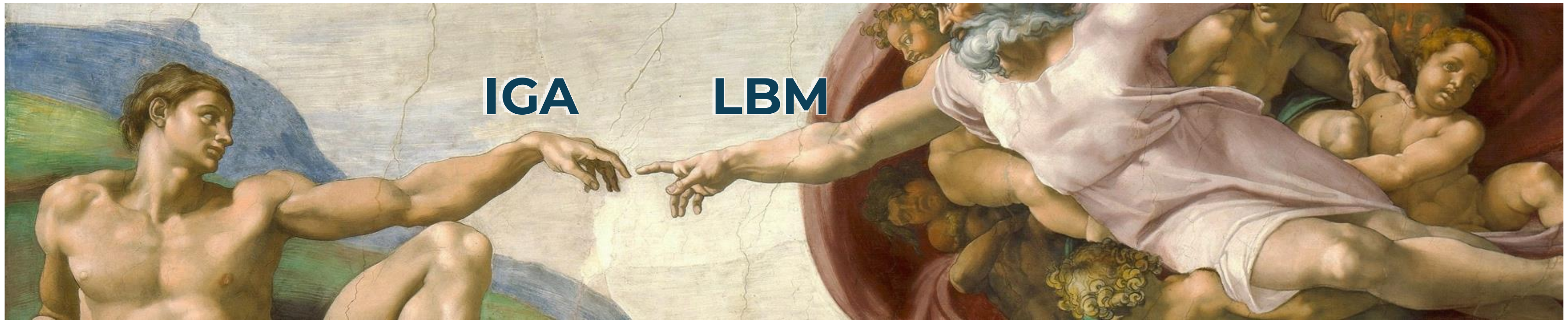


An isogeometric formulation of the lattice Boltzmann method on CAD-exact geometries

WCCM–ECCOMAS 2026 · Munich, Germany · 19–24 July 2026
MS091A: The Lattice Boltzmann method as a general-purpose tool in Computational Mechanics I

Ye Ji, Monica Lacatus & Matthias Möller
Department of Applied Mathematics
Delft University of Technology, The Netherlands

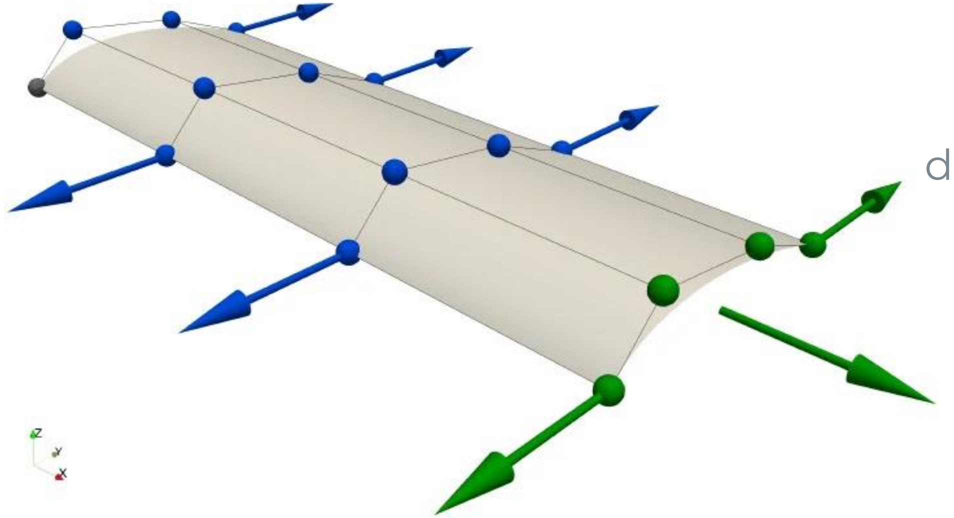
Outline

Outline

1. **Background and motivation**
2. Isogeometric lattice Boltzmann method
3. Numerical examples and applications to FSI
4. Conclusions and outlook

Geometry is LBM's missing piece

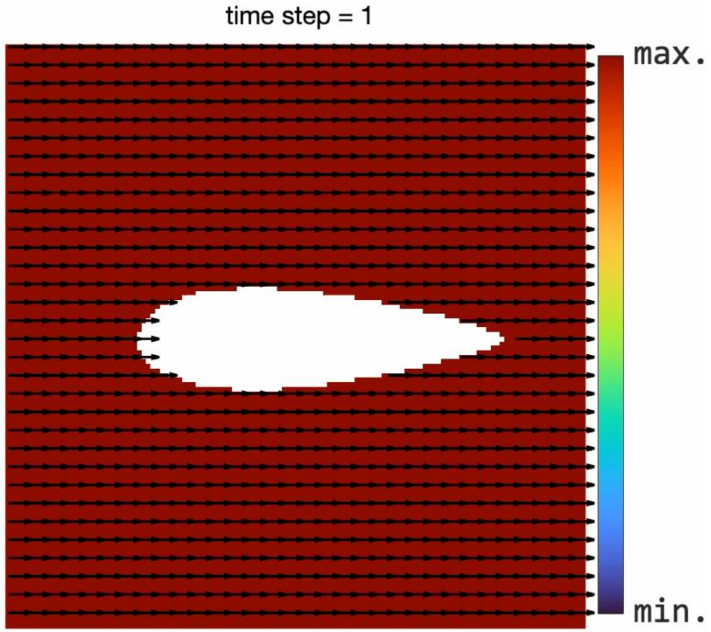
- LBM is **fast**, **scalable**, and **general-purpose**;
- But its **Cartesian grid can't represent curved geometry exactly**.



CAD-exact geometry

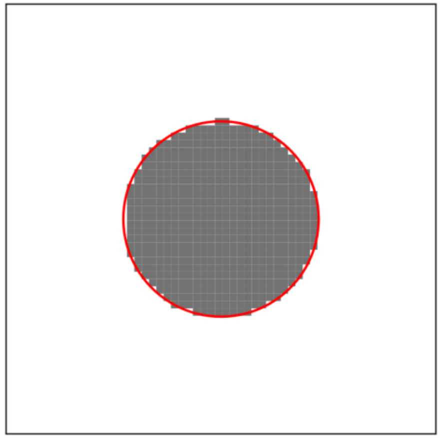
M. Ghommem et al. (2012). DOI: 10.4203/ccp.100.133

discretize with LBM

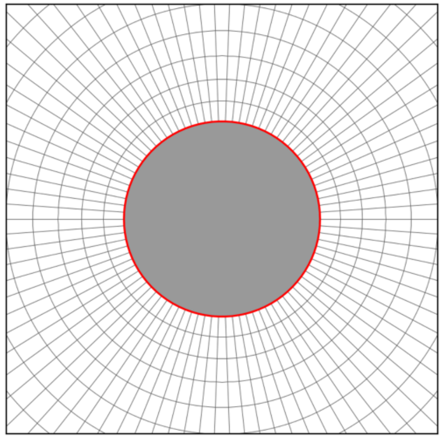


Cartesian LBM — staircased

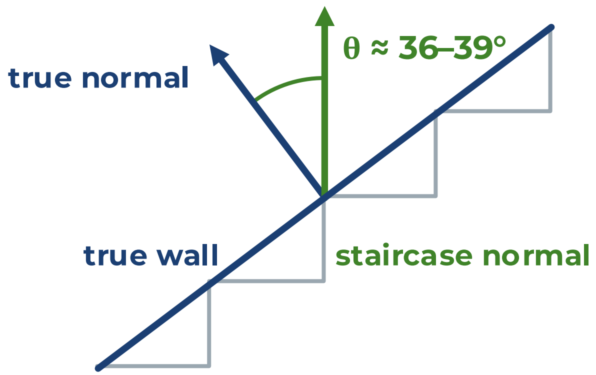
The problem: Cartesian lattice grid



Staircased wall

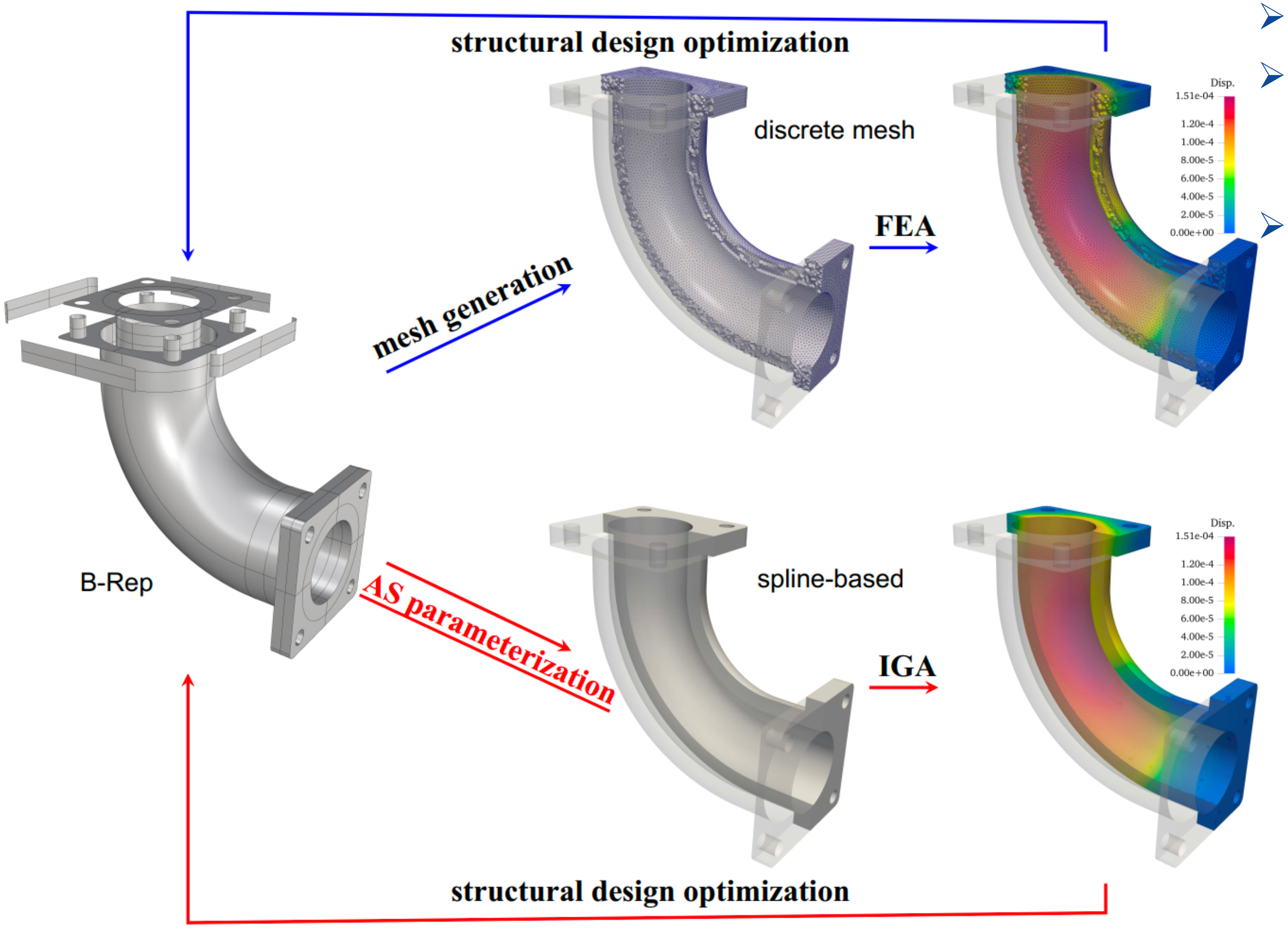


Body-fitted grid



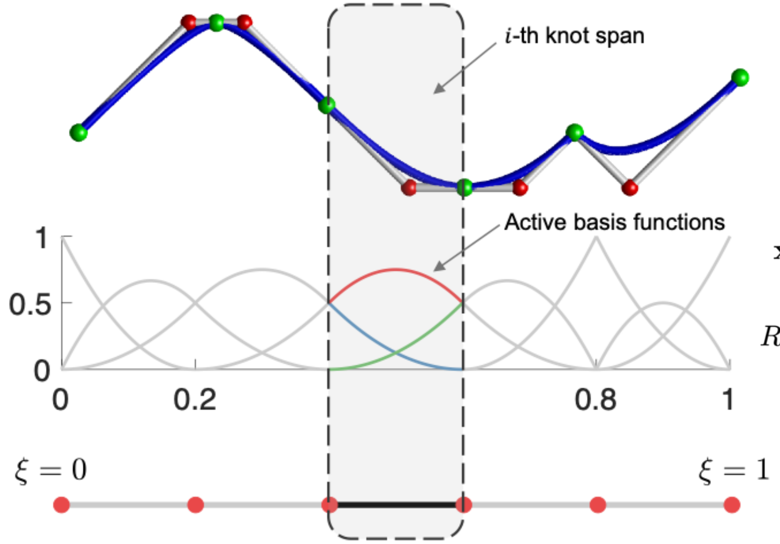
Steps shrink with h; facet normals stay axis-aligned

- **Exact streaming ⇒ uniform Cartesian lattice**
  - a curved wall reduces to a staircase
- **Position error O(h) — converges under refinement**
- **Normal error 36°–39° — does not converge**
  - facets remain axis-aligned at every resolution
  - spurious wall-normal velocity ≈ 0.6 U — no-penetration violated by ≈60%
- **Consequence: inaccurate wall** shear, pressure, drag/lift and heat transfer; spurious mass flux through the wall.



Design-analysis-optimization product development workflow

- **Proposed by Hughes et al., 2005.**
- **From CAD to analysis**
  - Conventional: CAD → mesh generation → FEA
  - IGA: CAD → analysis; geometry stays exact
- **Why it helps**
  - no data conversion, exact geometry;
  - design and analysis on one model;
  - uniform h-, p-, k-refinement;
  - high-order continuity;
  - higher accuracy per degree of freedom;



$$\mathbf{x}(\xi) = \sum_{i=1}^{N_{cp}} \mathbf{P}_i R_i(\xi)$$

$$R_i(\xi) = \frac{w_i N_{i,p}(\xi)}{\sum_j w_j N_{j,p}(\xi)}$$

NURBS curve, control net, and basis over ξ

Why put IGA and LBM together?

TU Delft Delft University of Technology

Navier–Stokes + FVM | FEA | IGA

\+
- **Exact geometry representation**
- High order / continuity approximations
- Higher accuracy per degree of freedom
- **CAD integration → design optimisation**

–
- Complex algorithm / implementation
- High(er) assembly and solver costs
- Difficult to achieve parallel scalability

Boltzmann + LBM

\+
- **Extremely simple algorithm**
- **Excellent parallel scalability**
- Easy handling of complex geometries
- Easy extension to multiphysics problems

–
- Cartesian grids → staircase approximation
- Poor QoI accuracy at curved boundaries
- Stability limits at high Re
- Weak compressibility assumption

GOAL: Keep LBM's fast, explicit solver, and replace its staircased boundaries with IGA's exact geometry.

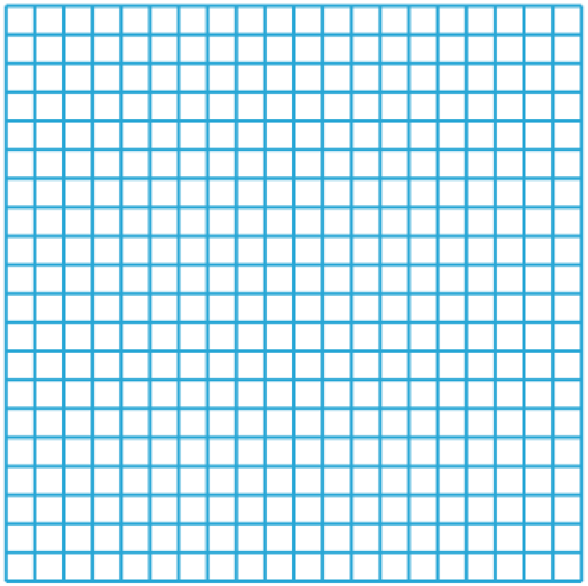
→ IGA-LBM: the isogeometric lattice Boltzmann method.

WCCM MUNICH 2026 ECCOMAS

Outline

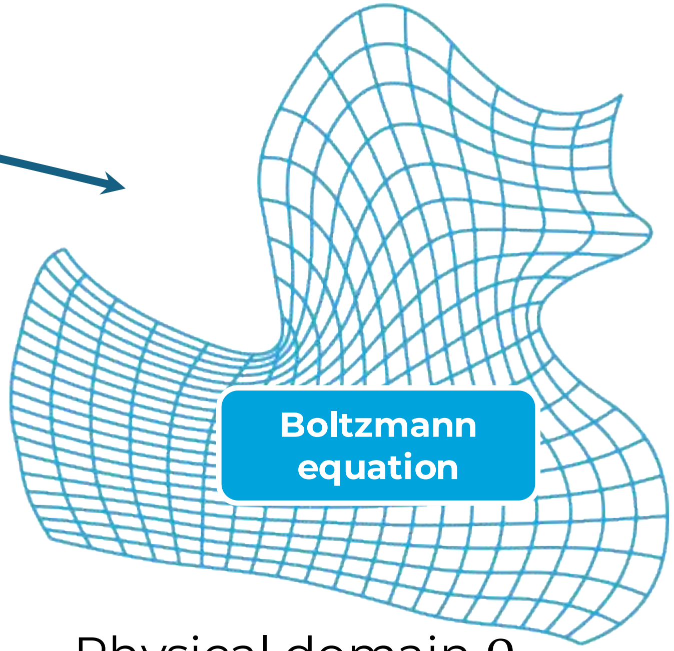
First step: geometry mapping

- The physical domain Ω is parameterized by a NURBS representation $\boldsymbol{x}$;
- The lattice BGK equation is transformed into the parameter domain.



Parameter domain $\hat{\Omega}$

$$\boldsymbol{x}(\xi,\eta) = \sum_i N_i(\xi,\eta)\boldsymbol{P}_i = \boldsymbol{\Phi}\mathbf{P}$$



Physical domain Ω

The relationship between the physical and computational domains satisfies

$$\begin{bmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{bmatrix} = \frac{1}{|J|}\begin{bmatrix} y_\eta & -x_\eta \\ -y_\xi & x_\xi \end{bmatrix}$$ with J being the Jacobian matrix

- Using the chain rule, the **convection term** in the **Boltzmann equation** can be rewritten as

$$\boldsymbol{e}_\alpha \cdot \nabla f_\alpha = e_{\alpha x}\frac{\partial f_\alpha}{\partial x} + e_{\alpha y}\frac{\partial f_\alpha}{\partial y}$$

$$= e_{\alpha x}\left(\frac{\partial f_\alpha}{\partial \xi}\xi_x + \frac{\partial f_\alpha}{\partial \eta}\eta_x\right) + e_{\alpha y}\left(\frac{\partial f_\alpha}{\partial \xi}\xi_y + \frac{\partial f_\alpha}{\partial \eta}\eta_y\right)$$

can be precomputed

$$= \left(e_{\alpha x}\xi_x + e_{\alpha y}\xi_y\right)\frac{\partial f_\alpha}{\partial \xi} + \left(e_{\alpha x}\eta_x + e_{\alpha y}\eta_y\right)\frac{\partial f_\alpha}{\partial \eta} = \tilde{e}_{\alpha\xi}\frac{\partial f_\alpha}{\partial \xi} + \tilde{e}_{\alpha\eta}\frac{\partial f_\alpha}{\partial \eta}$$

- Represent the **particle distribution function** using NURBS representation

$$f_\alpha(\xi,\eta) = \sum_i N_i(\xi,\eta) f_{i,\alpha} = \mathbf{\Phi}\mathbf{F}_\alpha \quad \Rightarrow \quad \frac{\partial f_\alpha}{\partial \xi} = \mathbf{\Phi}_\xi \mathbf{F}_\alpha, \quad \frac{\partial f_\alpha}{\partial \eta} = \mathbf{\Phi}_\eta \mathbf{F}_\alpha$$

- Unlike IGA, LBM requires **pointwise updates** at discrete velocities. Collocation!

Evaluation of differential operators

- Evaluate differential operators at the **Greville abscissae** (→ Schonberg-Whiteney condition)

$$\frac{\partial f_\alpha}{\partial \xi}(\hat{\xi}, \hat{\eta}) = \hat{\boldsymbol{\Phi}}_\xi \hat{\boldsymbol{\Phi}}^{-1}\hat{\mathbf{F}}_\alpha, \qquad \frac{\partial f_\alpha}{\partial \eta}(\hat{\xi}, \hat{\eta}) = \hat{\boldsymbol{\Phi}}_\eta \hat{\boldsymbol{\Phi}}^{-1}\hat{\mathbf{F}}_\alpha$$

  where $\hat{\boldsymbol{\Phi}}$, $\hat{\boldsymbol{\Phi}}_\xi$, and $\hat{\boldsymbol{\Phi}}_\eta$ are the collocation matrices and $\hat{\mathbf{F}}_\alpha$ are the collection of the particle

- ⚠ Naively computing $\hat{\boldsymbol{\Phi}}_* \hat{\boldsymbol{\Phi}}^{-1}$ at each time step is expensive. Precompute!

- Define $\boldsymbol{X}_* \hat{\boldsymbol{\Phi}} = \hat{\boldsymbol{\Phi}}_*$, store $\boldsymbol{X}_*$, and reuse in each time step

$$\hat{\boldsymbol{\Phi}}^{\mathrm{T}} \boldsymbol{X}_*^{\mathrm{T}} = \hat{\boldsymbol{\Phi}}_*^{\mathrm{T}} \quad \Rightarrow \quad \boldsymbol{X}_*^{\mathrm{T}} = \left(\hat{\boldsymbol{\Phi}}^{\mathrm{T}}\right)^{-1} \hat{\boldsymbol{\Phi}}_*^{\mathrm{T}} \quad \Rightarrow \quad \boldsymbol{X}_* = \hat{\boldsymbol{\Phi}}_* \hat{\boldsymbol{\Phi}}^{-1}$$

- ⚠ $\boldsymbol{X}_*$ does not rely on the parameterization, only on the basis functions and collocation points

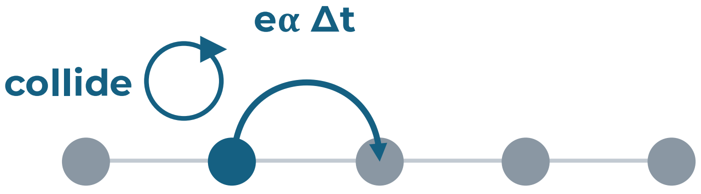
Semi-discrete discrete-velocity Boltzmann equation (DVBE)

$$\frac{\partial f_\alpha}{\partial t} = S_\alpha(f) + C_\alpha(f), \qquad S_\alpha(f) = -\left(\tilde{e}_{\alpha\xi} \boldsymbol{X}_\xi \hat{\mathbf{F}}_\alpha + \tilde{e}_{\alpha\eta} \boldsymbol{X}_\eta \hat{\mathbf{F}}_\alpha\right), \qquad C_\alpha(f) = -\frac{1}{\tau}\left(f_\alpha - f_\alpha^{eq}\right)$$

WCCM MUNICH 2026 ECCOMAS

What we change: the object being discretised

Classical LBM: stream–collide



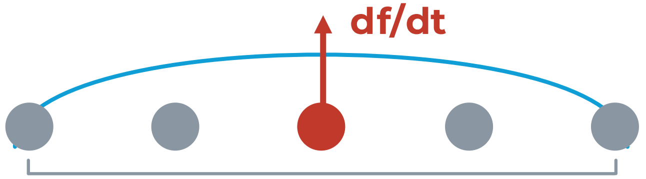
collide (relax), then stream

$$f_\alpha(\boldsymbol{x} + \boldsymbol{e}_\alpha \Delta t,\, t + \Delta t) = f_\alpha(\boldsymbol{x}, t) - \frac{\Delta t}{\tau}(f_\alpha - f_\alpha^{\mathrm{eq}})$$

fused collide + stream in one lattice step

- packets hop along fixed links
- space–time locked ⇒ uniform lattice only
- $\nu = c_s^2(\tau - \tfrac{1}{2}\Delta t)$ — time-step artefact
- curved walls ⇒ staircased bounce-back

This work: collocation method of lines



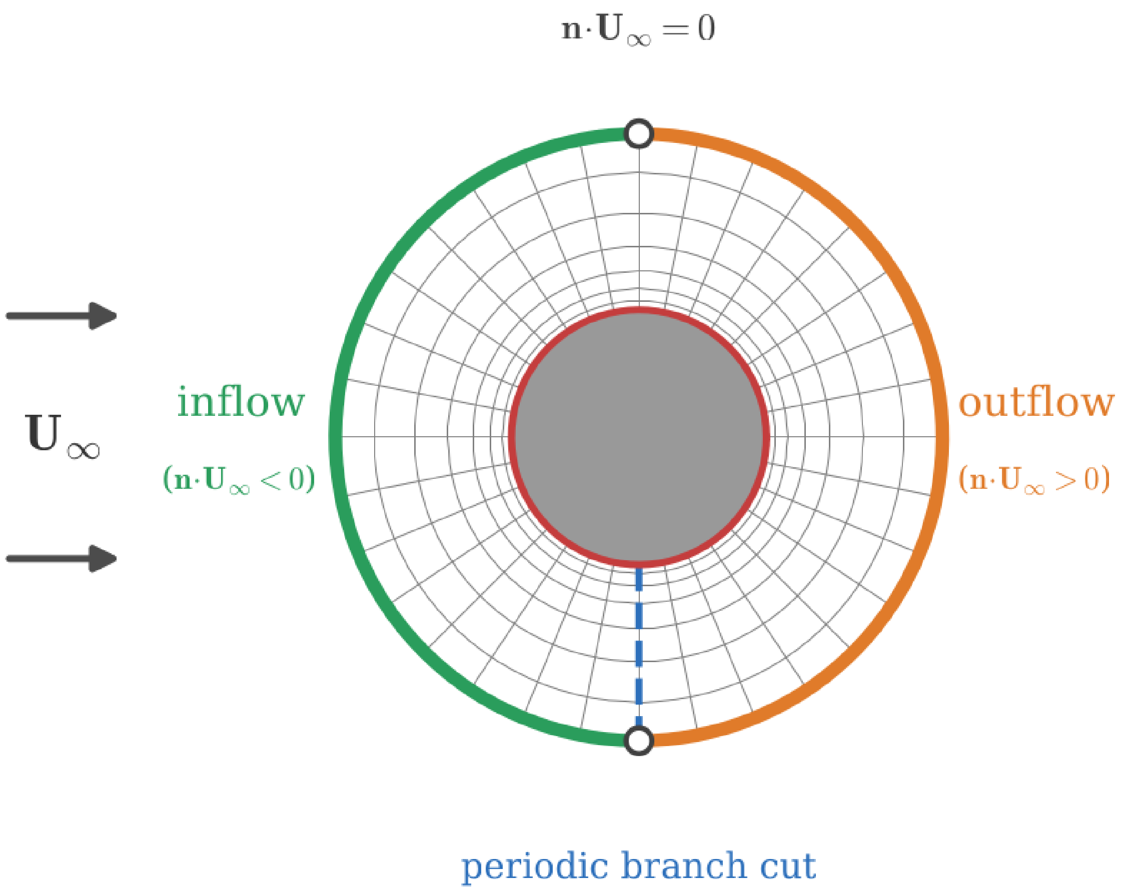
collocation derivative Dξ (banded 5-point)

$$\frac{\partial f_\alpha}{\partial t} + \underbrace{\tilde{e}_{\alpha\xi}\frac{\partial f_\alpha}{\partial \xi} + \tilde{e}_{\alpha\eta}\frac{\partial f_\alpha}{\partial \eta}}_{\text{transformed advection}} = -\frac{1}{\tau}(f_\alpha - f_\alpha^{\mathrm{eq}})$$

$\tilde{e}_\alpha = J^{-1} e_\alpha$ (contravariant, precomputed once)

- nodal spline values evolve in place
- no Lagrangian streaming
- continuous-time BGK → RK4 + low-pass filter
- **$\nu = c_s^2\tau$ — no time-step artefact**

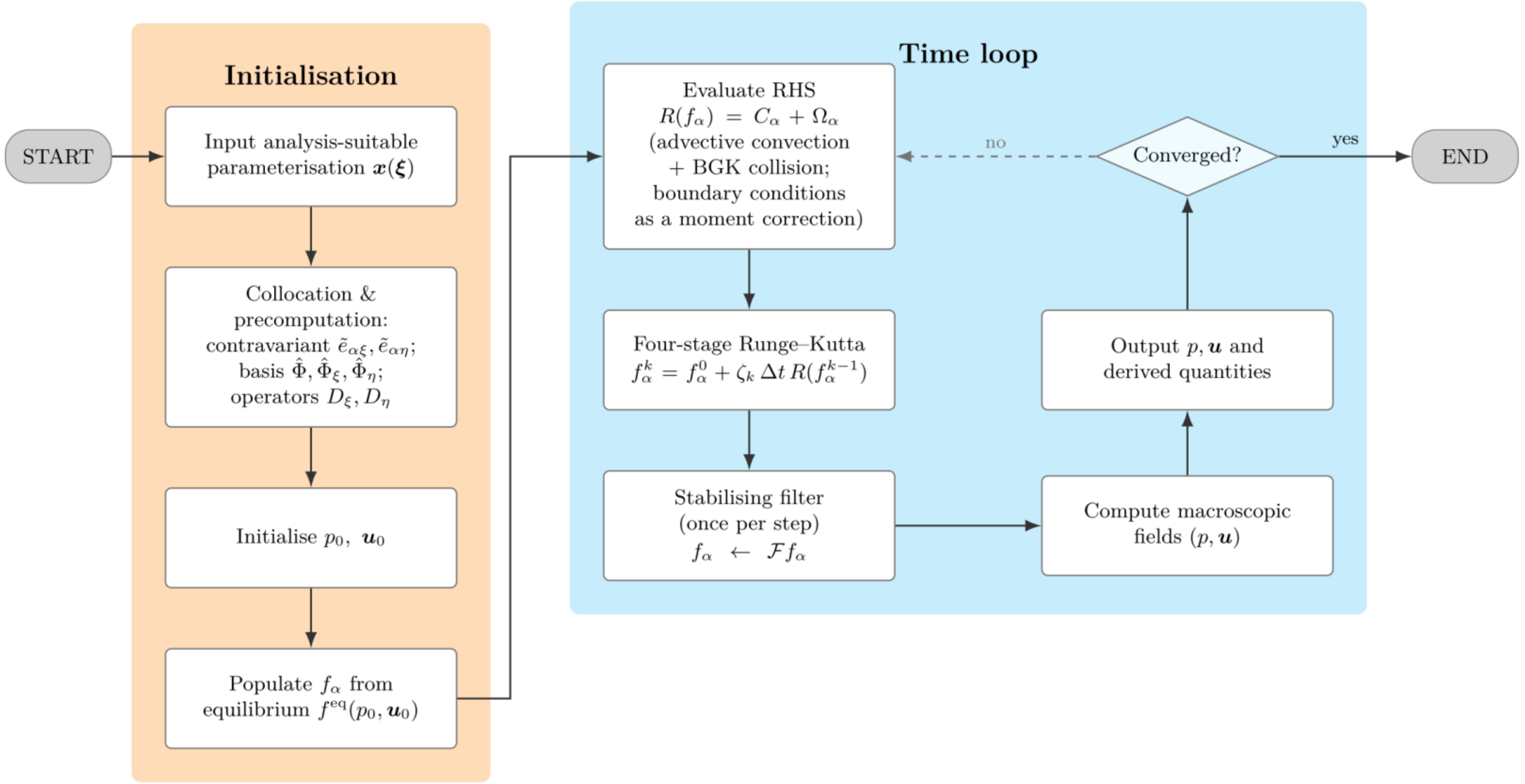
Boundary conditions without bounce-back



$$\boldsymbol{u}(\boldsymbol{x}_b) = \boldsymbol{u}_{\text{wall}}, \quad \left.\frac{\partial p}{\partial n}\right|_{\boldsymbol{x}_b} = 0, \quad f_\alpha^{\text{eq}}(\boldsymbol{x}_b) = f_\alpha^{\text{eq}}\big(p_b, \boldsymbol{u}_{\text{wall}}\big)$$

- Impose at the moment level: prescribe u, p → equilibrium target (f_α never overwritten).
  - **Wall (no-slip):** $u = u_{\text{wall}}$, $\partial p/\partial n = 0$
  - **Inflow (Dirichlet):** $u = U_\infty$, $p = p_\infty$
  - **Outflow (Neumann):** $\partial u/\partial n = 0$
  - **Branch cut (periodic):** $f(\theta = 0) = f(\theta = 2\pi)$

The algorithm overview



$$\frac{\mathrm{d}f_\alpha}{\mathrm{d}t} = R_\alpha(f) := \underbrace{C_\alpha}_{\text{advection}} + \underbrace{\Omega_\alpha}_{\text{BGK}}$$

Semi-discrete transport

$$f_\alpha^{(s)} = f_\alpha^{n} + \zeta_s \,\Delta t\, R_\alpha\big(f^{(s-1)}\big), \qquad \zeta_s = \left(\tfrac{1}{4}, \tfrac{1}{3}, \tfrac{1}{2}, 1\right)$$

Low-storage RK4 · 4 stages

$$\Delta t \le \min\Big\{ 2.785\,\tau, \quad \mathrm{CFL}/s_{\max} \Big\}, \qquad \tau = \mathcal{O}(\mathrm{Re}^{-1})$$

Explicit stability bound

Per step

- 4 RK stages + 1 filter
- BCs enter the RHS at every stage

Cost

- local B-spline support ⇒ banded operators ⇒ O(N) per step
- **0.36–0.38 μs / DOF / step**
- ≈ a few classic LBM updates

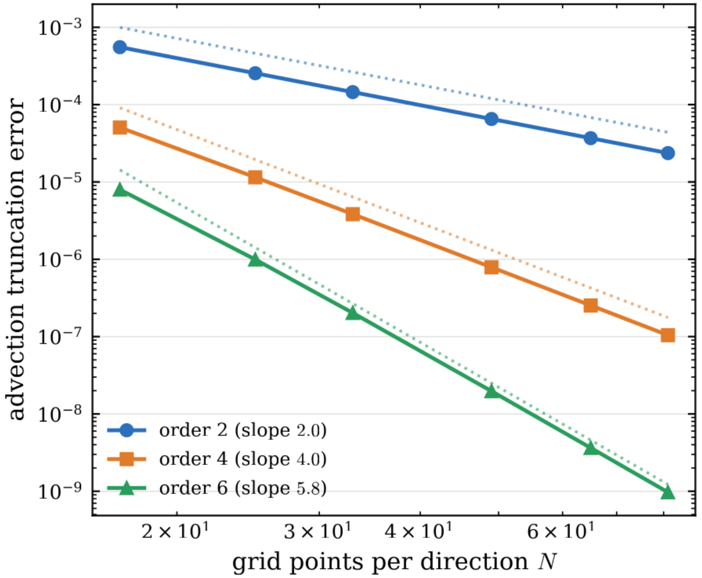
Time step

- **Δt free, unlike classic LBM**
- collision sets Δt, not flow speed
- use Δt = min{2τ, CFL/s_max}, CFL ≈ 0.1–0.3

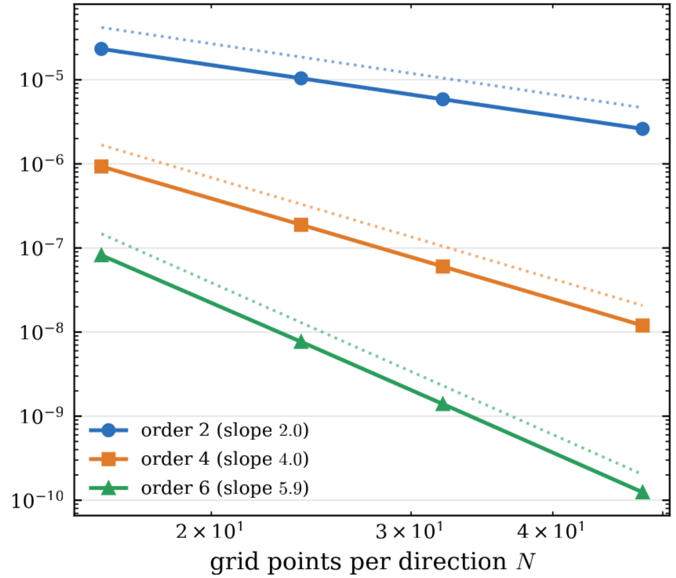
Outline

1. Background and motivation
2. Isogeometric lattice Boltzmann method
3. **Numerical examples and applications to FSI**
4. Conclusions and outlook

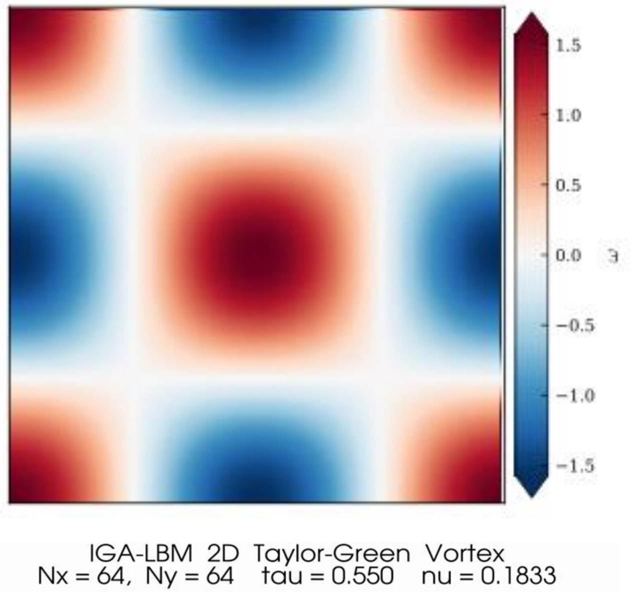
Spatial order of accuracy: 2 / 4 / 6

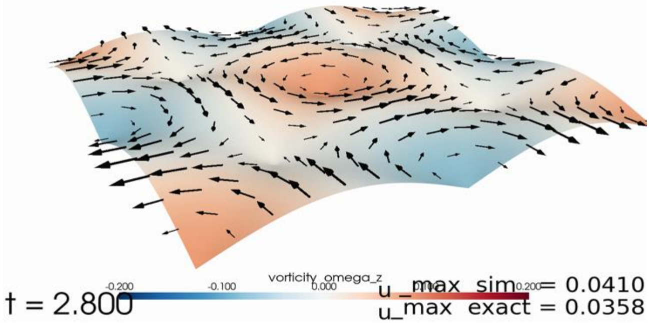


2D Taylor–Green vortex



3D ABC / Beltrami flow (D3Q19)

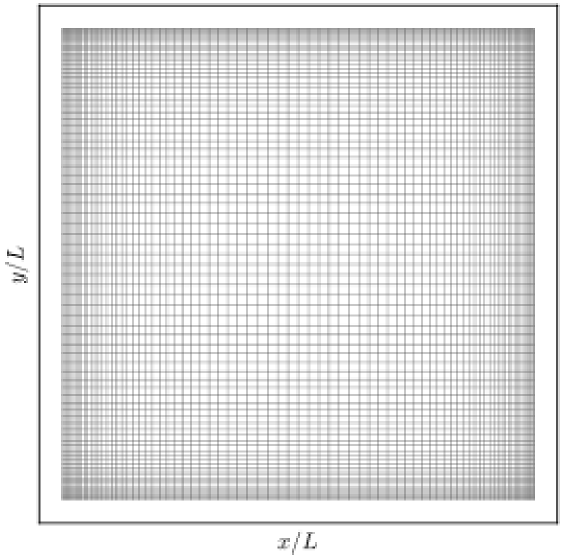




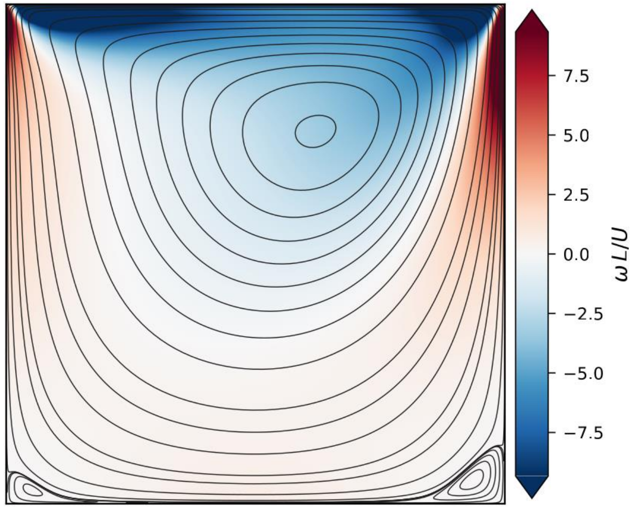
2D Taylor–Green vortex

- **Exact-solution benchmarks:** Taylor–Green vortex and 3D Arnold–Beltrami–Childress flow flow: error measured against truth.
- **Design orders recovered:** measured slopes hit 2 / 4 / 6 in every case.
- **No loss at the wall**.

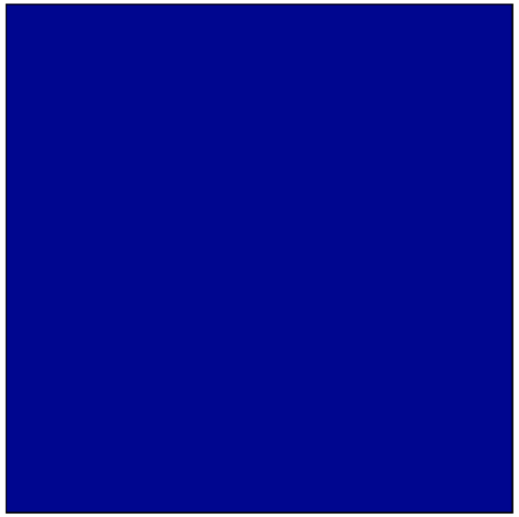
Lid-driven cavity, Re = 100

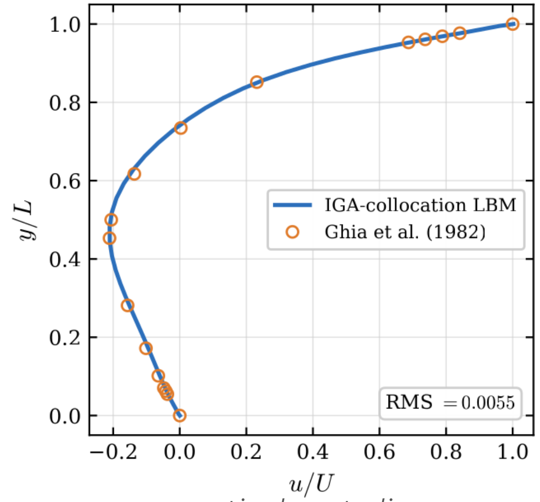


Clustered grid



vorticity + streamlines





u on vertical centreline

Setup: 48×48 stretched grid · Re = 100

- Square domain

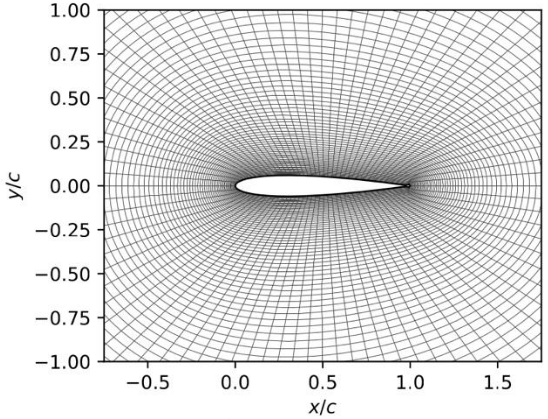
Benchmark: Ghia et al. (1982)

- Centreline velocity matches: RMS 5.5×10^{-3} (u), 4.3×10^{-3} (v)

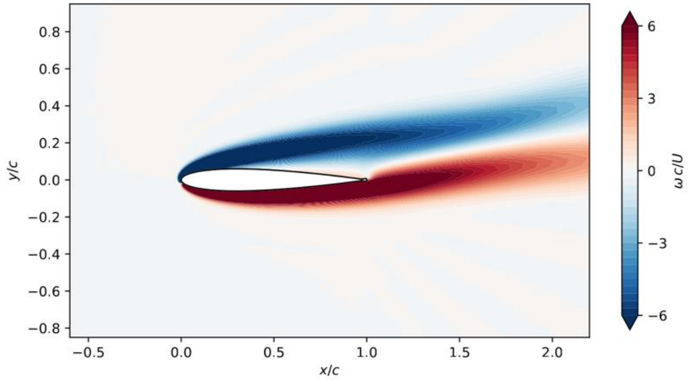
Grid K×K	24	32	48	64
RMS vs Ghia	1.92×10^{-2}	1.11×10^{-2}	5.50×10^{-3}	5.50×10^{-3}

- **Refine the mesh → error falls, then flattens by 48^2**
- That plateau is the $O(Ma^2)$ compressibility floor: the athermal-LBM model limit, not the mesh
- **So high order reaches the floor on a very coarse grid**

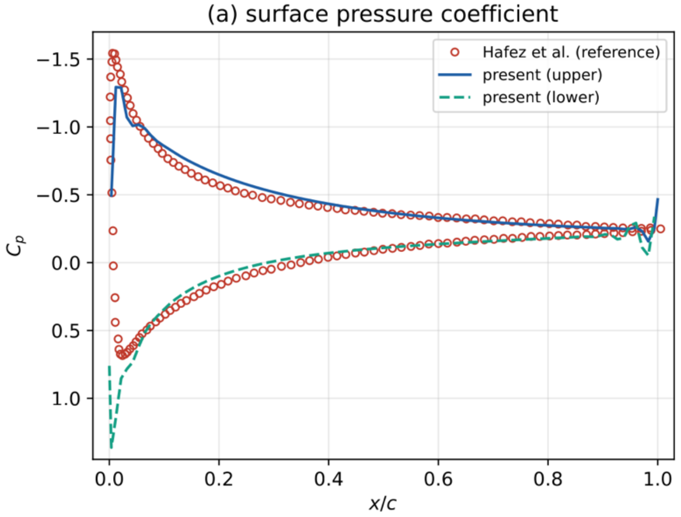
NACA0012, Re = 500, α = 10°

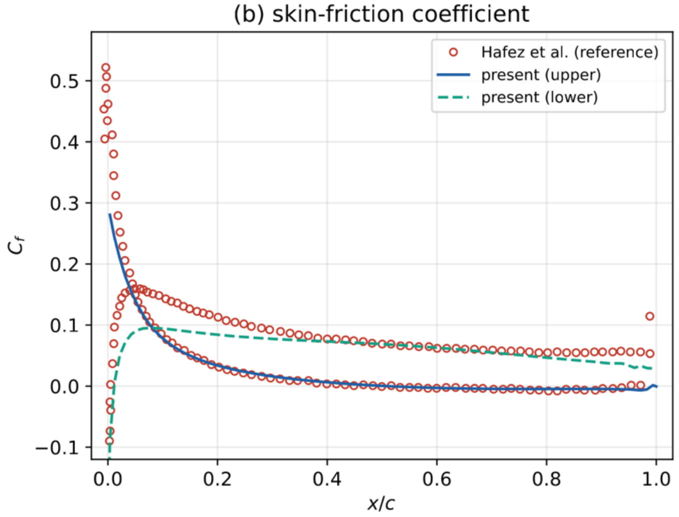


elliptic body-fitted O-grid (180×110)



vorticity: LE suction, shear layers, thin wake



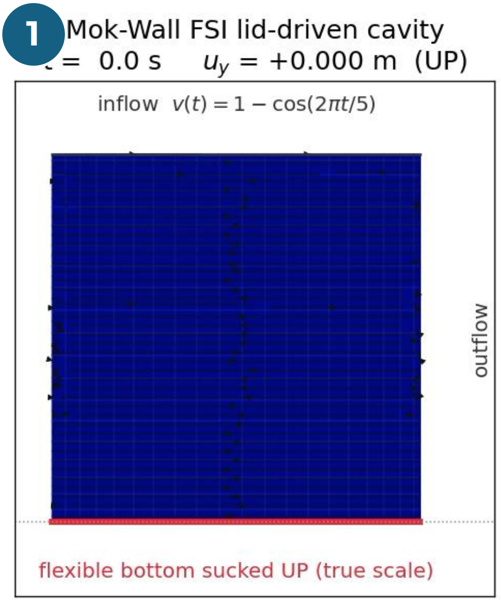


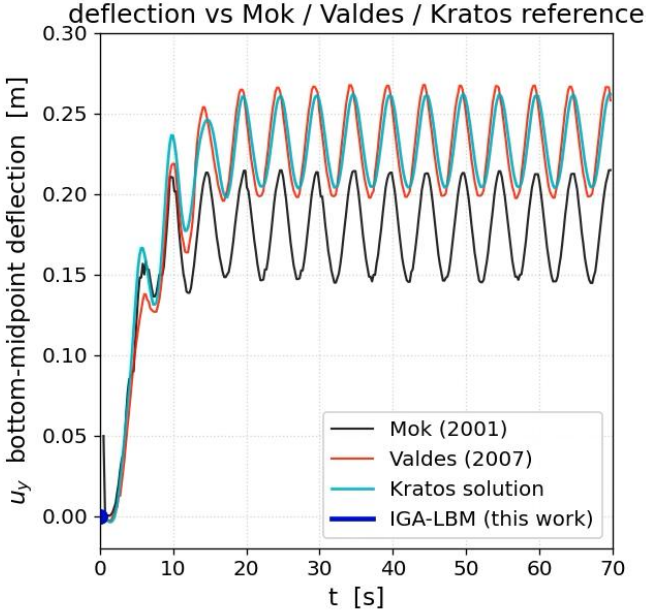
surface pressure and skin friction vs Hafez et al. (2006)

- **Setup: NACA0012 · Re = 500 · α = 10°**
  - Body-fitted O-grid 180×110
- **Real CAD grid**
  - Grid generated by an external mesher, only sampled coordinates (no analytic map)
- **It works**
  - **Uniform flow exact to $7×10^{-15}$**
  - Forces match references: C_l = 0.295, C_d = 0.134
  - Ref. Hafez et al. (2006)

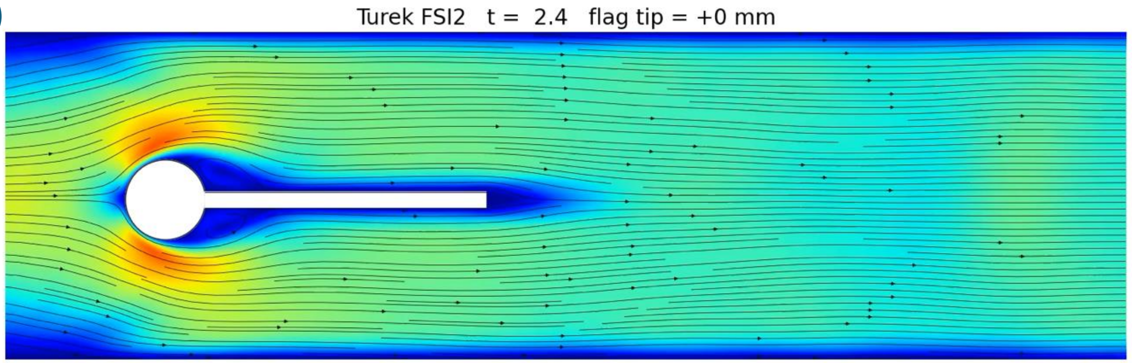
Application to Fluid-Structure Interaction

1

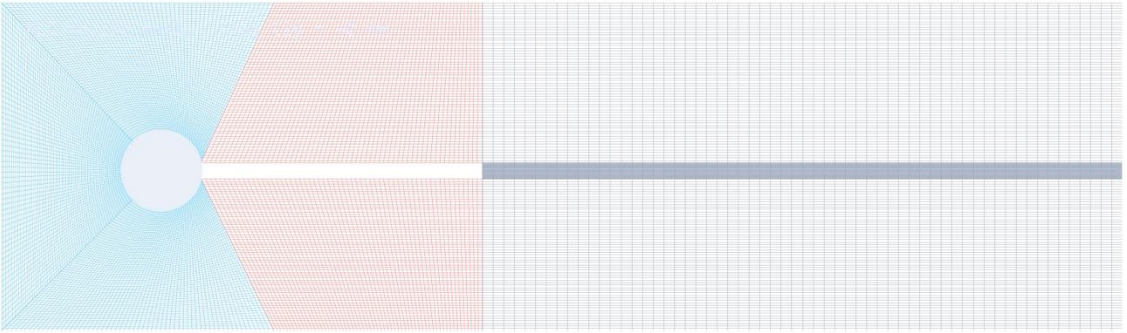


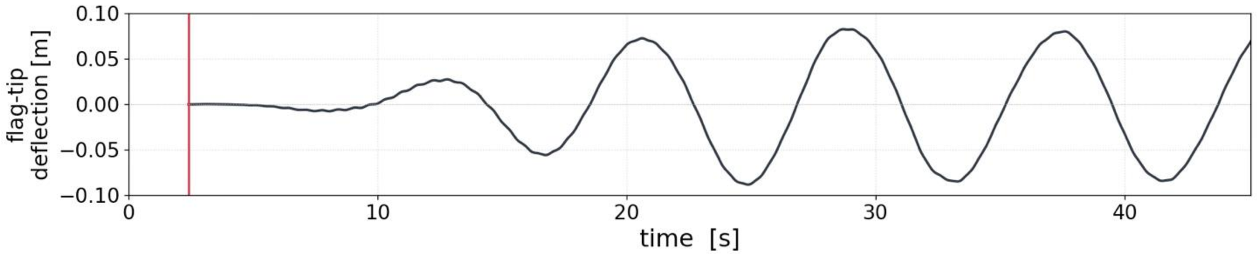


2



3





Moving boundaries on a body-fitted grid — same solver.

1 **Mok flexible wall** — panel deflecting in channel flow; FSI verification case.

2 **Turek–Hron FSI2** — elastic flag oscillating behind a fixed cylinder.

3 **Body-fitted mesh motion** — grid deforms with the structure; no remeshing.

Outline

1. Background and motivation
2. Isogeometric lattice Boltzmann method
3. Numerical examples and applications to FSI
4. **Conclusions and outlook**

Conclusions & Outlook

Conclusions

- **Body-fitted isogeometric LBM:** strong-form BGK on NURBS, RK4, high-order filter.
- **Geometry exactness:** high order accuracy holds right up to the wall.
- **Validated & general:** TGV, lid-driven cavity, NACA0012; also 3D and FSI.

Outlook

- **Complex multipatch CAD:** extend and apply to trimmed, multipatch geometries.
- **Stiff-accurate IMEX:** lift the explicit $\Delta t = O(Re^{-1})$ limit.
- **GPU / matrix-free:** body-fitted 3D and moving-boundary FSI.

Preprint: https://arxiv.org/pdf/2509.11427v2

SCAN ME

Work with us at TU Delft

We are hiring!

PhD POSITION · 4 YEARS · FULLY FUNDED

Quantum Algorithms for Numerical Simulation in Aerospace

- **Quantum-CFD Lab** · quantum lattice Boltzmann for CFD
- Based at **TU Delft**, the Netherlands
- MSc in **Maths, CS, Aerospace or Physics**
- Strong **Python / C++** skills are preferred

Scan to apply

Apply by **31 Aug 2026** · Contact: **Matthias Möller <M.Moller@tudelft.nl>**

Many thanks for your kind attention!

Q&A

If you are interested in my research, please feel free to contact me! ;-)

- **Email:** y.ji-1@tudelft.nl
- **GitHub:** jiyess
- **Homepage:** https://jiyess.github.io/