IgaNets: Physics-Informed Machine Learning Embedded Into Isogeometric Analysis

Matthias Möller, Deepesh Toshniwal, Frank van Ruiten

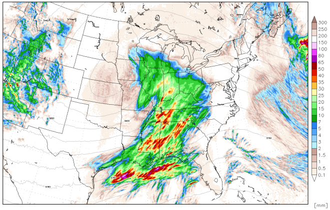
Numerical Analysis, Delft Institute of Applied Mathematics

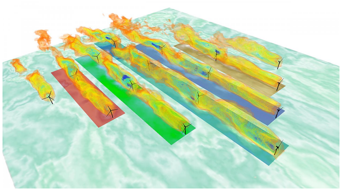
EEMCS, Delft University of Technology

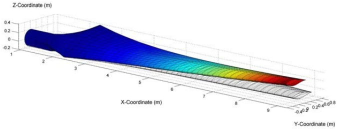
NMC Scientific Days

19 & 20 April 2022

Numerics for PDE analysis





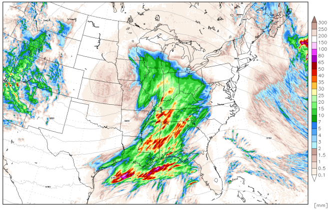


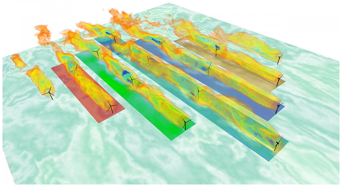
Many physical processes are modelled mathematically by (systems of) PDEs that require fast & accurate numerical methods to compute approximate solutions:

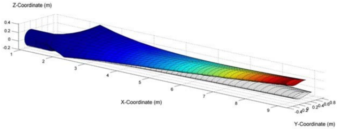
- particle methods: PIC (1955), SPH (1977), DPD (1992), RKPM (1995), ...
- hybrid particle-mesh methods: MPM (1990s), ...
- mesh-based methods: FEM (1940s), FDM (1950s), FVM (1971), IGA (2005), ...

Credit: `www.superzelle.de` – Janek Zimmer; University of Texas at Dallas (DOI: 10.1063/5.0036640); University of Minnesota – Eolos Wind Energy Research

Numerics for PDE analysis







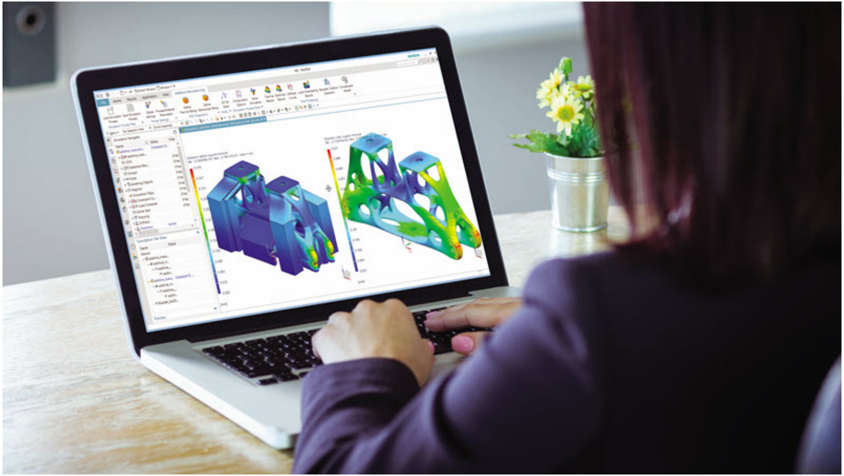
Many physical processes are modelled mathematically by (systems of) PDEs that require fast & accurate numerical methods to compute approximate solutions:

- particle methods: PIC (1955), SPH (1977), DPD (1992), RKPM (1995), ...
- hybrid particle-mesh methods: MPM (1990s), ...
- mesh-based methods: FEM (1940s), FDM (1950s), FVM (1971), IGA (2005), ...

How fast is fast? And is it just about analysis?

Credit: www.superzelle.de – Janek Zimmer; University of Texas at Dallas (DOI: 10.1063/5.0036640); University of Minnesota – Eolos Wind Energy Research

Design through Analysis

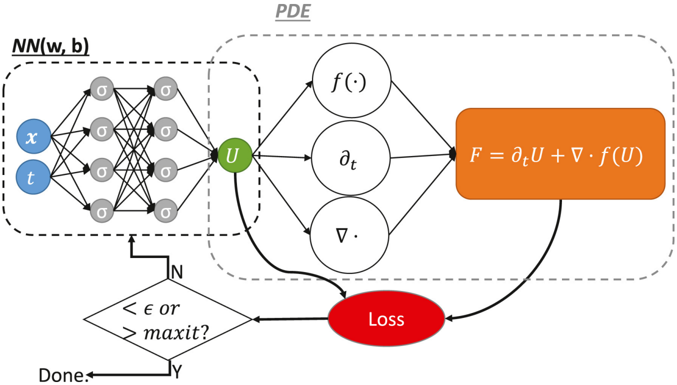


We want it all: from really fast & moderately accurate to moderately fast & highly accurate!

Credit: Siemens – Simulation for Design Engineers

SciML for PDE analysis

- Physics-informed neural networks (PINNs) [Raissi, Perdikaris, Karniadakis, 2019]



SciML for PDE analysis

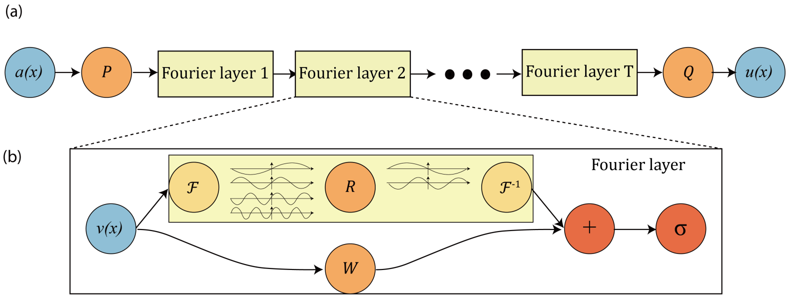
- Physics-informed neural networks (PINNs) [Raissi, Perdikaris, Karniadakis, 2019]

  + No pre-calculated data needed (unsupervised learning)
  + Can be augmented with data (faster decay of loss function)
  + Applicable to arbitrary PDEs (extra effort might be needed to impose 'physics')

  – Convergence theory is in its infancy (different from classical numerical methods theory)
  – Poor extrapolation capabilities (different geometries, problem parameters)
  – Space-time treatment of time-dependent problems

SciML for PDE analysis

- Physics-informed neural networks (PINNs) [Raissi, Perdikaris, Karniadakis, 2019]
- Fourier neural operators (FNO) [Li, Kovachki, Azizzadenesheli, Liu, Bhattacharya, Stuart, Anandkumar, 2020]



SciML for PDE analysis

- Physics-informed neural networks (PINNs) [Raissi, Perdikaris, Karniadakis, 2019]
- Fourier neural operators (FNO) [Li, Kovachki, Azizzadenesheli, Liu, Bhattacharya, Stuart, Anandkumar, 2020]

  + Aims to learn the operator (not the PDE problem)

  – Pre-calculated training data is needed (supervised learning)
  – Assumes an efficient Fourier approximation of the solution
  – Designed for time-dependent PDEs

SciML for PDE analysis

- Physics-informed neural networks (PINNs) [Raissi, Perdikaris, Karniadakis, 2019]
- Fourier neural operators (FNO) [Li, Kovachki, Azizzadenesheli, Liu, Bhattacharya, Stuart, Anandkumar, 2020]
- Learning nonlinear operators (DeepONets) [Lu, Jin, Pang, Zhang, Karniadakis, 2021]

$$G_\theta(u)(y) = \sum_{k=1}^{q} \underbrace{b_k(u(x_1), u(x_2), \ldots, u(x_m))}_{\text{branch}} \underbrace{t_k(y)}_{\text{trunk}}$$

\+ Aims to learn the operator (not the PDE problem)
\+ Claims to have excellent extrapolation capabilities

– Pre-calculated training data is needed (supervised learning)
– Designed for time-dependent PDEs

SciML for PDE analysis

- Physics-informed neural networks (PINNs) [Raissi, Perdikaris, Karniadakis, 2019]
- Fourier neural operators (FNO) [Li, Kovachki, Azizzadenesheli, Liu, Bhattacharya, Stuart, Anandkumar, 2020]
- Learning nonlinear operators (DeepONets) [Lu, Jin, Pang, Zhang, Karniadakis, 2021]

$$G_\theta(u)(y) = \sum_{k=1}^{q} \underbrace{b_k(u(x_1), u(x_2), \ldots, u(x_m))}_{\text{branch}} \underbrace{t_k(y)}_{\text{trunk}}$$

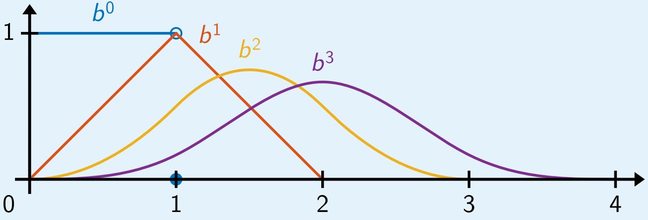
\+ Aims to learn the operator (not the PDE problem)
\+ Claims to have excellent extrapolation capabilities

– Pre-calculated training data is needed (supervised learning)
– Designed for time-dependent PDEs

Combine mesh-based numerics with SciML for PDE analysis

Isogeometric Analysis

B-spline basis functions



knot vector $\Xi = [0, 1, 2, 3, 4]$

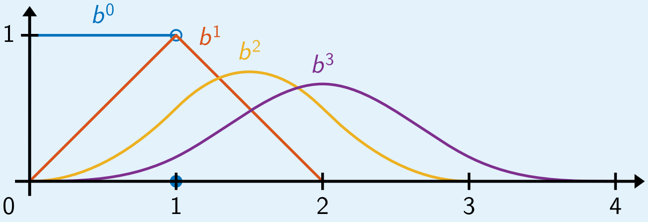
$$b_\ell^0(\xi) = \begin{cases} 1 & \text{if } \xi_\ell \leq \xi < \xi_{\ell+1} \\ 0 & \text{otherwise} \end{cases}$$

$$b_\ell^p(\xi) = \frac{\xi - \xi_\ell}{\xi_{\ell+p} - \xi_\ell} b_\ell^{p-1}(\xi) + \frac{\xi_{\ell+p+1} - \xi}{\xi_{\ell+p+1} - \xi_{\ell+1}} b_{\ell+1}^{p-1}(\xi)$$

T.J.R. Hughes, J.A.Cottrell, Y.Bazilevs: Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. CMAME 194, 2005.

Isogeometric Analysis

B-spline basis functions



knot vector $\Xi = [0, 1, 2, 3, 4]$

$$b_\ell^0(\xi) = \begin{cases} 1 & \text{if } \xi_\ell \le \xi < \xi_{\ell+1} \\ 0 & \text{otherwise} \end{cases}$$

$$b_\ell^p(\xi) = \frac{\xi - \xi_\ell}{\xi_{\ell+p} - \xi_\ell} b_\ell^{p-1}(\xi) + \frac{\xi_{\ell+p+1} - \xi}{\xi_{\ell+p+1} - \xi_{\ell+1}} b_{\ell+1}^{p-1}(\xi)$$

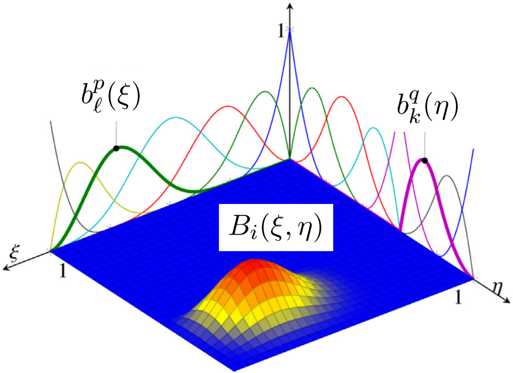
Many good properties: compact support $[\xi_\ell, \xi_{\ell+p+1})$, positive function values over support interval, derivatives of B-splines are combinations of lower-order B-splines, ...

T.J.R. Hughes, J.A.Cottrell, Y.Bazilevs: Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. CMAME 194, 2005.

Isogeometric Analysis

Paradigm: represent 'everything' in terms of tensor products of B-spline basis functions

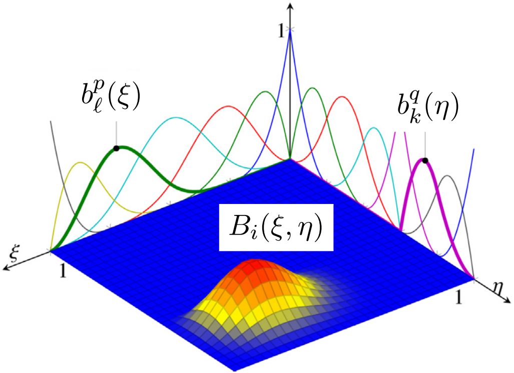
$$B_i(\xi,\eta) := b_\ell^p(\xi)\cdot b_k^q(\eta), \qquad i := (k-1)\cdot n_\ell + \ell, \quad 1\le \ell \le n_\ell, \quad 1 \le k \le n_k,$$



Isogeometric Analysis

Paradigm: represent 'everything' in terms of tensor products of B-spline basis functions

$$B_i(\xi,\eta) := b_\ell^p(\xi)\cdot b_k^q(\eta), \qquad i := (k-1)\cdot n_\ell + \ell, \quad 1\le \ell \le n_\ell, \quad 1 \le k \le n_k,$$

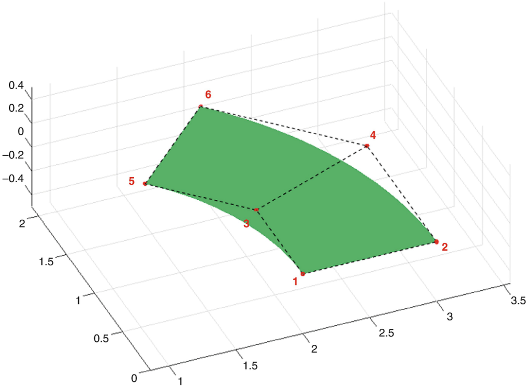


Many more good properties: partition of unity $\sum_{i=1}^{n} B_i(\xi,\eta) \equiv 1$, C^{p-1} continuity, ...

Isogeometric Analysis

Geometry: bijective mapping from the unit square to the physical domain $\Omega_h \subset \mathbb{R}^d$

$$\mathbf{x}_h(\xi,\eta) = \sum_{i=1}^{n} B_i(\xi,\eta) \cdot \mathbf{x}_i \qquad \forall(\xi,\eta) \in [0,1]^2 =: \hat{\Omega}$$

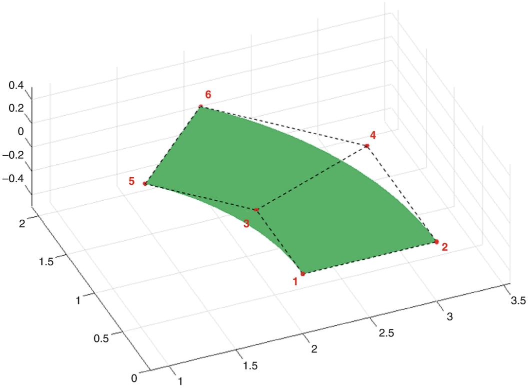


- the shape of Ω_h is fully specified by the set of **control points** $\mathbf{x}_i \in \mathbb{R}^d$

Isogeometric Analysis

Geometry: bijective mapping from the unit square to the physical domain $\Omega_h \subset \mathbb{R}^d$

$$\mathbf{x}_h(\xi,\eta) = \sum_{i=1}^{n} B_i(\xi,\eta) \cdot \mathbf{x}_i \qquad \forall(\xi,\eta) \in [0,1]^2 =: \hat{\Omega}$$

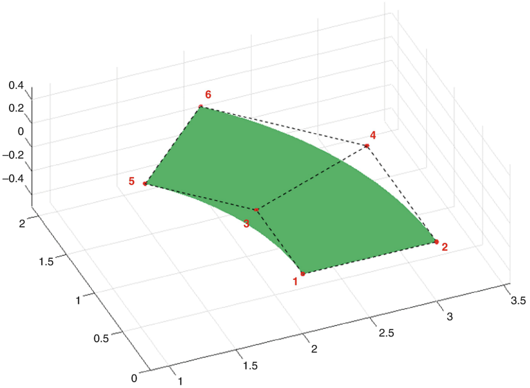


- the shape of Ω_h is fully specified by the set of **control points** $\mathbf{x}_i \in \mathbb{R}^d$
- interior control points must be chosen such that 'grid lines' do not fold as this violates the bijectivity of $\mathbf{x}_h : \hat{\Omega} \to \Omega_h$

Isogeometric Analysis

Geometry: bijective mapping from the unit square to the physical domain $\Omega_h \subset \mathbb{R}^d$

$$\mathbf{x}_h(\xi,\eta) = \sum_{i=1}^{n} B_i(\xi,\eta) \cdot \mathbf{x}_i \qquad \forall(\xi,\eta) \in [0,1]^2 =: \hat{\Omega}$$



- the shape of Ω_h is fully specified by the set of **control points** $\mathbf{x}_i \in \mathbb{R}^d$
- interior control points must be chosen such that 'grid lines' do not fold as this violates the bijectivity of $\mathbf{x}_h : \hat{\Omega} \to \Omega_h$
- refinement in h (knot insertion) and p (order elevation) preserves the shape of Ω_h and can be used to generate finer computational 'grids' for the analysis

Isogeometric Analysis

Data, boundary conditions, and solution: forward mappings from the unit square

$$\text{(r.h.s vector)} \qquad f_h \circ \mathbf{x}_h(\xi,\eta) = \sum_{i=1}^{n} B_i(\xi,\eta) \cdot f_i \qquad \forall(\xi,\eta) \in [0,1]^2$$

$$\text{(boundary conditions)} \qquad g_h \circ \mathbf{x}_h(\xi,\eta) = \sum_{i=1}^{n} B_i(\xi,\eta) \cdot g_i \qquad \forall(\xi,\eta) \in \partial[0,1]^2$$

$$\text{(solution)} \qquad u_h \circ \mathbf{x}_h(\xi,\eta) = \sum_{i=1}^{n} B_i(\xi,\eta) \cdot u_i \qquad \forall(\xi,\eta) \in [0,1]^2$$

Isogeometric Analysis

Data, boundary conditions, and solution: forward mappings from the unit square

$$\text{(r.h.s vector)} \qquad f_h \circ \mathbf{x}_h(\xi,\eta) = \sum_{i=1}^{n} B_i(\xi,\eta) \cdot f_i \qquad \forall(\xi,\eta) \in [0,1]^2$$

$$\text{(boundary conditions)} \qquad g_h \circ \mathbf{x}_h(\xi,\eta) = \sum_{i=1}^{n} B_i(\xi,\eta) \cdot g_i \qquad \forall(\xi,\eta) \in \partial[0,1]^2$$

$$\text{(solution)} \qquad u_h \circ \mathbf{x}_h(\xi,\eta) = \sum_{i=1}^{n} B_i(\xi,\eta) \cdot u_i \qquad \forall(\xi,\eta) \in [0,1]^2$$

Model problem: Poisson's equation

$$-\Delta u_h = f_h \quad \text{in} \quad \Omega_h, \qquad u_h = g_h \quad \text{on} \quad \partial\Omega_h$$

Isogeometric Analysis

Different solution approaches

- Galerkin-type IGA (Hughes *et al.* 2005 and many more)
- Isogeometric collocation methods (Reali, Hughes, 2015)
- Variational collocation method (Gomez, De Lorenzis, 2016)

Isogeometric Analysis

Different solution approaches

- Galerkin-type IGA (Hughes *et al.* 2005 and many more)
- Isogeometric collocation methods (Reali, Hughes, 2015)
- Variational collocation method (Gomez, De Lorenzis, 2016)

Abstract representation

Given $\mathbf{x}_i$ (geometry), f_i (r.h.s. vector), and g_i (boundary conditions), **compute**

$$\begin{bmatrix} u_1 \\ \vdots \\ u_n \end{bmatrix} = A^{-1}\left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_n \end{bmatrix}, \begin{bmatrix} g_1 \\ \vdots \\ g_n \end{bmatrix}\right) \cdot b\left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_n \end{bmatrix}, \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix}, \begin{bmatrix} g_1 \\ \vdots \\ g_n \end{bmatrix}\right)$$

Any point of the solution can afterwards be obtained by a simple **function evaluation**

$$(\xi,\eta) \in [0,1]^2 \quad \mapsto \quad u_h \circ \mathbf{x}_h(\xi,\eta) = [B_1(\xi,\eta),\ldots,B_n(\xi,\eta)] \cdot \begin{bmatrix} u_1 \\ \vdots \\ u_n \end{bmatrix}$$

Isogeometric Analysis

Abstract representation

Given $\mathbf{x}_i$ (geometry), f_i (r.h.s. vector), and g_i (boundary conditions), **compute**

$$\begin{bmatrix} u_1 \\ \vdots \\ u_n \end{bmatrix} = A^{-1}\left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_n \end{bmatrix}, \begin{bmatrix} g_1 \\ \vdots \\ g_n \end{bmatrix}\right) \cdot b\left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_n \end{bmatrix}, \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix}, \begin{bmatrix} g_1 \\ \vdots \\ g_n \end{bmatrix}\right)$$

Any point of the solution can afterwards be obtained by a simple **function evaluation**

$$(\xi, \eta) \in [0,1]^2 \quad \mapsto \quad u_h \circ \mathbf{x}_h(\xi, \eta) = [B_1(\xi, \eta), \dots, B_n(\xi, \eta)] \cdot \begin{bmatrix} u_1 \\ \vdots \\ u_n \end{bmatrix}$$

Let us interpret the sets of B-spline coefficients $\{\mathbf{x}_i\}$, $\{f_i\}$, and $\{g_i\}$ as an efficient encoding of our PDE problem that is fed into our IGA machinery as **input**.
The **output** of our IGA machinery are the B-spline coefficients $\{u_i\}$ of the solution.

Isogeometric Analysis + PINNs

IgaNet: replace **computation** by **physics-informed machine learning**

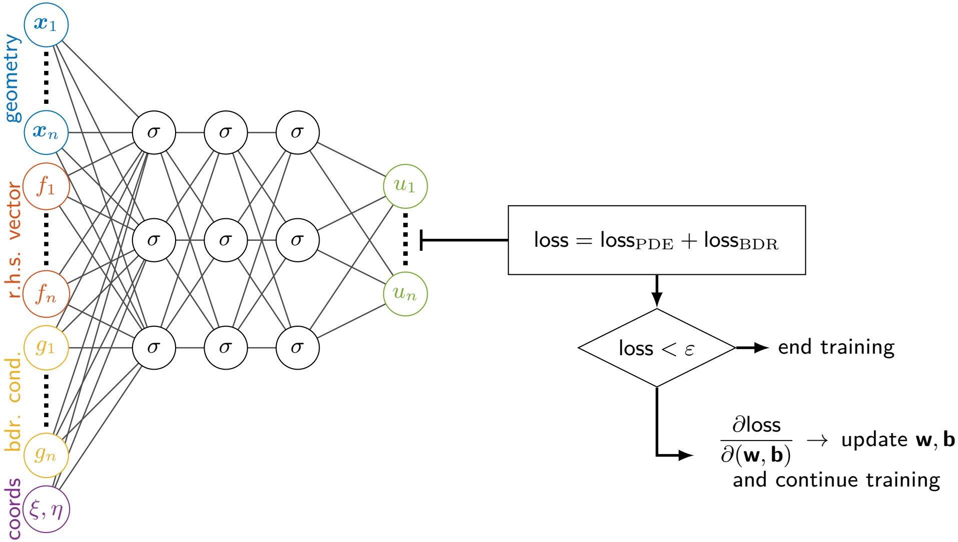
$$\begin{bmatrix} u_1 \\ \vdots \\ u_n \end{bmatrix} = A^{-1}\left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_n \end{bmatrix}, \begin{bmatrix} g_1 \\ \vdots \\ g_n \end{bmatrix}\right) \cdot b\left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_n \end{bmatrix}, \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix}, \begin{bmatrix} g_1 \\ \vdots \\ g_n \end{bmatrix}\right)$$

$$\begin{bmatrix} u_1 \\ \vdots \\ u_n \end{bmatrix} = \text{PINN}\left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_n \end{bmatrix}, \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix}, \begin{bmatrix} g_1 \\ \vdots \\ g_n \end{bmatrix}; (\xi_k, \eta_k)_{k=1}^{N_\text{samples}}\right)$$

Compute the solution by evaluating the trained neural network

$$u_h(\xi, \eta) \approx [B_1(\xi, \eta), \dots, B_n(\xi, \eta)] \cdot \begin{bmatrix} u_1 \\ \vdots \\ u_n \end{bmatrix} = \text{PINN}\left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_n \end{bmatrix}, \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix}, \begin{bmatrix} g_1 \\ \vdots \\ g_n \end{bmatrix}; (\xi, \eta)\right)$$

IgaNet architecture



Loss function

$$\text{loss}_{\mathrm{PDE}} = \frac{\alpha}{N_\Omega} \sum_{k=1}^{N_\Omega} \left|\Delta[u_h \circ \mathbf{x}_h(\xi_k, \eta_k)] - f_h \circ \mathbf{x}_h(\xi_k, \eta_k)\right|^2$$

$$\text{loss}_{\mathrm{BDR}} = \frac{\beta}{N_\Gamma} \sum_{k=1}^{N_\Gamma} \left|u_h \circ \mathbf{x}_h(\xi_k, \eta_k) - g_h \circ \mathbf{x}_h(\xi_k, \eta_k)\right|^2$$

Express derivatives with respect to physical space variables using the Jacobian J, the Hessian H and the matrix of squared first derivatives Q [Schillinger *et al.* 2013]:

$$\begin{bmatrix} \frac{\partial^2 B}{\partial x^2} \\ \frac{\partial^2 B}{\partial x \partial y} \\ \frac{\partial^2 B}{\partial y^2} \end{bmatrix} = Q^{-\top} \left(\begin{bmatrix} \frac{\partial^2 B}{\partial \xi^2} \\ \frac{\partial^2 B}{\partial \xi \partial \eta} \\ \frac{\partial^2 B}{\partial \eta^2} \end{bmatrix} - H^\top J^{-\top} \begin{bmatrix} \frac{\partial B}{\partial \xi} \\ \frac{\partial B}{\partial \eta} \end{bmatrix} \right)$$

Two-level training strategy

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

For a batch of randomly sampled $(\xi_k, \eta_k) \in [0,1]^2$ **do**

$$\text{Train PINN}\left(\begin{bmatrix}\mathbf{x}_1\\ \vdots \\ \mathbf{x}_n\end{bmatrix}, \begin{bmatrix}f_1\\ \vdots \\ f_n\end{bmatrix}, \begin{bmatrix}g_1\\ \vdots \\ g_n\end{bmatrix}; (\xi_k, \eta_k)_{k=1}^{N_{\text{samples}}}\right) \mapsto \begin{bmatrix}u_1\\ \vdots \\ u_n\end{bmatrix}$$

EndFor

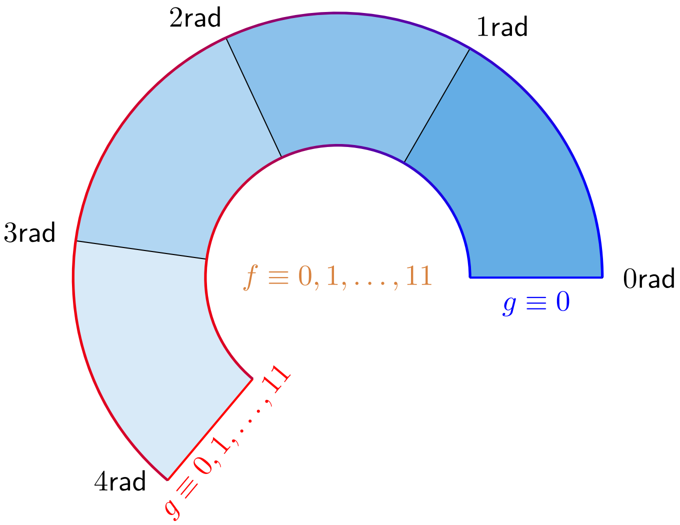
EndFor

IGA details: 7×7 bi-cubic tensor-product B-splines for $\mathbf{x}_h$ and u_h, C^2-continuous

PINN details: TensorFlow 2.6, 7-layer neural network with 50 neurons per layer and ReLU activation function (except for output layer), Adam optimizer, 30.000 epochs, training is stopped after 3.000 epochs w/o improvement of the loss value

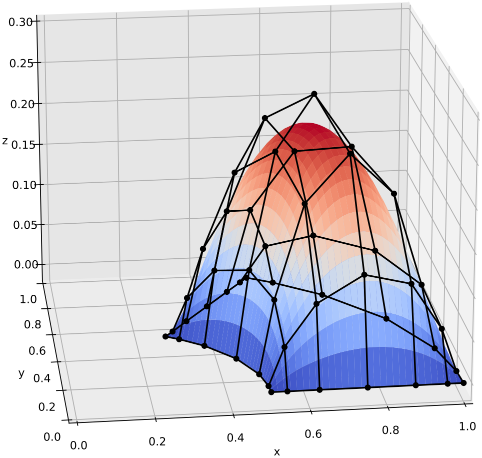
Ongoing master thesis work of Frank van Ruiten, TU Delft

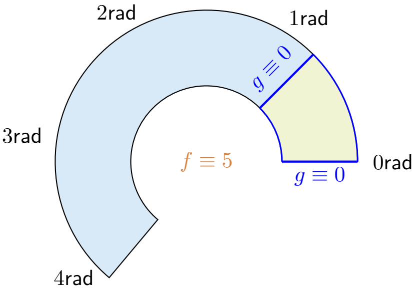
Test case: Poisson's equation on a variable annulus



Ongoing master thesis work of Frank van Ruiten, TU Delft

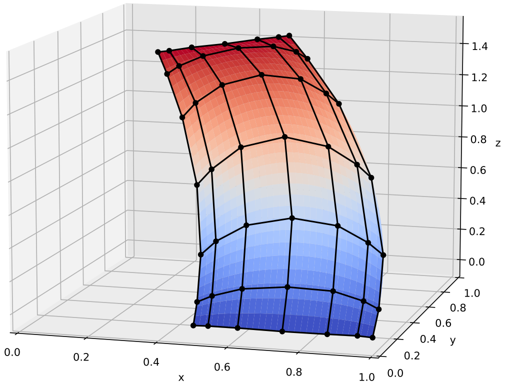
Preliminary results

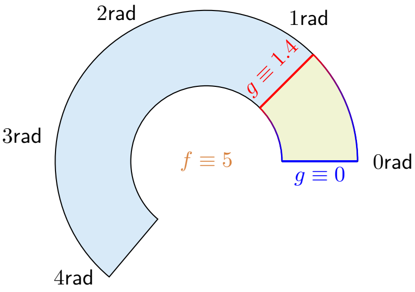




Ongoing master thesis work of Frank van Ruiten, TU Delft

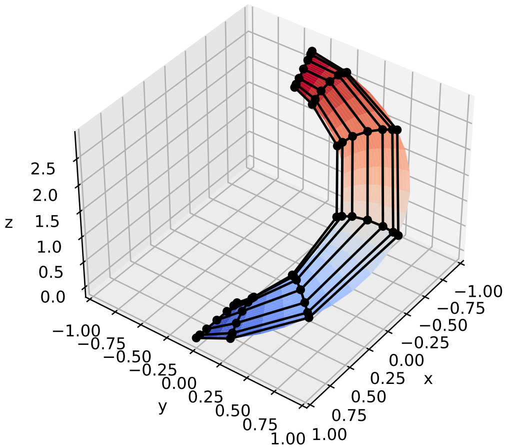
Preliminary results

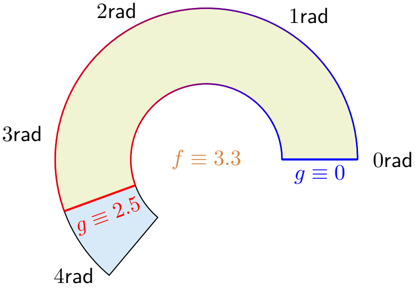




Ongoing master thesis work of Frank van Ruiten, TU Delft

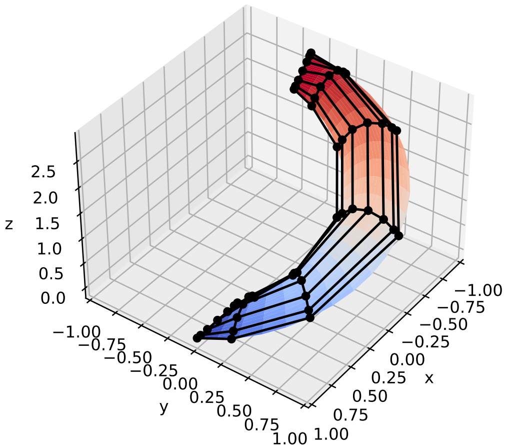
Preliminary results

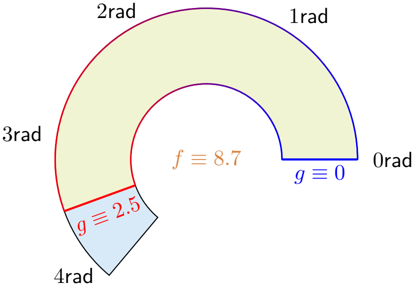




Ongoing master thesis work of Frank van Ruiten, TU Delft

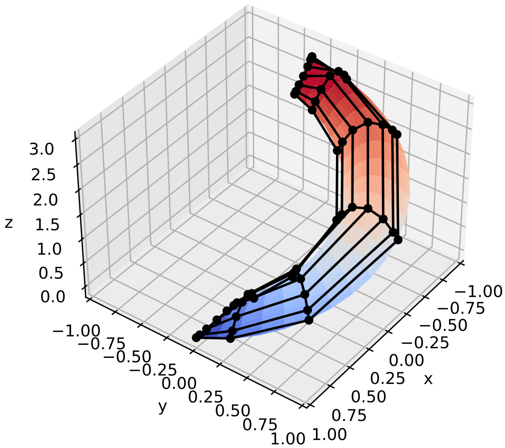
Preliminary results

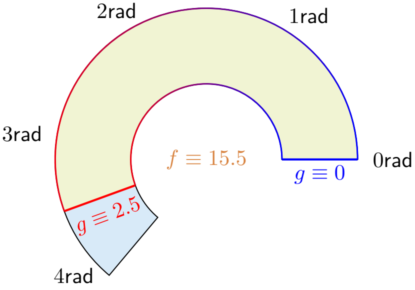




Ongoing master thesis work of Frank van Ruiten, TU Delft

Preliminary results





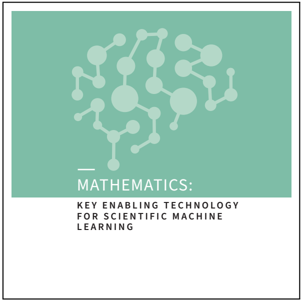
Ongoing master thesis work of Frank van Ruiten, TU Delft

Conclusion and outlook

IgaNets combine classical numerics with scientific machine learning and may finally enable integrated and interactive computer-aided **design-through-analysis** workflows

Todo

- performance and hyper-parameter tuning
- extension to multi-patch topologies
- use of IGA and IgaNets in concert
- transfer learning upon basis refinement



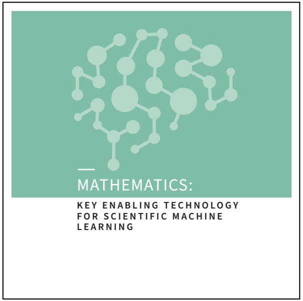
Short paper: Möller, Toshniwal, van Ruiten: *Physics-informed machine learning embedded into isogeometric analysis*, 2021.

Conclusion and outlook

IgaNets combine classical numerics with scientific machine learning and may finally enable integrated and interactive computer-aided **design-through-analysis** workflows

Todo

- performance and hyper-parameter tuning
- extension to multi-patch topologies
- use of IGA and IgaNets in concert
- transfer learning upon basis refinement



Short paper: Möller, Toshniwal, van Ruiten: *Physics-informed machine learning embedded into isogeometric analysis*, 2021. ☞

We are hiring! AIO position will open soon! Thank you for your attention!