# Cut Galerkin difference methods and more

## 11th deal.II Users and Developers Workshop, Fort Collins, Colorado, USA

Peter Munch†

*in collaboration with Gunilla Kreiss, Simon Sticko, Ivy Weber*

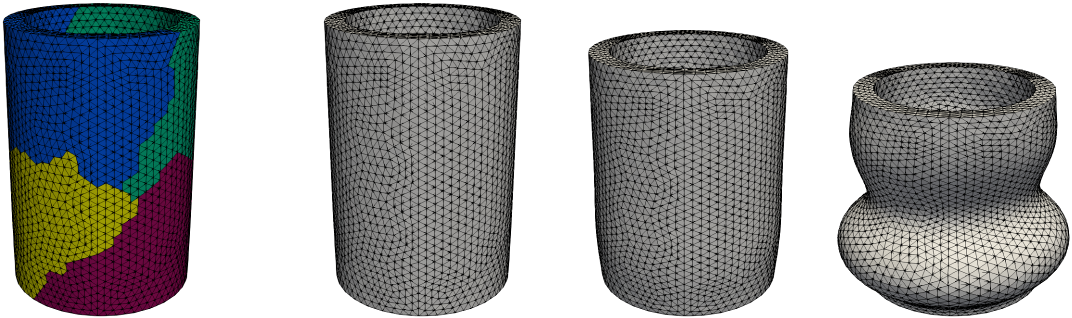
†Department of Information Technology, Uppsala University, Sweden

August 15, 2024

# About myself

- PhD at University of Augsburg, Germany

  "Matrix-free finite-element computations at extreme scale and for challenging applications"
- most significant contribution to deal.II: simplex- and mixed-mesh support



- postdoc at Department of Information Technology at Uppsala University, Sweden
  - Division of Scientific Computing: Gunilla Kreiss, Murtazo Nazarov
  - Uppsala Architecture Research Team: Stefanos Kaxiras

# About myself (cont.)

my research interests:

## high-performance computing

- matrix-free operator evaluation
- simulations up to 300k procs/5TDoFs
- consensus-based algorithms

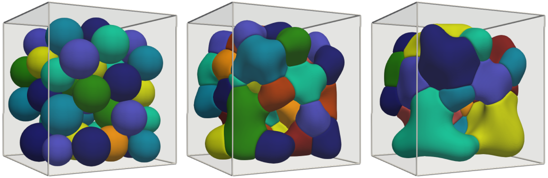
## preconditioning

- multigrid for locally refined meshes + hp
- smoother development (e.g., ASM)
- stage-parallel IRK
- monolithic GMG for stabilized NS
- non-nested multigrid
- multigrid for space-time FEM

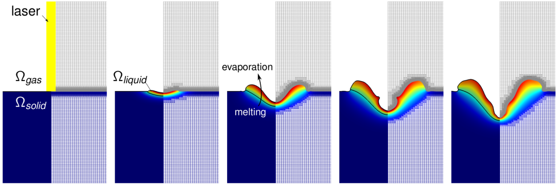
## non-matching grids $\rightarrow$ step-87 & step-89 (new)

## applications

- computational fluid mechanics
- computational plasma physics (6D)
- solid-state sintering



- additive manufacturing



# deal.II at Uppsala University

contributions to deal.II:

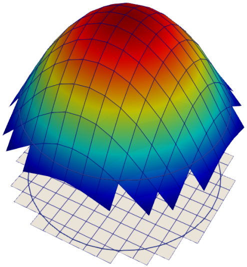
- matrix-free infrastructure $\rightarrow$ Martin Kronbichler, Katharina Kormann (2012)
- matrix-free GPU support $\rightarrow$ Karl Ljungkvist
- CutFEM $\rightarrow$ Simon Sticko (2022)
- Hermite elements $\rightarrow$ Ivy Weber (2023)

applications:

- two-phase flow (diffuse/sharp interface methods) $\rightarrow$ Gunilla Kreiss et al.
- stage-parallel Runge–Kutta methods $\rightarrow$ Ivo Dravins, Maya Neytcheva
- Galerkin difference methods (GDM) $\rightarrow$ Gunilla Kreiss, PM

# deal.II at Uppsala University (cont.)

1 CutFEM/DG $\rightarrow$ immersed domains



Challenge: small cuts $\rightarrow$ stabilization

2 Time stepping

Time-step restriction for Lagrange elements:

$$\Delta t = \mathcal{O}\left(\frac{1}{p^{\alpha}}\right) \quad \text{with } \alpha > 1.$$

No time-step restriction for:

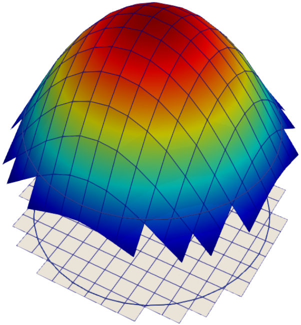
- Hermite elements
- Galerkin difference methods

1+2 CutFEM + Hermite elements/Galerkin difference methods?

Part 1:

# CutFEM

# Problem statement



Example: solve Poisson problem on $\Omega$, using a background mesh

$$a_h(u_h, v_h) = L_h(v_h), \quad \forall v_h \in V_\Omega^h,$$

where

$$a_h(u_h, v_h) = (\nabla u_h, \nabla v_h)_\Omega - (\partial_n u_h, v_h)_\Gamma - (u_h, \partial_n v_h)_\Gamma + \left(\frac{\gamma_D}{h} u_h, v_h\right)_\Gamma,$$

$$L_h(v_h) = (f, v)_\Omega + \left(u_D, \frac{\gamma_D}{h} v_h - \partial_n v_h\right)_\Gamma.$$

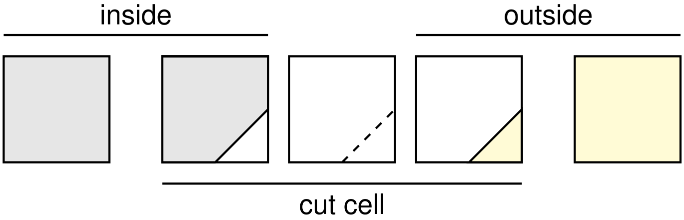
- stabilization for small cut cells: $A_h(u_h, v_h) := A_h(u_h, v_h) + \gamma_A h^{-2} j(v, u_h)$ with, e.g., :

$$j(v, u_h) = \sum_{F \in \mathcal{F}_\Gamma} \sum_{k=1}^{p} h^{2k+1} \left\langle \left[\partial_n^k v\right], \left[\partial_n^k u_h\right] \right\rangle \qquad \rightarrow \text{ ghost penalty}$$

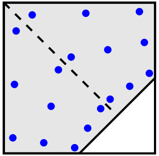
- extension: two domains, two-phase flow $\rightarrow$ moving interface

# Types of integration

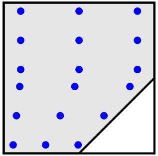
- cell can be inside/cut/outside




- similar for faces
- options for definition of quadrature rules:



Carraro & Wetterauer, 2015



Saye, 2015

# Workflow in deal.II

full example: step-85

- define level-set field

```
Functions::SignedDistance::Sphere<dim> signed_distance_sphere;
```

- categorize cells

```
NonMatching::MeshClassifier<dim> mesh_classifier(/*...*/);
mesh_classifier.reclassify();
```

- perform different integrals depending on the category and position of cut

```
NonMatching::FEValues<dim> nm_fe_values(/*...*/, mesh_classifier,/*...*/ , ls);
for (const auto &cell : dof_handler.active_cell_iterators())
  {
    non_matching_fe_values.reinit(cell);
    if (const auto fe_values = non_matching_fe_values.get_inside_fe_values())
      {
        // continue as normal
      }
  }
```

- similar on surface (NM::FEImmersedSurfaceValues) and faces (NM::FEInterfaceValues).

# Outlook

Alternatively, integration on a set of unstructured quadrature rules can be performed in a matrix-free way, using FEPointEvaluation[1]

```
FEPointEvaluation<dim> phi(/*...*/);

phi.reinit(/*...*/);

phi.evaluate(buffer, EvaluationFlags::value);

for(const auto q : phi.quadrature_point_indices ())
  phi.submit_value(phi.get_value(q), q);

phi.integrate(buffer, EvaluationFlags::value);
```

Notes:

- can exploit tensor-product structure of shape functions to reach high performance
- to be used together with DoFCellAccessor or FEEvaluation
- example: step-87 (sharp interface method)
- major challenge: preconditioning

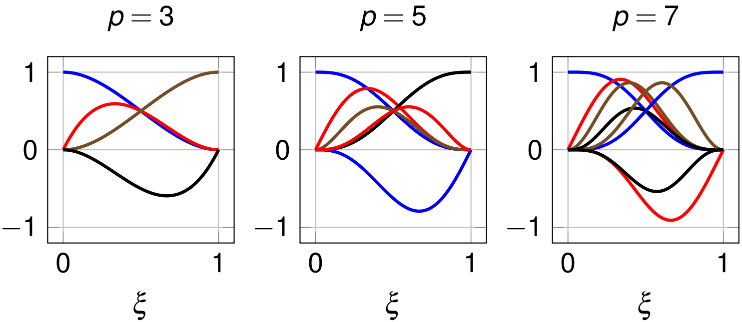
[1] Bergbauer, Munch, Wall, Kronbichler, 2024, High-performance matrix-free unfitted finite element operator evaluation, arxiv

Part 2:

# Hermite elements

# Hermite elements in a nutshell

- $p$-degree Hermite elements have $\frac{p-1}{2}$ regularity
- shape functions: ▷ *all* $(r+1)^d$ *DoFs assigned to node*




- Weber et al. (2022) showed that time-step restriction for the wave equation is independent of $p$

in deal.II available as:

```
FE_Hermite<dim> fe(fe_degree);
```

# Challenges & outlook

<u>Challenge:</u> linear solver

- iterative solver $\rightarrow$ additive Schwarz method (ASM)
- direct solver $\rightarrow$ Adrianna Gillman (CU Boulder)

<u>Limitations & outlook:</u>

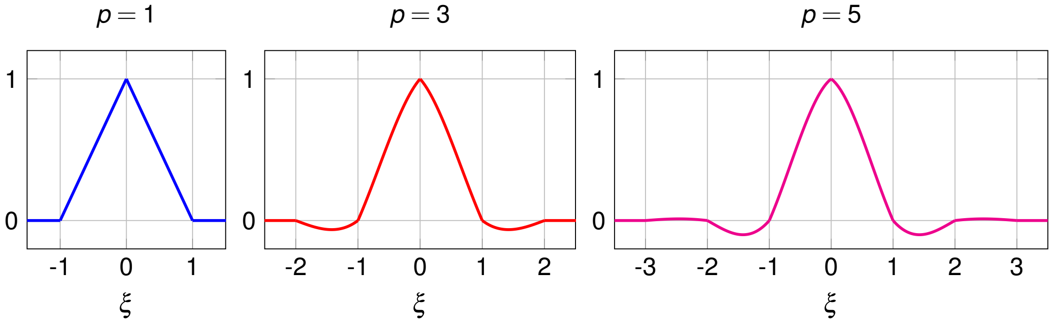
- only Cartesian meshes $\rightarrow$ different meshes/mappings (unstructured, codim)
- application: beams in solid mechanics (Euler–Bernoulli, Kirchhoff, Timoschenko)

Part 3:

# Galerkin difference methods (GDM)

# Galerkin difference methods in a nutshell

Galerkin difference methods: a type of FEM with shape functions spanning over more cells:



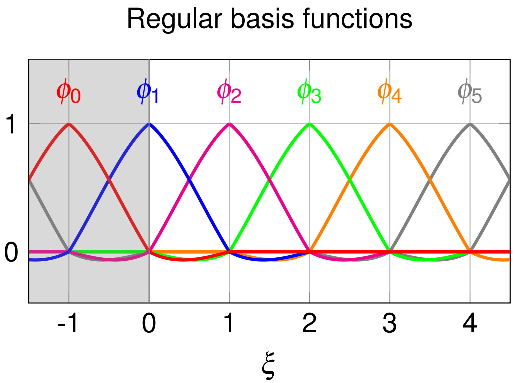

▷ *Lagrange functions associated with continuous piecewise polynomials*

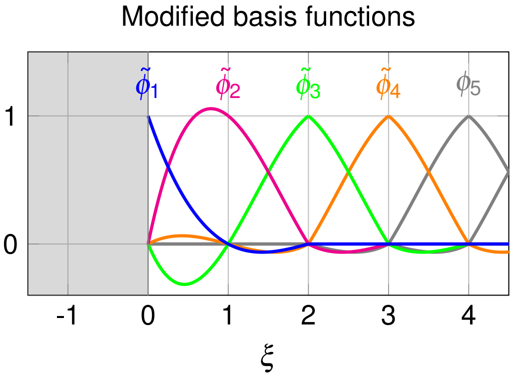
Banks and Hagstrom (2016) showed "for first-order systems no significant CFL penalty".

<u>Advantages:</u> no additional DoFs, same stencil for each internal point (similar to FDM)

## Galerkin difference methods in a nutshell (cont.)

- at boundary: ① express basis functions that are outside of the computational domain as linear combinations of internal basis functions, ② eliminate from system.

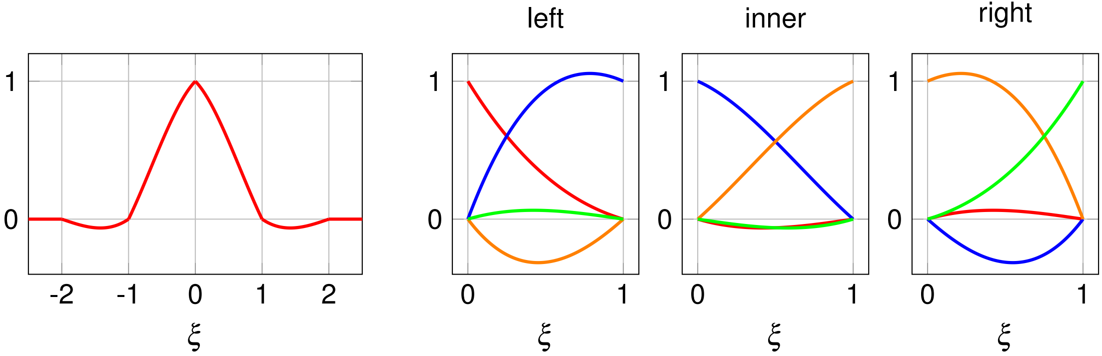



- extension to higher dimensions: tensor product

# Implementation details in deal.II

① consider all non-zero basis functions in a cell $\rightarrow$ “elements”




basis-function-centric $\longleftrightarrow$ cell-centric view ($p$ unique “elements” in 1D)

② custom element:
hp::FECollection with $p^d$ entries $\rightarrow$ FE_Q_Base $\rightarrow$ AnisotropicPolynomials $\rightarrow$ Polynomial

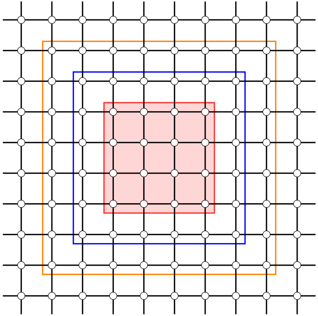
# Implementation details in deal.II (cont.)

```
template <int dim>
class FE_GDM : public FE_Q_Base<dim>
{
public:
  FE_GDM(const ScalarPolynomialsBase<dim> &poly)
    : FE_Q_Base<dim>(poly, create_data(poly.n()), std::vector<bool>(1, false))
  {}

  std::string get_name() const override {/*...*/}

  std::unique_ptr<FiniteElement<dim>> clone() const override {/*...*/}

private:
  static FiniteElementData<dim>
  create_data(const unsigned int n)
  {
    std::vector<unsigned int> dofs_per_object(dim + 1);
    dofs_per_object[dim] = n;
    FiniteElementData<dim> fe_data(dofs_per_object, 1, 0 /*not relevant*/);
    return fe_data;
  }
};
```

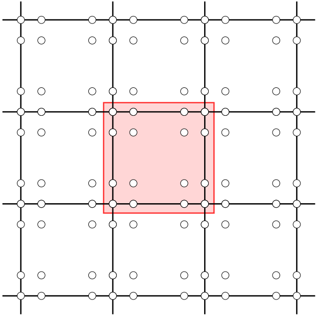
# Implementation details in deal.II (cont.)

③ custom gathering

GDM ($p = 3/5/7$)



FEM ($p = 3$)



all deal.II functions that access global matrices/vectors had to be rewritten

# Implementation details in deal.II (cont.)

4. distributed Cartesian mesh
   - lexicographic ordering of cells and DoFs allows the conversion

     $$f(i,j) = i + j * N_0 \leftrightarrow g(i) = \begin{pmatrix} i\%N_0 \\ i/N_0 \end{pmatrix}$$

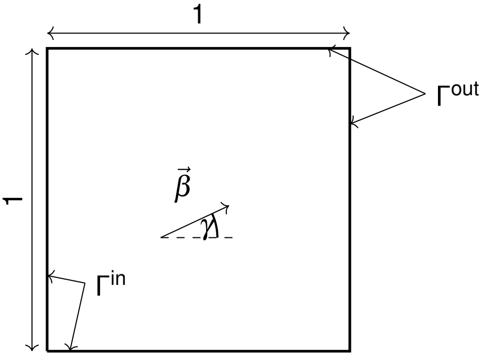
     and, as a consequence, to easily determine neighbors and patch indices
   - layer-wise partitioning with arbitrary number of ghost layers

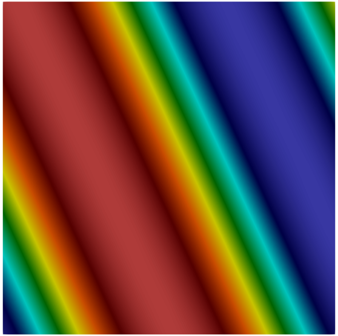
# Experiment

Solve advection equation with RK4 (CFL=$||\beta||\Delta t/h$=0.4):

$$\dot{u} + \beta \cdot \nabla u = 0 \qquad \text{in } \Omega \times (0, T)$$

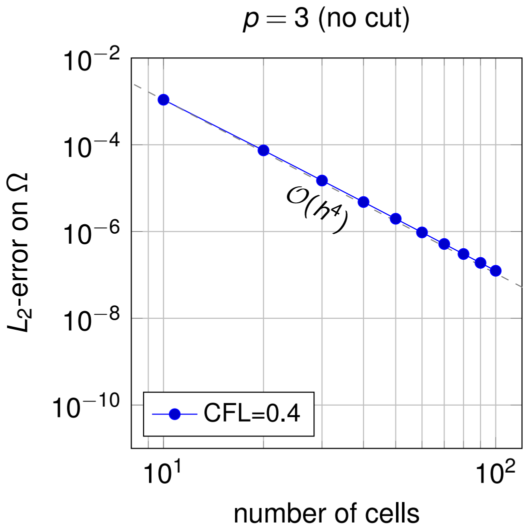
with the setup:

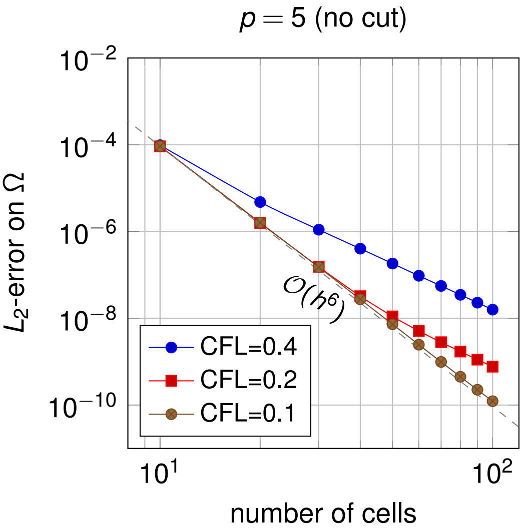


*... with prescribed boundary and initial condition.*

# Experiment (cont.)



Part 4:

# CutGDM

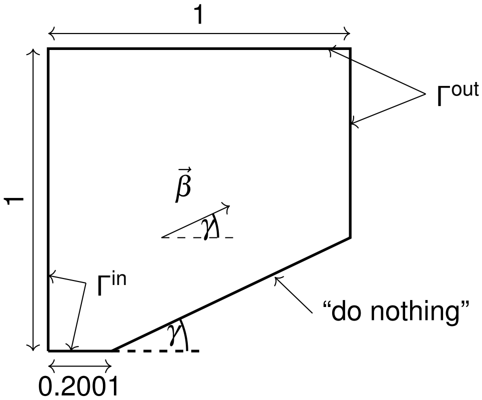
# Extension to CutGDM

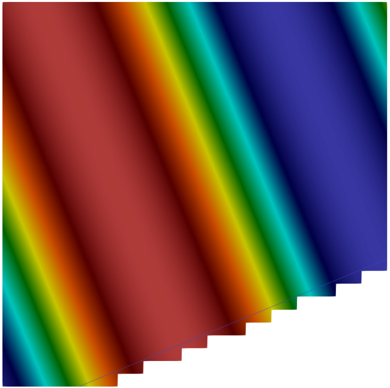
- conceptually easy since we evaluate the shape functions cell by cell
- however, missing features:
  - several classes in NonMatching namespace were not working for hp
  - several methods were only working for DoFCellAccessor but not for CellAccessor
- we use ghost penalty, but only consider gradients:

$$j(v, u_h) = \sum_{F \in \mathcal{F}_\Gamma} h^{2k+1} \langle [\partial_n v], [\partial_n u_h] \rangle$$

# Experiment

Same setup as before, but with ramp[2] (and small cuts):

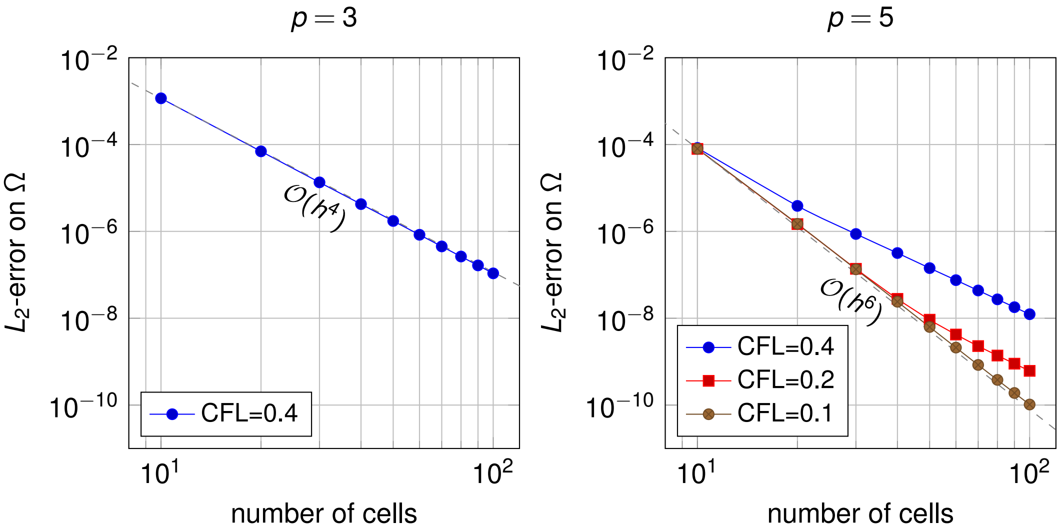


- note: similar results for non-parallel ramp

[2] C Engwer, S May, A Nüßing, F Streitbürger, 2020, A stabilized DG cut cell method for discretizing the linear transport equation, SISC.

## Experiment (cont.)




- Future work: stability analysis, A-priori error estimation, ...

Part 5:

# Outlook

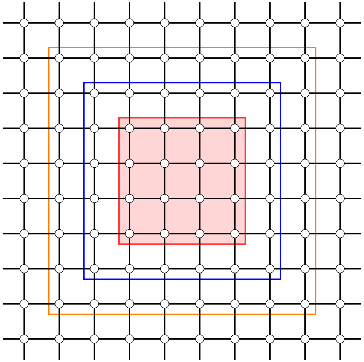
# Discussion: Cartesian meshes

Many applications do not need unstructured meshes but are satisfied/need Cartesian/structured meshes:

- Galerkin difference methods (GDM)
- immersed boundary methods (IBM)
- localized orthogonal decomposition (LOD)
- fast Fourier transform (FFT)

# Application: Galerkin difference methods (GDM)

GDM ($p = 3/5/7$)

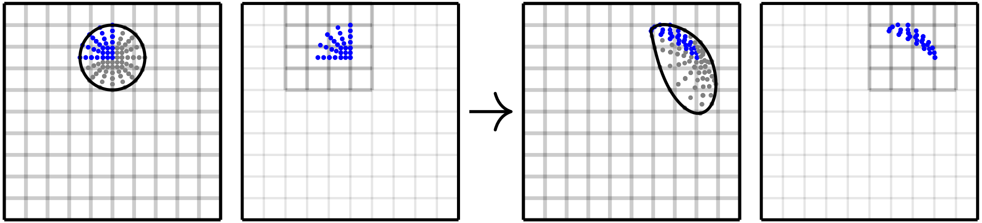


Requirements:

- get DoFs of patches with arbitrary size
- determine: inside/boundary patch

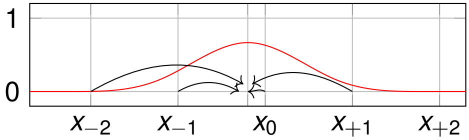
# Application: immersed boundary methods (IBM)

E.g.: circular solid in Cartesian fluid mesh ... *and, similarly, DEM-CFD coupling*

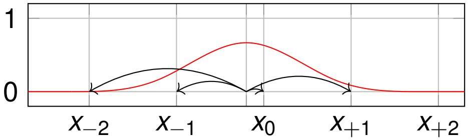


- with typical operations (4-point B-spline Dirac kernel function):

interpolation of velocity




spreading of forces



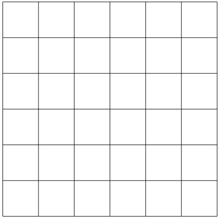

- challenges: given $\vec{x}$ determine cell, reference position, partition, communication (fast)

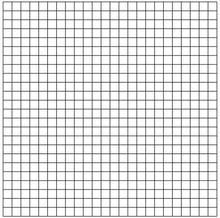
# Application: localized orthogonal decomposition (LOD)

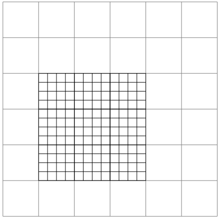
Idea: solve

$$\boldsymbol{A}_{\text{LOD}}\boldsymbol{x}_{\text{LOD}} = \boldsymbol{b}_{\text{LOD}} \qquad \text{vs.} \qquad \boldsymbol{A}_{\text{FEM}}\boldsymbol{x}_{\text{FEM}} = \boldsymbol{b}_{\text{FEM}}$$

with $\boldsymbol{x}_{\text{FEM}} \approx \boldsymbol{C}\boldsymbol{b}_{\text{LOD}}$, $\boldsymbol{b}_{\text{LOD}} = \boldsymbol{C}^\top \boldsymbol{b}_{\text{FEM}}$, $\underbrace{\boldsymbol{A}_{\text{LOD}} = \boldsymbol{C}^\top \boldsymbol{A}_{\text{FEM}}\boldsymbol{C} \text{ (Galerkin projection)}}_{\text{expensive offline phase!}}$.


coarse mesh (LOD)


fine mesh (FEM)


fine patch to compute one column of $\boldsymbol{C}$

- Cartesian background mesh; solve patch problem of arbitrary number of layers
- fine mesh/matrices/vectors never needed: however, the global indices of fine system

# Application: fast Fourier transform (FFT)

Computation of energy spectrum:

$$E(\kappa) = \frac{1}{2} \int ||\vec{u}'(\kappa)||_2^2 \mathrm{d}S(\kappa)$$

with $\vec{u}(\vec{x}) = \sum_{\vec{\kappa}} \vec{u}(\vec{\kappa}) \cos(\kappa_1 x) \cos(\kappa_2 y) \cos(\kappa_3 z)$; $\vec{\kappa}^\top = \begin{pmatrix} \kappa_1 & \kappa_2 & \kappa_3 \end{pmatrix}$; $\kappa = ||\vec{\kappa}||_2$.

To transform between physical space and wavenumber space, we used FFTW[3], which needs lexicographic (layer-wise) ordering of data $\rightarrow$ conversion of data layout

[3]Note: similar issues with coupling to other external libraries

# Challenges & requirements

- identification of a cell, based on coordinates: $(i, j, k)$ or $\vec{x}$
- identification of neighboring cells to construct patches
- determine DoFs of a patch
- parallelization:
  - lexicographical ordering of cells + layer-wise partitioning of mesh
  - permute data, e.g., via Utilities::MPI::NoncontiguousPartitioner

# Summary

- CutFEM infrastructure → SignedDistance, NM::MeshClassifier, NM::FEValues
- Hermite elements → FE_Hermite
- (Cut) Galerkin difference methods
- Cartesian meshes

# Cut Galerkin difference methods and more

**Department of Information Technology, Uppsala University, Sweden**

Peter Munch

Questions?

## References


- Banks, J.W. and Hagstrom, T., 2016. On Galerkin difference methods. Journal of Computational Physics, 313, pp.310-327.
- Bergbauer, M., Munch, P., Wall, W.A. and Kronbichler, M., 2024. High-performance matrix-free unfitted finite element operator evaluation. arXiv preprint arXiv:2404.07911.
- Carraro, T. and Wetterauer, S., 2015. On the implementation of the eXtended finite element method (XFEM) for interface problems. arXiv preprint arXiv:1507.04238.
- Engwer, C., May, S., Nüßing, A. and Streitbürger, F., 2020. A stabilized DG cut cell method for discretizing the linear transport equation. SIAM Journal on Scientific Computing, 42(6), pp.A3677-A3703.
- Saye, R.I., 2015. High-order quadrature methods for implicitly defined surfaces and volumes in hyperrectangles. SIAM Journal on Scientific Computing, 37(2), pp.A993-A1019.
- Weber, I., Kreiss, G. and Nazarov, M., 2022. Stability analysis of high order methods for the wave equation. Journal of Computational and Applied Mathematics, 404, p.113900.

Part 6:

# Appendix

## Shape functions

Hermite element ($p = 3$):

$$\phi_0(\xi) = 2\xi^3 - 3x^2 + 1$$
$$\phi_1(\xi) = -2\xi^3 + 3x^2$$
$$\phi_2(\xi) = \xi^3 - 2x^2 + x$$
$$\phi_3(\xi) = \xi^3 - x^2$$

GDM ($p = 3$):

$$\phi = \begin{cases} +\frac{1}{6}(\xi+3)(\xi+2)(\xi+1) & -2 < \xi \leq -1 \\ -\frac{1}{2}(\xi+2)(\xi+1)(\xi-1) & -1 < \xi \leq -0 \\ +\frac{1}{2}(\xi+1)(\xi-1)(\xi-2) & +0 < \xi \leq +1 \\ -\frac{1}{6}(\xi-1)(\xi-2)(\xi-3) & +1 < \xi \leq +2 \\ 0 & \text{else.} \end{cases}$$