}(\xi_k)$$

with $\mathbf{i} = (i_1, \ldots, i_p)$, $\mathbf{d} = (d_1, \ldots, d_p)$,

$\boldsymbol{\xi} = (\xi_1, \ldots, \xi_p)$, $\boldsymbol{\Xi} = (\Xi_1, \ldots, \Xi_p)$,

and parametric domain

$$\hat{\Omega}_{\boldsymbol{\Xi}} = \bigotimes_{k=1}^{p} [\xi_{k,d_k+1}, \xi_{k,n_k})$$

$$\mathbb{S}^{\mathbf{d},s}_{\boldsymbol{\Xi}} = \text{span}\left\{b^{\mathbf{d}}_{\mathbf{1};\boldsymbol{\Xi}}, \ldots, b^{\mathbf{d}}_{\mathbf{n};\boldsymbol{\Xi}}\right\} = \left\{\sum_{\mathbf{i}=\mathbf{1}}^{\mathbf{n}} c_{\mathbf{i}}\, b^{\mathbf{d}}_{\mathbf{i};\boldsymbol{\Xi}}(\boldsymbol{\xi}) \;:\; c_{\mathbf{i}} \in \mathbb{R}^s, \text{ for } \mathbf{1} \le \mathbf{i} \le \mathbf{n},\ \boldsymbol{\xi} \in \hat{\Omega}_{\boldsymbol{\Xi}}\right\}$$

# Collocation IgA

**PDE problem**

$$\mathcal{L}u = f \qquad \text{in } \Omega$$

$$\mathcal{B}u = g \qquad \text{on } \Gamma$$

**Weighted residual form**

$$\int_{\Omega} \phi_{\Omega}(\mathcal{L}u - f)\,\mathrm{d}\mathbf{x} + \int_{\Gamma} \phi_{\Gamma}(\mathcal{B}u - g)\,\mathrm{d}s = 0$$

# Collocation IgA

**PDE problem**

$$\mathcal{L}u = f \qquad \text{in } \Omega$$

$$\mathcal{B}u = g \qquad \text{on } \Gamma$$

**Weighted residual form**

$$\int_\Omega \phi_\Omega(\mathcal{L}u - f)\,\mathrm{d}\mathbf{x} + \int_\Gamma \phi_\Gamma(\mathcal{B}u - g)\,\mathrm{d}s = 0$$

Let

$$\phi_\Omega = \sum_{i=1}^{k} \delta_\Omega(\mathbf{x} - \mathbf{x}_i)\, c_i \quad (\mathbf{x}_i \in \Omega) \qquad \text{and} \qquad \phi_\Gamma = \sum_{i=k+1}^{n} \delta_\Gamma(\mathbf{x} - \mathbf{x}_i)\, c_i \quad (\mathbf{x}_i \in \Gamma)$$

then

$$\sum_{i=1}^{k} \left(\mathcal{L}u(\mathbf{x}_i) - f(\mathbf{x}_i)\right)\, c_i + \sum_{i=1+k}^{n} \left(\mathcal{B}u(\mathbf{x}_i) - g(\mathbf{x}_i)\right)\, c_i = 0$$

# Collocation IgA cont'd

As the coefficients $c_i$ are arbitrary we obtain

$$\mathcal{L}u(\mathbf{x}_i) = f(\mathbf{x}_i) \qquad i = 1, \ldots, k$$

$$\mathcal{B}u(\mathbf{x}_i) = g(\mathbf{x}_i) \qquad i = k+1, \ldots, n$$

# Collocation IgA cont'd

As the coefficients $c_i$ are arbitrary and replacing $u \approx u_h = \sum_{j=1}^{n} b_j(\mathbf{x})u_j$ we obtain

$$\begin{bmatrix} \mathcal{L}b_1(\mathbf{x}_1) & \dots & \mathcal{L}b_n(\mathbf{x}_1) \\ \vdots & \ddots & \vdots \\ \mathcal{L}b_1(\mathbf{x}_k) & \dots & \mathcal{L}b_n(\mathbf{x}_k) \\ \mathcal{B}b_1(\mathbf{x}_{k+1}) & \dots & \mathcal{B}b_n(\mathbf{x}_{k+1}) \\ \vdots & \ddots & \vdots \\ \mathcal{B}b_1(\mathbf{x}_n) & \dots & \mathcal{B}b_n(\mathbf{x}_n) \end{bmatrix} \begin{bmatrix} u_1 \\ \vdots \\ u_k \\ u_{k+1} \\ \vdots \\ u_n \end{bmatrix} = \begin{bmatrix} f(\mathbf{x}_1) \\ \vdots \\ f(\mathbf{x}_k) \\ g(\mathbf{x}_{k+1}) \\ \vdots \\ g(\mathbf{x}_n) \end{bmatrix}$$

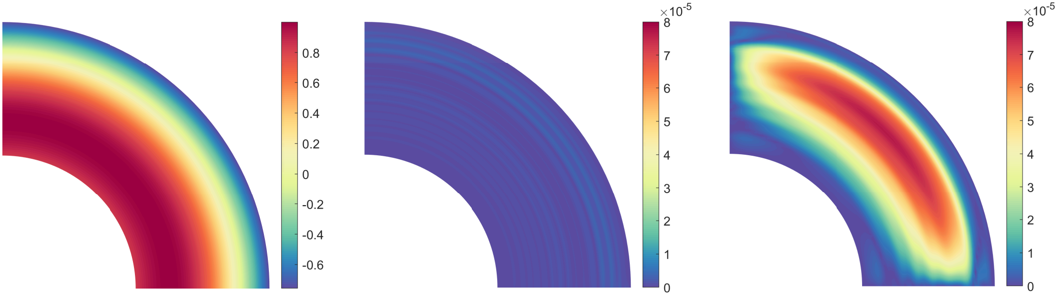
# Collocation IgA cont'd

As the coefficients $c_i$ are arbitrary and replacing $u \approx u_h = \sum_{j=1}^{n} b_j(\mathbf{x})u_j$ we obtain

$$\begin{bmatrix} \mathcal{L}b_1(\mathbf{x}_1) & \dots & \mathcal{L}b_n(\mathbf{x}_1) \\ \vdots & \ddots & \vdots \\ \mathcal{L}b_1(\mathbf{x}_k) & \dots & \mathcal{L}b_n(\mathbf{x}_k) \\ \mathcal{B}b_1(\mathbf{x}_{k+1}) & \dots & \mathcal{B}b_n(\mathbf{x}_{k+1}) \\ \vdots & \ddots & \vdots \\ \mathcal{B}b_1(\mathbf{x}_n) & \dots & \mathcal{B}b_n(\mathbf{x}_n) \end{bmatrix} \begin{bmatrix} u_1 \\ \vdots \\ u_k \\ u_{k+1} \\ \vdots \\ u_n \end{bmatrix} = \begin{bmatrix} f(\mathbf{x}_1) \\ \vdots \\ f(\mathbf{x}_k) \\ g(\mathbf{x}_{k+1}) \\ \vdots \\ g(\mathbf{x}_n) \end{bmatrix}$$

- basis functions $b_i$ need to be at least $C^\ell$ such that $\mathcal{L}$ and $\mathcal{B}$ can be applied
- regular system matrix requires that #collocation points = #basis functions and all collocation points must be pairwise distinct

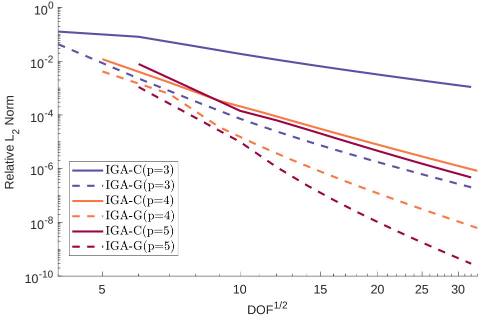
# Comparison between Galerkin and collocation IgA



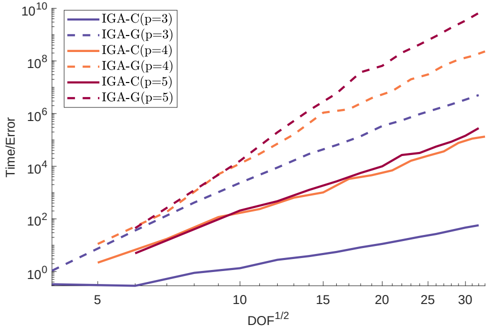

solution | IgA-G abs. error | IgA-C abs. error

Results by Mengyun Wang

# Comparison between Galerkin and collocation IgA



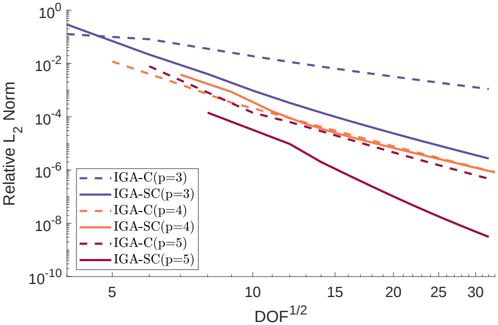

rel. $L_2$-error



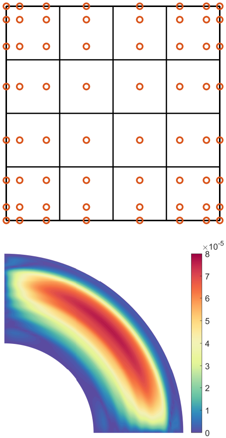

wallclock time/error

Results by Mengyun Wang

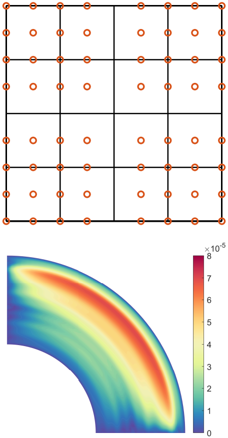
# Comparison between Greville and clustered superconvergent points




rel. $L_2$-error



Greville



SC points

Results by Mengyun Wang; SC points from Montardini et al. [2017]

# Least-squares collocation IgA

Idea: When #collocation points $(m)$ > #unknowns $(n)$ then the system matrix is over-determined and the system can be solved in least-squares manner

$$\min_{u_h} \frac{1}{k} \sum_{i=1}^{k} \|\mathcal{L} u_h(\mathbf{x}_i) - f(\mathbf{x}_i)\|^2 + \frac{1}{m-k} \sum_{i=k+1}^{m} \|\mathcal{B} u_h(\mathbf{x}_i) - g(\mathbf{x}_i)\|^2$$

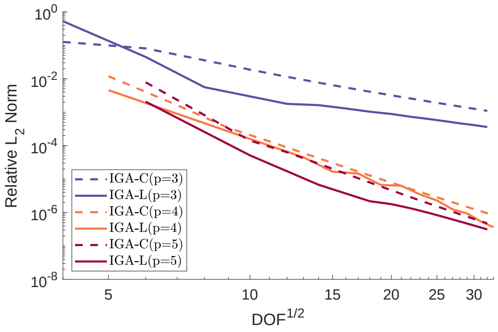
# Least-squares collocation IgA

Idea: When #collocation points ($m$) $>$ #unknowns ($n$) then the system matrix is over-determined and the system can be solved in least-squares manner

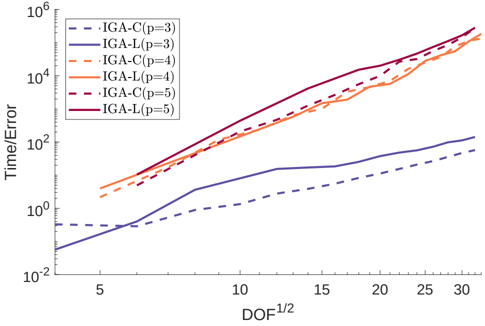
$$\min_{u_h} \frac{1}{k} \sum_{i=1}^{k} \|\mathcal{L}u_h(\mathbf{x}_i) - f(\mathbf{x}_i)\|^2 + \frac{1}{m-k} \sum_{i=k+1}^{m} \|\mathcal{B}u_h(\mathbf{x}_i) - g(\mathbf{x}_i)\|^2$$

[Lin et al., 2020] derives rigorous conditions under which least-squares collocation IgA (IgA-L) is consistent and convergent. In essence, there must be *at least one collocation point per element* (e.g., Greville points) but we can use more to increase the resolution.

# Comparison between collocation and least-squares collocation IgA




rel. $L_2$-error




wallclock time/error

Results by Mengyun Wang

# Least-squares collocation IgA revisited

Replacing $f$, and $g$ by their approximations $f_h$, and $g_h$ we obtain

$$\min_{\{u_j\}_j} \frac{1}{k} \sum_{i=1}^{k} \| \sum_{j=1}^{n} \mathcal{L} b_j(\mathbf{x}_i) u_j - b_j(\mathbf{x}_i) f_j \|^2 + \frac{1}{m-k} \sum_{i=k+1}^{m} \| \sum_{j=1}^{n} \mathcal{B} b_j(\mathbf{x}_i) u_j - b_j(\mathbf{x}_i) g_j \|^2$$

- B-spline basis functions $\hat{b}_j(\boldsymbol{\xi})$ are defined in the reference space $\hat{\Omega} = (0,1)^d$ and are mapped into physical space $\Omega$ through the *push-forward mapping*

$$\mathbf{x}_h(\boldsymbol{\xi}) = \sum_{j=1}^{n} \hat{b}_j(\boldsymbol{\xi}) \mathbf{x}_j,$$

# Least-squares collocation IgA revisited

Replacing $f$, and $g$ by their approximations $f_h$, and $g_h$ we obtain

$$\min_{\{u_j\}_j} \underbrace{\frac{1}{k}\sum_{i=1}^{k}\|\sum_{j=1}^{n}\mathcal{L}b_j(\mathbf{x}_i)u_j - b_j(\mathbf{x}_i)f_j\|^2}_{\text{loss}_{\text{PDE}}(\{u_j\}_j,\{f_j\}_j;\{\mathbf{x}_i\}_i)} + \underbrace{\frac{1}{m-k}\sum_{i=k+1}^{m}\|\sum_{j=1}^{n}\mathcal{B}b_j(\mathbf{x}_i)u_j - b_j(\mathbf{x}_i)g_j\|^2}_{\text{loss}_{\text{BC}}(\{u_j\}_j,\{g_j\}_j;\{\mathbf{x}_i\}_i)}$$

- B-spline basis functions $\hat{b}_j(\boldsymbol{\xi})$ are defined in the reference space $\hat{\Omega} = (0,1)^d$ and are mapped into physical space $\Omega$ through the *push-forward mapping*

$$\mathbf{x}_h(\boldsymbol{\xi}) = \sum_{j=1}^{n}\hat{b}_j(\boldsymbol{\xi})\mathbf{x}_j,$$

- problem is fully parameterized through $f_j$'s, $g_j$'s, and $\mathbf{x}_j$'s relative to a fixed basis $\hat{b}_j$

# Least-squares collocation IgA revisited

Replacing $f$, and $g$ by their approximations $f_h$, and $g_h$ we obtain

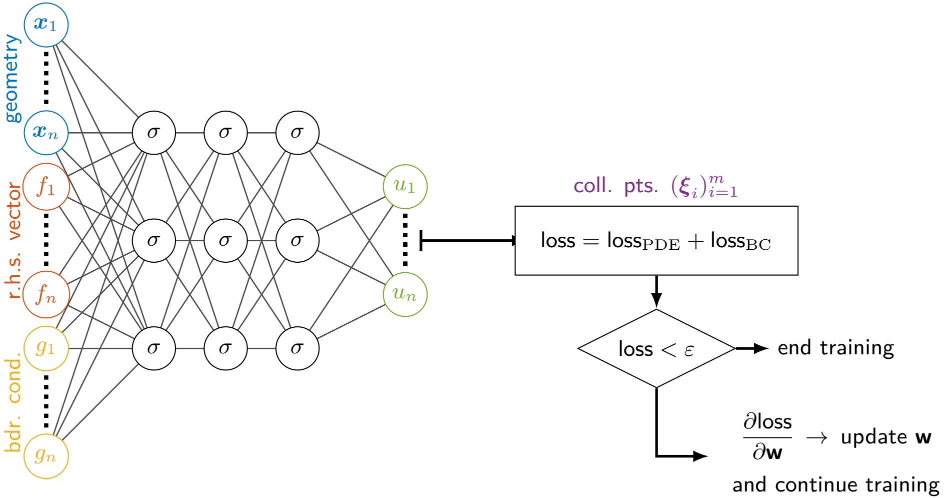
$$\min_{\{u_j\}_j} \underbrace{\frac{1}{k}\sum_{i=1}^{k}\|\sum_{j=1}^{n}\mathcal{L}b_j(\mathbf{x}_i)u_j - b_j(\mathbf{x}_i)f_j\|^2}_{\text{loss}_{\text{PDE}}(\{u_j\}_j,\{f_j\}_j;\{\mathbf{x}_i\}_i)} + \underbrace{\frac{1}{m-k}\sum_{i=k+1}^{m}\|\sum_{j=1}^{n}\mathcal{B}b_j(\mathbf{x}_i)u_j - b_j(\mathbf{x}_i)g_j\|^2}_{\text{loss}_{\text{BC}}(\{u_j\}_j,\{g_j\}_j;\{\mathbf{x}_i\}_i)}$$

- B-spline basis functions $\hat{b}_j(\boldsymbol{\xi})$ are defined in the reference space $\hat{\Omega} = (0,1)^d$ and are mapped into physical space $\Omega$ through the *push-forward mapping*

$$\mathbf{x}_h(\boldsymbol{\xi}) = \sum_{j=1}^{n} \hat{b}_j(\boldsymbol{\xi})\mathbf{x}_j,$$

- problem is fully parameterized through $f_j$'s, $g_j$'s, and $\mathbf{x}_j$'s relative to a fixed basis $\hat{b}_j$

# IgANet architecture




Can be interpreted as an alternative way to solve the least-squares problem (IgA-L)

# Training and evaluation

**Training**

**For** $[f_1, \ldots, f_n] \in \mathcal{S}_{\text{rhs}}$, $[g_1, \ldots, g_n] \in \mathcal{S}_{\text{bcond}}$, $[\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathcal{S}_{\text{geo}}$ **do**

**For** a batch of collocation points $\boldsymbol{\xi}_i \in [0,1]^2$ (e.g., Greville points + more) **do**

Train IgANet $([f_1, \ldots, f_n], [g_1, \ldots, g_n], [\mathbf{x}_1, \ldots, \mathbf{x}_n]) \mapsto [u_1, \ldots, u_n]$

**EndFor**

**EndFor**

# Training and evaluation

**Training**

**For** $[f_1, \dots, f_n] \in \mathcal{S}_{\text{rhs}}$, $[g_1, \dots, g_n] \in \mathcal{S}_{\text{bcond}}$, $[\mathbf{x}_1, \dots, \mathbf{x}_n] \in \mathcal{S}_{\text{geo}}$ **do**

**For** a batch of collocation points $\boldsymbol{\xi}_i \in [0,1]^2$ (e.g., Greville points + more) **do**

Train IgANet $([f_1, \dots, f_n], [g_1, \dots, g_n], [\mathbf{x}_1, \dots, \mathbf{x}_n]) \mapsto [u_1, \dots, u_n]$

**EndFor**

**EndFor**

**Evaluation**

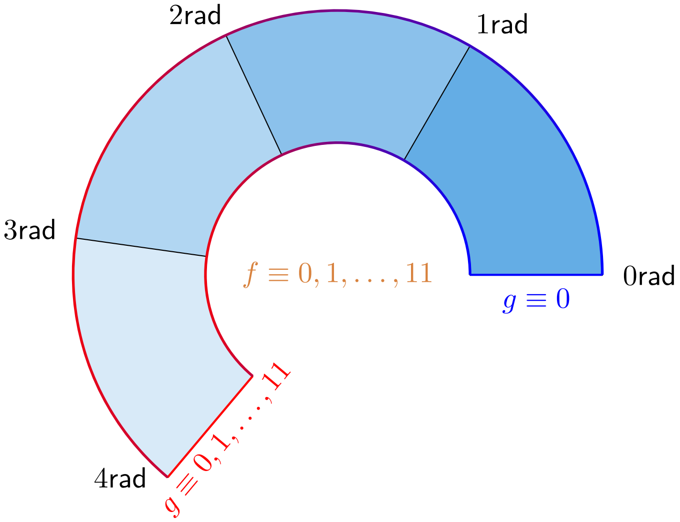
**For** $[f_1, \dots, f_n] \in \mathcal{S}_{\text{rhs}}$, $[g_1, \dots, g_n] \in \mathcal{S}_{\text{bcond}}$, $[\mathbf{x}_1, \dots, \mathbf{x}_n] \in \mathcal{S}_{\text{geo}}$ **do**

Evaluate IgANet $([f_1, \dots, f_n], [g_1, \dots, g_n], [\mathbf{x}_1, \dots, \mathbf{x}_n]) \mapsto [u_1, \dots, u_n]$

Use basis representation $u_h(\mathbf{x}) = \sum_{j=1}^{n} b_j(\mathbf{x}) u_j$ for all further purposes

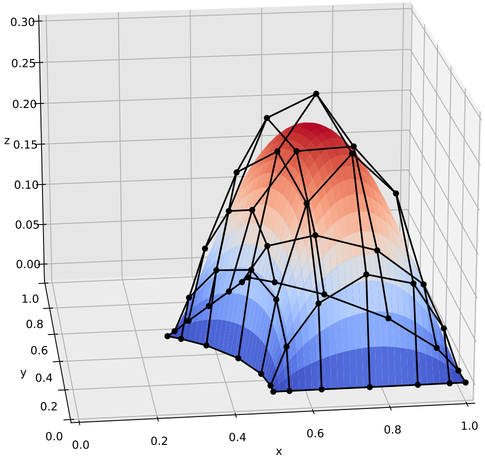
**EndFor**

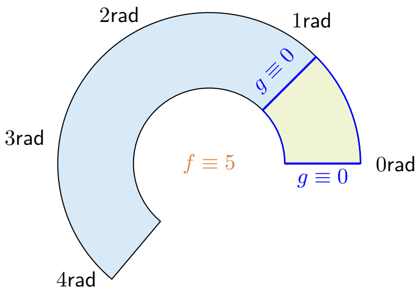
# Test case: Poisson's equation on a variable annulus




Master thesis work by Frank van Ruiten, TU Delft

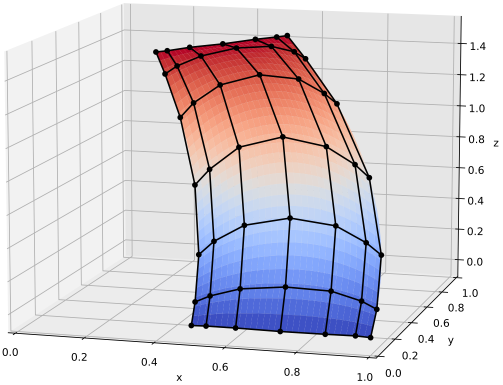
# Validation results

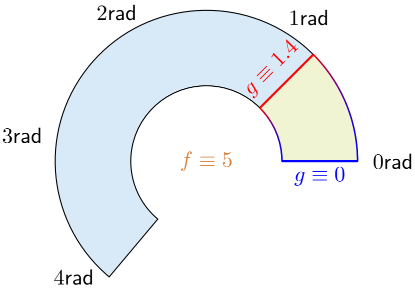



Master thesis work by Frank van Ruiten, TU Delft

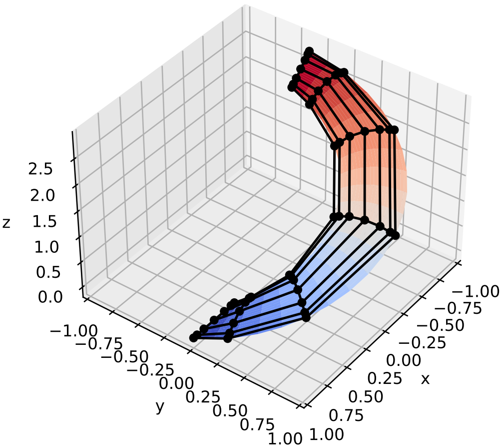
# Validation results

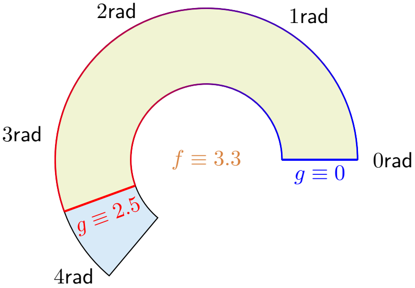



Master thesis work by Frank van Ruiten, TU Delft

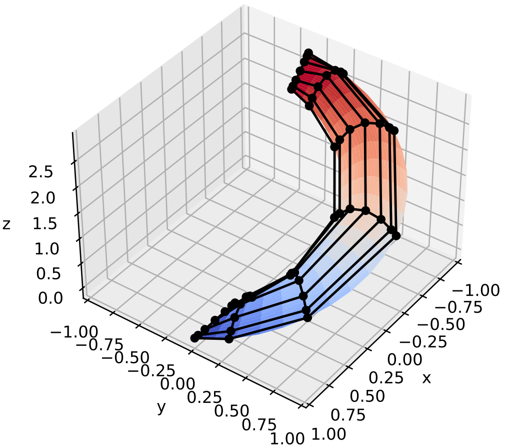
# Validation results

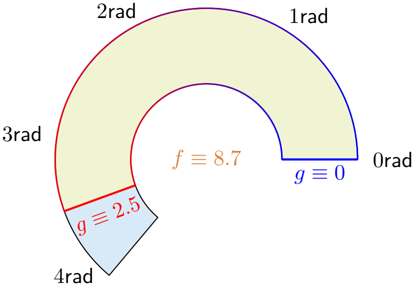



Master thesis work by Frank van Ruiten, TU Delft

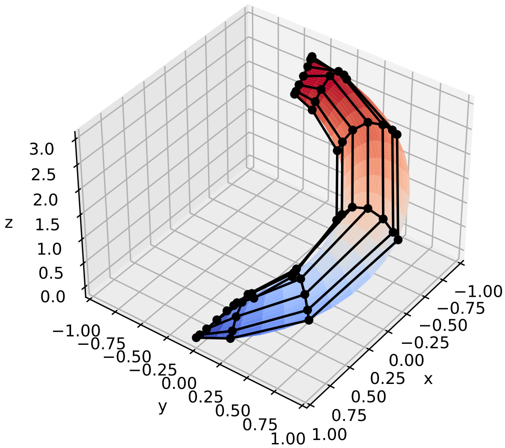
# Validation results

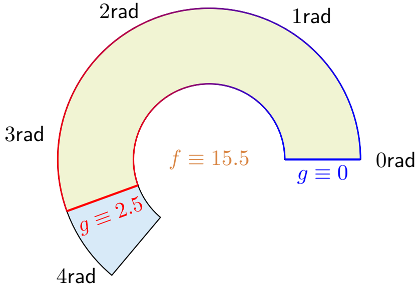



Master thesis work by Frank van Ruiten, TU Delft

# Validation results



# Computing derivatives

Derivatives occurring in the loss function can be computed in the traditional way, i.e.

$$\text{loss}_{\text{PDE}} = \frac{1}{k}\sum_{i=1}^{k} \left|\Delta\left[u_h \circ \mathbf{x}_h\left(\boldsymbol{\xi}_i\right)\right] - f_h \circ \mathbf{x}_h\left(\boldsymbol{\xi}_i\right)\right|^2$$

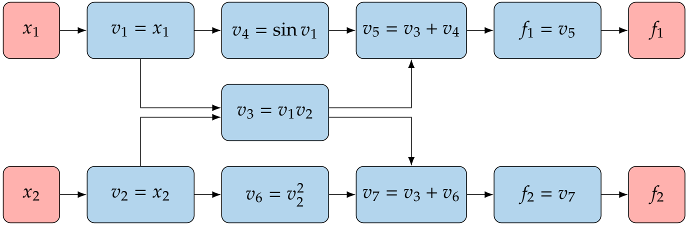
Implementation: `bspline.ilaplace(Geo, Xi)`

Derivatives of the loss function w.r.t. the weights and biases of the neural network – only needed during training – are computed using *reverse-mode algorithmic differentiation*

Implementation:

```
nn.zero_grad(); out = nn.forward(Geo, Xi);
loss = L(out, ...); loss.backward();
```

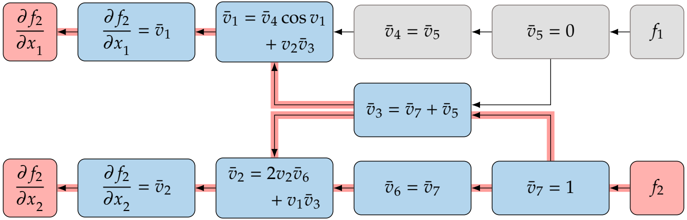
# Reverse-mode AD example




Reverse-mode AD requires two passes. The forward pass creates the computational graph.

Source: Fig 6.17 from Martins and Ning [2021].

# Reverse-mode AD example

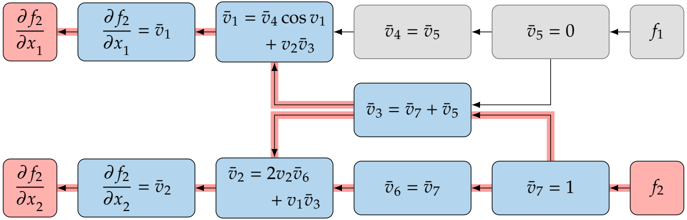



Reverse-mode AD requires two passes. The forward pass creates the computational graph. The backward pass computes the gradients from the values stored on the 'tape'.

$$
\begin{aligned}
\bar{v}_7 &= 1 \\
\bar{v}_6 &= \frac{\partial v_7}{\partial v_6}\bar{v}_7 = \bar{v}_7 = 1 \\
\bar{v}_5 &= 0 \\
\bar{v}_4 &= \frac{\partial v_5}{\partial v_4} = \bar{v}_5 = 0 \\
\bar{v}_3 &= \frac{\partial v_7}{\partial v_3}\bar{v}_7 + \frac{\partial v_5}{\partial v_3}\bar{v}_5 \\
&= \bar{v}_7 + \bar{v}_5 = 1 \\
\bar{v}_2 &= \frac{\partial v_6}{\partial v_2}\bar{v}_6 + \frac{\partial v_3}{\partial v_2}\bar{v}_3 \\
&= 2v_2\bar{v}_6 + v_1\bar{v}_3 = 4.785 = \frac{\partial f_2}{\partial x_2} \\
\bar{v}_1 &= \frac{\partial v_4}{\partial v_1}\bar{v}_4 + \frac{\partial v_3}{\partial v_1}\bar{v}_3 \\
&= (\cos v_1)\bar{v}_4 + v_2\bar{v}_3 = 2 = \frac{\partial f_2}{\partial x_1}
\end{aligned}
$$

Source: Fig 6.17 from Martins and Ning [2021].

# Reverse-mode AD example




$$
\begin{aligned}
\bar{v}_7 &= 1 \\
\bar{v}_6 &= \frac{\partial v_7}{\partial v_6}\bar{v}_7 = \bar{v}_7 = 1 \\
\bar{v}_5 &= 0 \\
\bar{v}_4 &= \frac{\partial v_5}{\partial v_4} = \bar{v}_5 = 0 \\
\bar{v}_3 &= \frac{\partial v_7}{\partial v_3}\bar{v}_7 + \frac{\partial v_5}{\partial v_3}\bar{v}_5 \\
&= \bar{v}_7 + \bar{v}_5 = 1 \\
\bar{v}_2 &= \frac{\partial v_6}{\partial v_2}\bar{v}_6 + \frac{\partial v_3}{\partial v_2}\bar{v}_3 \\
&= 2v_2\bar{v}_6 + v_1\bar{v}_3 = 4.785 = \frac{\partial f_2}{\partial x_2} \\
\bar{v}_1 &= \frac{\partial v_4}{\partial v_1}\bar{v}_4 + \frac{\partial v_3}{\partial v_1}\bar{v}_3 \\
&= (\cos v_1)\bar{v}_4 + v_2\bar{v}_3 = 2 = \frac{\partial f_2}{\partial x_1}
\end{aligned}
$$

Reverse-mode AD requires two passes. The forward pass creates the computational graph. The backward pass computes the gradients from the values stored on the 'tape'.

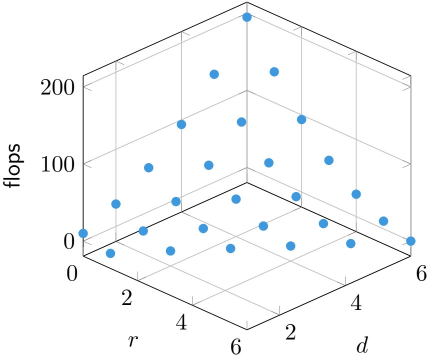
All operations must be of the form `out = f(in);`

Source: Fig 6.17 from Martins and Ning [2021].

# An efficient algorithm for evaluating univariate B-splines

Algorithm 2.22 from [Lyche and Mørken, 2018] with modifications

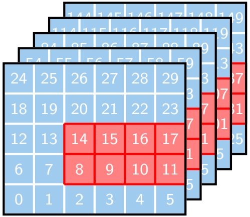
1. $\mathbf{b} = 1$
2. For $k = 1, \ldots, d - r$
    1. $\mathbf{t}_1 = (\xi_{i-k+1}, \ldots, \xi_i)$
    2. $\mathbf{t}_{21} = (\xi_{i+1}, \ldots, \xi_{i+k}) - \mathbf{t}_1$
    3. $\textsf{mask} = (\mathbf{t}_{21} < \textsf{tol})$
    4. $\mathbf{w} = (\xi - \mathbf{t}_1 - \textsf{mask}) \div (\mathbf{t}_{21} - \textsf{mask})$
    5. $\mathbf{b} = [(1 - \mathbf{w}) \odot \mathbf{b}, 0] + [0, \mathbf{w} \odot \mathbf{b}]$
3. For $k = d - r + 1, \ldots, d$
    1. $\mathbf{t}_1 = (\xi_{i-k+1}, \ldots, \xi_i)$
    2. $\mathbf{t}_{21} = (\xi_{i+1}, \ldots, \xi_{i+k}) - \mathbf{t}_1$
    3. $\textsf{mask} = (\mathbf{t}_{21} < \textsf{tol})$
    4. $\mathbf{w} = (1 - \textsf{mask}) \div (\mathbf{t}_{21} - \textsf{mask})$
    5. $\mathbf{b} = [-\mathbf{w} \odot \mathbf{b}, 0] + [0, \mathbf{w} \odot \mathbf{b}]$




where $\div$ and $\odot$ denote the element-wise division and multiplication of vectors, respectively.

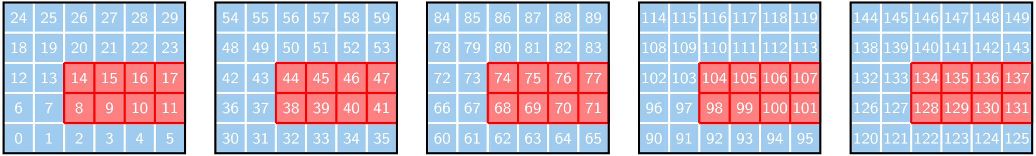
# Memory layout of tensors

Example: $n_1 = 6, n_2 = 5, n_3 = 5$ and $d_1 = 3, d_2 = 1, d_3 = 4$



# Memory layout of tensors

Example: $n_1 = 6, n_2 = 5, n_3 = 5$ and $d_1 = 3, d_2 = 1, d_3 = 4$




$$\mathbf{c} = [8 \quad 9 \quad 10 \quad 11 \quad 14 \quad 15 \quad 16 \quad 17 \quad 38 \quad 39 \quad \ldots \quad 136 \quad 137]$$

# A brief recap of the Kronecker product

Definition

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} \mathbf{A}_{11}\mathbf{B} & \mathbf{A}_{12}\mathbf{B} \\ \mathbf{A}_{21}\mathbf{B} & \mathbf{A}_{22}\mathbf{B} \end{bmatrix} \qquad (n_A \cdot m_A \cdot n_B \cdot m_B \text{ flops})$$

Mixed-product property (if matrices are so that $\mathbf{AC}$ and $\mathbf{BD}$ exists)

$$(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = (\mathbf{AC}) \otimes (\mathbf{BD})$$

Multiplicative decomposition (extendable to arbitrary number of matrices)

$$(\mathbf{A}_1 \otimes \mathbf{I}_2)(\mathbf{I}_1 \otimes \mathbf{A}_2) = (\mathbf{A}_1\mathbf{I}_1) \otimes (\mathbf{A}_2\mathbf{I}_2) = \mathbf{A}_1 \otimes \mathbf{A}_2$$

# Efficient evaluation of multi-variate B-splines

It follows form the multiplicative decomposition of the Kronecker product that

$$f(\xi,\eta,\zeta) = \left(\mathbf{b}^{d_1} \otimes \mathbf{b}^{d_2} \otimes \mathbf{b}^{d_3}\right) \cdot \mathbf{c} = \left(\mathbf{I}_1 \otimes \mathbf{I}_2 \otimes \mathbf{b}^{d_3}\right) \cdot \left(\mathbf{I}_1 \otimes \mathbf{b}^{d_2} \otimes \mathbf{I}_3\right) \cdot \left(\mathbf{b}^{d_1} \otimes \mathbf{I}_2 \otimes \mathbf{I}_3\right) \cdot \mathbf{c}$$

# Efficient evaluation of multi-variate B-splines

It follows form the multiplicative decomposition of the Kronecker product that

$$f(\xi,\eta,\zeta) = \left(\mathbf{b}^{d_1} \otimes \mathbf{b}^{d_2} \otimes \mathbf{b}^{d_3}\right) \cdot \mathbf{c} = \left(\mathbf{I}_1 \otimes \mathbf{I}_2 \otimes \mathbf{b}^{d_3}\right) \cdot \left(\mathbf{I}_1 \otimes \mathbf{b}^{d_2} \otimes \mathbf{I}_3\right) \cdot \left(\mathbf{b}^{d_1} \otimes \mathbf{I}_2 \otimes \mathbf{I}_3\right) \cdot \mathbf{c}$$

Algorithm 993 from [Fackler, 2019] with modifications

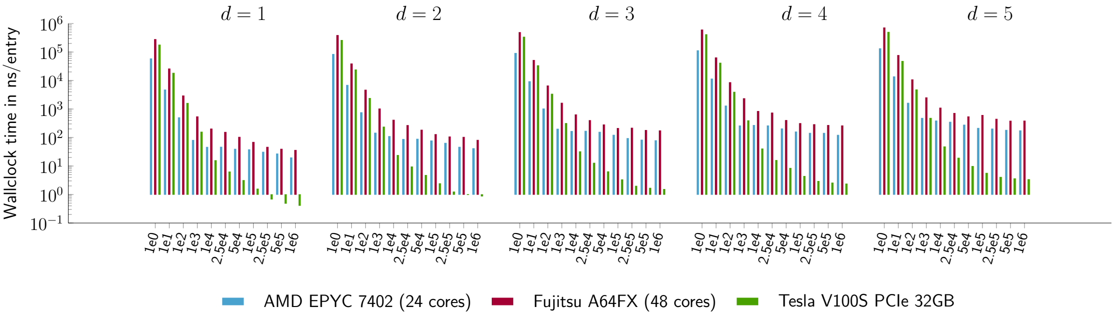
Set $\mathbf{f} := \mathbf{c}$
For $\ell = 1, 2, 3$

1. $\mathbf{f} := \text{reshape}(\mathbf{f}, [\cdot], d_\ell + 1)$
2. $\mathbf{f} := \mathbf{b}^{d_\ell} \cdot \mathbf{f}^\top$

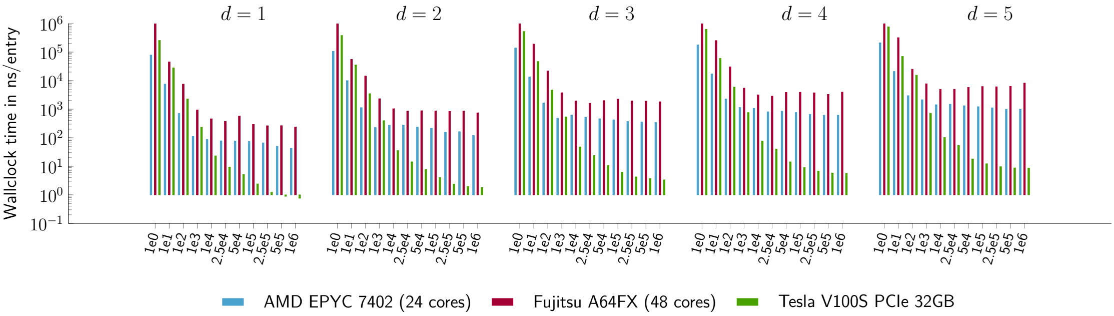
Output: $\mathbf{f} = f(\xi,\eta,\zeta)$

$$\mathbf{c} = \begin{array}{cc} \underbrace{\begin{bmatrix} 8 & 9 & 10 & 11 \\ 14 & 15 & 16 & 17 \\ 38 & 39 & 40 & 41 \\ \vdots & \vdots & \vdots & \vdots \\ 134 & 135 & 136 & 137 \end{bmatrix}}_{d_1+1 \text{ columns}} & \begin{array}{c} (d_2+1)(d_3+1) \\ \text{rows} \end{array} \end{array}$$

# Performance evaluation - bivariate B-splines



# Performance evaluation - trivariate B-splines



# Parametrization techniques

## Requirements (not just) for interactive modeling and analysis

1. Automatic creation of analysis-suitable parametrizations from boundary description

   *bivariate planar parametrizations, trivariate volumetric parametrizations*

2. Automatic reparametrization for improving the quality of parametrizations

   *as before* $+$ *bivariate surface parametrizations*

# Parametrization techniques

## Requirements (not just) for interactive modeling and analysis

1. Automatic creation of analysis-suitable parametrizations from boundary description

   *bivariate planar parametrizations, trivariate volumetric parametrizations*

2. Automatic reparametrization for improving the quality of parametrizations

   *as before + bivariate surface parametrizations*

## Parametrization techniques

1. Algebraic approaches, e.g., discrete Coons method
2. PDE-based approaches, e.g., $H^1$ and $H^2$ method
3. Optimization-based approaches, e.g., barrier or penalty function method

# Parametrization techniques

## Requirements (not just) for interactive modeling and analysis

1. Automatic creation of analysis-suitable parametrizations from boundary description

   *bivariate planar parametrizations, trivariate volumetric parametrizations*

2. Automatic reparametrization for improving the quality of parametrizations

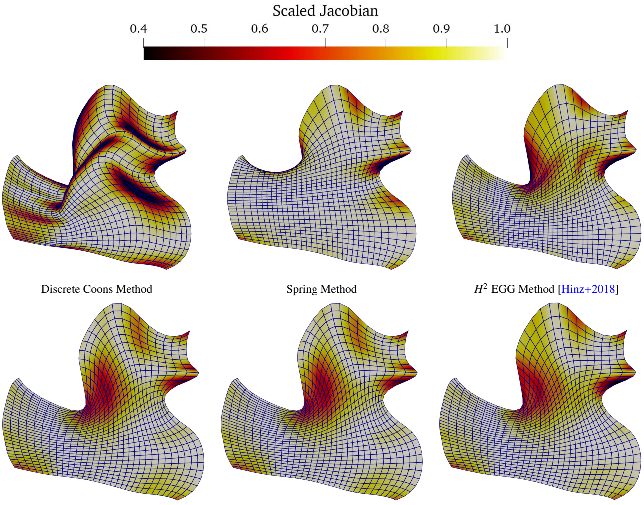
   *as before + bivariate surface parametrizations*

## Parametrization techniques

1. Algebraic approaches, e.g., discrete Coons method
2. PDE-based approaches, e.g., $H^1$ and $H^2$ method
3. Optimization-based approaches, e.g., barrier or penalty function method

Our workflow: init by Coons patch $\Rightarrow$ a.s. parametrization by $H^1$ $\Rightarrow$ optimize if needed

# Comparison of the different approaches




Discrete Coons Method

Spring Method

$H^2$ EGG Method [Hinz+2018]

Barrier Function-Based Method [Ji+2021]

Penalty Function-Based Method [Ji+2022]

$H^1$ Scaled EGG Method [Ji+2023]

Results by Ye Ji

# Automatic placement of interior control points

**Harmonic mapping**: $\mathbf{x} : \hat{\Omega} \to \Omega$ by solving

$$\begin{aligned} \nabla \cdot \nabla \xi(x,y) &= 0 \\ \nabla \cdot \nabla \eta(x,y) &= 0 \end{aligned} \qquad \text{such that } \mathbf{x}^{-1}|_{\Gamma} = \hat{\Gamma}$$

- $\mathbf{x}^{-1}$ exists and is unique if the curvature of $\hat{\Omega}$ is non-positive and the boundary $\hat{\Gamma}$ when considered with respect to the metric on $\Omega$ is convex [Eells and Lemaire, 1978]
- $\mathbf{x}^{-1}$ is one-to-one by the Radó-Kneser-Choquet theorem [Duren and Hengartner, 1997]

# Automatic placement of interior control points cont'd

**Weak form** in $H^2$ [Hinz et al., 2020]

$$\begin{aligned} \int_{\hat{\Omega}} \mathbf{w} \tilde{\mathcal{L}} x \, \mathrm{d}\hat{\Omega} &= \mathbf{0} \\ \int_{\hat{\Omega}} \mathbf{w} \tilde{\mathcal{L}} y \, \mathrm{d}\hat{\Omega} &= \mathbf{0} \end{aligned} \qquad \text{such that } \mathbf{x}^{-1}|_{\Gamma} = \hat{\Gamma}$$

where

$$\tilde{\mathcal{L}} = \left( g_{22} \frac{\partial^2}{\partial \xi^2} - 2 g_{12} \frac{\partial^2}{\partial \xi \partial \eta} + g_{11} \frac{\partial^2}{\partial \eta^2} \right) / \left( g_{11} + g_{22} \right)$$

# Automatic placement of interior control points cont'd

**Weak form** in $H^2$ [Hinz et al., 2020]

$$\begin{aligned} \int_{\hat\Omega} \mathbf{w}\tilde{\mathcal{L}}x \, \mathrm{d}\hat\Omega &= \mathbf{0} \\ \int_{\hat\Omega} \mathbf{w}\tilde{\mathcal{L}}y \, \mathrm{d}\hat\Omega &= \mathbf{0} \end{aligned} \qquad \text{such that } \mathbf{x}^{-1}|_\Gamma = \hat\Gamma$$

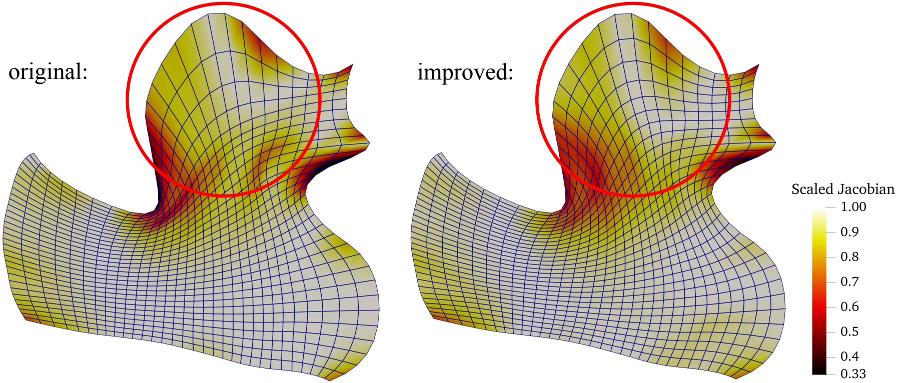
where

$$\tilde{\mathcal{L}} = \left( g_{22}\frac{\partial^2}{\partial\xi^2} - 2g_{12}\frac{\partial^2}{\partial\xi\partial\eta} + g_{11}\frac{\partial^2}{\partial\eta^2} \right) / \left(g_{11} + g_{22}\right)$$

New **weak form** in $H^1$ [Ji et al., 2023]

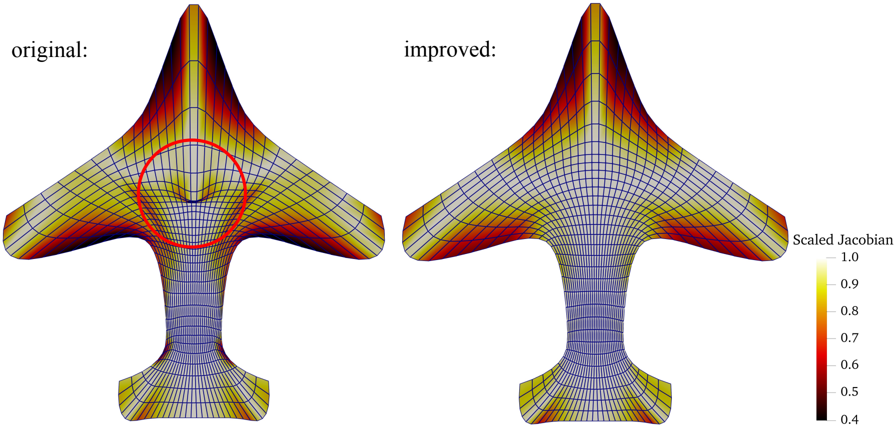
$$\begin{aligned} \int_{\hat\Omega} \nabla_{\mathbf{x}}\mathbf{w} \cdot \nabla_{\mathbf{x}}\xi \, \mathrm{d}\hat\Omega &= \mathbf{0} \\ \int_{\hat\Omega} \nabla_{\mathbf{x}}\mathbf{w} \cdot \nabla_{\mathbf{x}}\eta \, \mathrm{d}\hat\Omega &= \mathbf{0} \end{aligned} \qquad \text{such that } \mathbf{x}^{-1}|_\Gamma = \hat\Gamma$$

# Comparison between $H^1$ and $H^2$ approaches



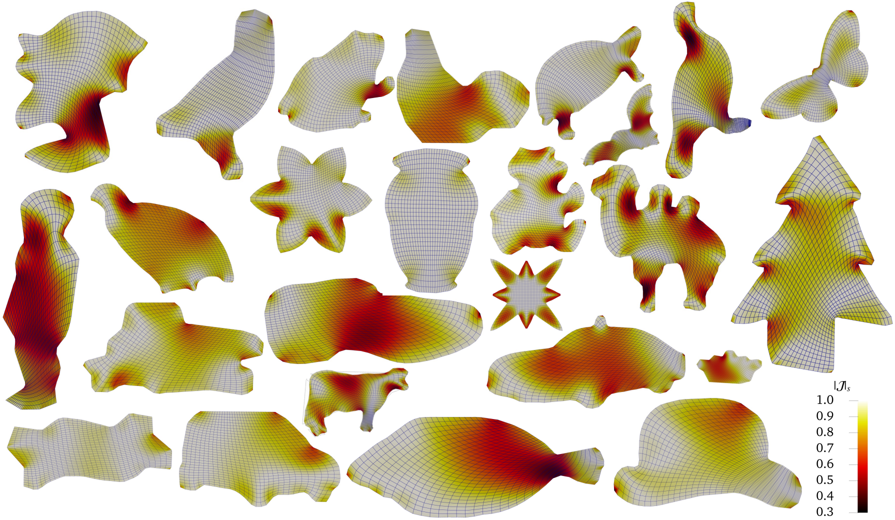

Results by Ye Ji

# Comparison between $H^1$ and $H^2$ approaches



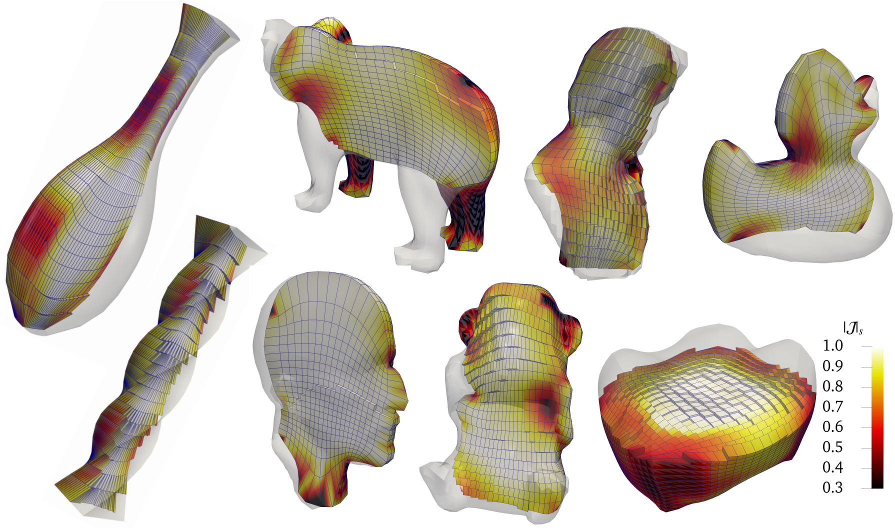

Results by Ye Ji

# Planar results



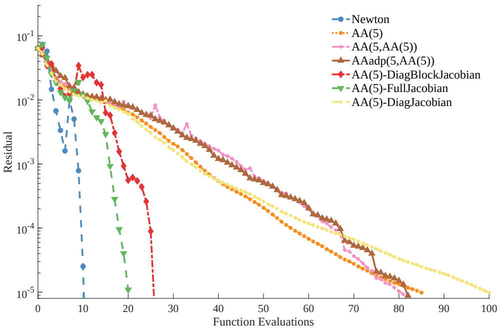

Results by Ye Ji

# Volumetric results

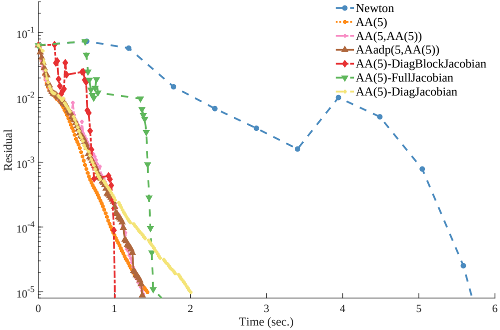



Results by Ye Ji

# Solution of nonlinear systems by preconditioned Anderson acceleration




function evaluations




wallclock time

Results from Ji et al. [2023]

# Summary and outlook

- Least-squares collocation IgA enables seamless in-paradigm blending between fast learning-based pre-analysis and in-depth simulation-based (post-)analysis
- Theory from IgA-L carries over to NN (e.g., interpretation of loss function)
- Iterative refinement of NN's output by 'classical' IgA-L is possible

What's next?

- Interactive workflow `https://visualization.surf.nl/iganet/`
- Least-squares collocation-based parametrization techniques
- Extension to multi-patch parametrizations

# Bridging the gap between isogeometric analysis and deep operator learning

Matthias Möller

Department of Applied Mathematics, TU Delft, The Netherlands

Seminar at Dipartimento di Matematica @ Università di Pavia, February 6, 2024

Joint work with Deepesh Toshniwal, Frank van Ruiten (TU Delft), Ye Ji, Mengyun Wang (TU Delft, Dalian), Casper van Leeuwen, Paul Melis (SURF), and Jaewook Lee (TU Vienna)

Thank you very much!

# References I

C. de Boor. Subroutine package for calculating with B-splines. (LA-4728), 1 1971. doi: 10.2172/4740859. URL https://www.osti.gov/biblio/4740859.

P. L. Duren and W. Hengartner. Harmonic mappings of multiply connected domains. *Pacific Journal of Mathematics*, 180:201–220, 1997. URL https://api.semanticscholar.org/CorpusID:121912227.

J. Eells and L. Lemaire. A report on harmonic maps. *Bulletin of the London Mathematical Society*, 10(1):1–68, Mar. 1978. doi: 10.1112/blms/10.1.1. URL https://doi.org/10.1112/blms/10.1.1.

P. L. Fackler. Algorithm 993. *ACM Transactions on Mathematical Software*, 45(2):1–9, May 2019. doi: 10.1145/3291041. URL https://doi.org/10.1145/3291041.

J. Hinz, M. Möller, and C. Vuik. An IGA framework for PDE-based planar parameterization on convex multipatch domains. In *Lecture Notes in Computational Science and Engineering*, pages 57–75. Springer International Publishing, Aug. 2020. doi: 10.1007/978-3-030-49836-8_4. URL https://doi.org/10.1007/978-3-030-49836-8_4.

# References II

Y. Ji, K. Chen, M. Möller, and C. Vuik. On an improved PDE-based elliptic parameterization method for isogeometric analysis using preconditioned anderson acceleration. *Computer Aided Geometric Design*, 102:102191, May 2023. doi: 10.1016/j.cagd.2023.102191. URL https://doi.org/10.1016/j.cagd.2023.102191.

H. Lin, Y. Xiong, X. Wang, Q. Hu, and J. Ren. Isogeometric least-squares collocation method with consistency and convergence analysis. *Journal of Systems Science and Complexity*, 33(5):1656–1693, Oct. 2020. doi: 10.1007/s11424-020-9052-9. URL https://doi.org/10.1007/s11424-020-9052-9.

T. Lyche and K. Mørken. Lecture notes: Spline methods, 2018. URL https://www.uio.no/studier/emner/matnat/math/MAT4170/v18/pensumliste/splinebook-2018.pdf.

J. R. R. A. Martins and A. Ning. *Engineering Design Optimization -*. Cambridge University Press, Cambridge, 2021. ISBN 978-1-108-98861-2.

# References III

M. Montardini, G. Sangalli, and L. Tamellini. Optimal-order isogeometric collocation at galerkin superconvergent points. *Computer Methods in Applied Mechanics and Engineering*, 316:741–757, Apr. 2017. doi: 10.1016/j.cma.2016.09.043. URL `https://doi.org/10.1016/j.cma.2016.09.043`.