Efficient Simulation of Fluid Flows using High-Order Finite Elements with the deal.II Library

Martin Kronbichler

Faculty of Mathematics
Ruhr University Bochum, Germany

Collaboration with Maximilian Bergbauer, Niklas Fehn, Katharina Kormann, Karl Ljungkvist, Peter Munch, Richard Schussnig, Wolfgang A. Wall, and the deal.II community

SPONSORED BY THE

deal.II

Federal Ministry of Education and Research

Funded by the European Union NextGenerationEU

EuroHPC Joint Undertaking

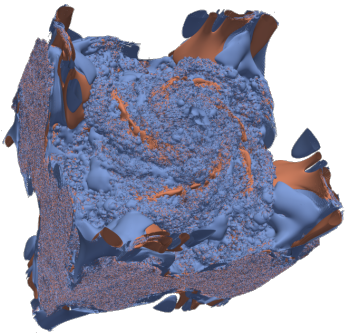
DFG Deutsche Forschungsgemeinschaft

December 17, 2024

Outline

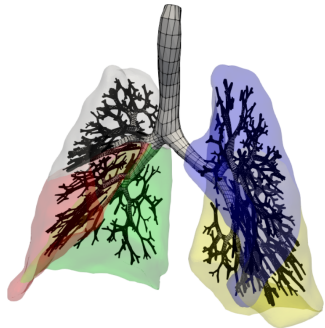
Application areas

Application: Turbulent flows



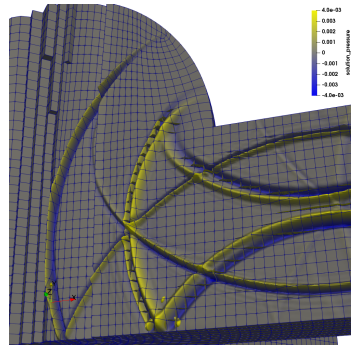
- Incompressible Navier-Stokes
- Fast solution of Poisson equation
- Multigrid

Application: Coupled flows



- Simulation of flow in biomedical settings
- Navier-Stokes coupled to transport
- Poisson + multigrid

Application: Acoustics



- Acoustic wave equation
- Explicit time stepping for DG

Model for incompressible fluid flow

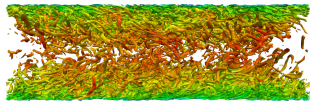
Incompressible Navier–Stokes equations

$$\frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot (\boldsymbol{u} \otimes \boldsymbol{u}) - \nabla \cdot (\nu \nabla \boldsymbol{u}) + \nabla p = \boldsymbol{f}$$

$$\nabla \cdot \boldsymbol{u} = 0$$

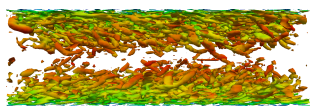
- Unknowns: velocity $\boldsymbol{u} = [u_1, u_2, u_3]$, pressure p
- Fluid kinematic viscosity ν
  - Turbulent flow: ν much smaller than $\boldsymbol{u}$
- External forces $\boldsymbol{f}$ (e.g. gravity)

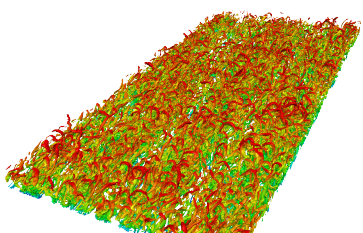
Goal: Efficient discretization method (without resolving all scales?)



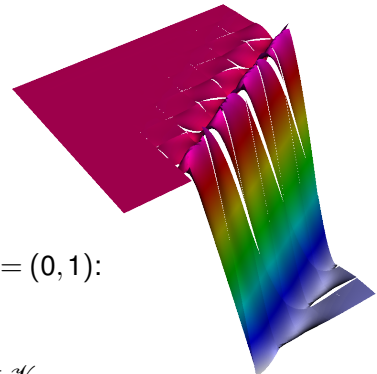
fine resolution – physical

coarse resolution – how good?





Outline



- Mesh of elements
- Polynomial solution of degree k on each element
- No continuity directly imposed between elements as in the conventional continuous FEM, but use flux functions instead
- Example for advection with constant speed a in 1D on $\Omega = (0,1)$:

$$\partial_t u + a\partial_x u = 0$$

$$\Rightarrow \quad \int_{\Omega_e} v\partial_t u \,\mathrm{d}x + \int_{\Omega_e} va\partial_x u \,\mathrm{d}x = 0 \quad \text{for all } v \in \mathcal{V}_h$$

$$\Rightarrow \quad \underbrace{\int_{\Omega_e} v\partial_t u \,\mathrm{d}x}_{(v,\partial_t u)_{\Omega_e}} - \underbrace{\int_{\Omega_e} \partial_x v au \,\mathrm{d}x}_{(\partial_x v, au)_{\Omega_e}} + \underbrace{\int_{\partial\Omega_e} v\boldsymbol{n}(au)^* \,\mathrm{d}x}_{\langle v\boldsymbol{n},(au)^*\rangle_{\partial\Omega_e}} = 0 \quad \text{for all } v \in \mathcal{V}_h$$

Here, $(au)^*$ is the numerical flux, e.g. upwind

Navier–Stokes spatial discretization: weak formulation

Find $\boldsymbol{u}_h \in \mathscr{V}_h^u$, $p_h \in \mathscr{V}_h^p$ such that for all $(\boldsymbol{v}_h, q_h) \in \mathscr{V}_h^u \times \mathscr{V}_h^p$

$$m_{h,u}\left(\boldsymbol{v}_h, \frac{\partial \boldsymbol{u}_h}{\partial t}\right) + c_h(\boldsymbol{v}_h, \boldsymbol{u}_h) + v_h(\boldsymbol{v}_h, \boldsymbol{u}_h) + g_h(\boldsymbol{v}_h, p_h) = f_h(\boldsymbol{v}_h) \ ,$$

$$-d_h(q_h, \boldsymbol{u}_h) = 0 \ .$$

DG discretization of basic operators:

- Convective operator $c_h(\boldsymbol{v}_h, \boldsymbol{u}_h)$: Local Lax–Friedrichs flux
- Viscous operator $v_h(\boldsymbol{v}_h, \boldsymbol{u}_h)$: Symmetric interior penalty method
- Gradient operator $g_h(\boldsymbol{v}_h, p_h)$: Central flux
- Divergence operator $d_h(q_h, \boldsymbol{u}_h)$: Central flux

Fehn, Wall, Kronbichler: On the stability of projection methods for the incompressible Navier–Stokes equations based on high-order discontinuous Galerkin discretizations, *J Comput Phys* 348, 2017

Navier–Stokes spatial discretization: weak formulation

Find $\boldsymbol{u}_h \in \mathscr{V}_h^u$, $p_h \in \mathscr{V}_h^p$ such that for all $(\boldsymbol{v}_h, q_h) \in \mathscr{V}_h^u \times \mathscr{V}_h^p$

$$m_{h,u}\left(\boldsymbol{v}_h, \frac{\partial \boldsymbol{u}_h}{\partial t}\right) + c_h(\boldsymbol{v}_h, \boldsymbol{u}_h) + v_h(\boldsymbol{v}_h, \boldsymbol{u}_h) + g_h(\boldsymbol{v}_h, p_h) = f_h(\boldsymbol{v}_h) \,,$$

$$-d_h(q_h, \boldsymbol{u}_h) = 0 \,.$$

DG discretization of basic operators:

- Convective operator $c_h(\boldsymbol{v}_h, \boldsymbol{u}_h)$: Local Lax–Friedrichs flux

$$c_h(\boldsymbol{v}_h, \boldsymbol{u}_h) = -(\nabla \boldsymbol{v}_h, \boldsymbol{u}_h \otimes \boldsymbol{u}_h)_{\Omega_h} + \left\langle [\![\boldsymbol{v}_h]\!], \{\{\boldsymbol{u}_h \otimes \boldsymbol{u}_h\}\} \cdot \boldsymbol{n} + \frac{\Lambda}{2} [\![\boldsymbol{u}_h]\!] \right\rangle_{\Gamma_h}$$

- Viscous operator $v_h(\boldsymbol{v}_h, \boldsymbol{u}_h)$: Symmetric interior penalty method
- Gradient operator $g_h(\boldsymbol{v}_h, p_h)$: Central flux
- Divergence operator $d_h(q_h, \boldsymbol{u}_h)$: Central flux

Fehn, Wall, Kronbichler: On the stability of projection methods for the incompressible Navier–Stokes equations based on high-order discontinuous Galerkin discretizations, *J Comput Phys* 348, 2017

Navier–Stokes spatial discretization: weak formulation

Find $\boldsymbol{u}_h \in \mathcal{V}_h^u$, $p_h \in \mathcal{V}_h^p$ such that for all $(\boldsymbol{v}_h, q_h) \in \mathcal{V}_h^u \times \mathcal{V}_h^p$

$$m_{h,u}\left(\boldsymbol{v}_h, \frac{\partial \boldsymbol{u}_h}{\partial t}\right) + c_h(\boldsymbol{v}_h, \boldsymbol{u}_h) + v_h(\boldsymbol{v}_h, \boldsymbol{u}_h) + g_h(\boldsymbol{v}_h, p_h) = f_h(\boldsymbol{v}_h) \ ,$$

$$-d_h(q_h, \boldsymbol{u}_h) = 0 \ .$$

DG discretization of basic operators:

- Convective operator $c_h(\boldsymbol{v}_h, \boldsymbol{u}_h)$: Local Lax–Friedrichs flux
- Viscous operator $v_h(\boldsymbol{v}_h, \boldsymbol{u}_h)$: Symmetric interior penalty method

$$v_h(\boldsymbol{v}_h, \boldsymbol{u}_h) = (\nabla \boldsymbol{v}_h, \nu \nabla \boldsymbol{u}_h)_{\Omega_h} - \langle \{\{\nabla \boldsymbol{v}_h\}\} \cdot \boldsymbol{n}, \nu [\![\boldsymbol{u}_h]\!] \rangle_{\Gamma_h} - \langle [\![\boldsymbol{v}_h]\!], \nu \{\{\nabla \boldsymbol{u}_h\}\} \cdot \boldsymbol{n} - \nu \tau [\![\boldsymbol{u}_h]\!] \rangle_{\Gamma_h}$$

- Gradient operator $g_h(\boldsymbol{v}_h, p_h)$: Central flux
- Divergence operator $d_h(q_h, \boldsymbol{u}_h)$: Central flux

Fehn, Wall, Kronbichler: On the stability of projection methods for the incompressible Navier–Stokes equations based on high-order discontinuous Galerkin discretizations, *J Comput Phys* 348, 2017

Navier–Stokes spatial discretization: weak formulation

Find $\boldsymbol{u}_h \in \mathscr{V}_h^u$, $p_h \in \mathscr{V}_h^p$ such that for all $(\boldsymbol{v}_h, q_h) \in \mathscr{V}_h^u \times \mathscr{V}_h^p$

$$m_{h,u}\left(\boldsymbol{v}_h, \frac{\partial \boldsymbol{u}_h}{\partial t}\right) + c_h(\boldsymbol{v}_h, \boldsymbol{u}_h) + v_h(\boldsymbol{v}_h, \boldsymbol{u}_h) + g_h(\boldsymbol{v}_h, p_h) = f_h(\boldsymbol{v}_h) \ ,$$

$$-d_h(q_h, \boldsymbol{u}_h) = 0 \ .$$

DG discretization of basic operators:

- Convective operator $c_h(\boldsymbol{v}_h, \boldsymbol{u}_h)$: Local Lax–Friedrichs flux
- Viscous operator $v_h(\boldsymbol{v}_h, \boldsymbol{u}_h)$: Symmetric interior penalty method
- Gradient operator $g_h(\boldsymbol{v}_h, p_h)$: Central flux

$$g_h(\boldsymbol{v}_h, p_h) = -(\nabla \cdot \boldsymbol{v}_h, p_h)_{\Omega_h} + \langle [\![\boldsymbol{v}_h]\!] \cdot \boldsymbol{n}, \{\{p_h\}\} \rangle_{\Gamma_h}$$

- Divergence operator $d_h(q_h, \boldsymbol{u}_h)$: Central flux

$$d_h(q_h, \boldsymbol{u}_h) = -(\nabla q_h, \boldsymbol{u}_h)_{\Omega_h} + \langle [\![q_h]\!], \{\{\boldsymbol{u}_h\}\} \cdot \boldsymbol{n} \rangle_{\Gamma_h}$$

Fehn, Wall, Kronbichler: On the stability of projection methods for the incompressible Navier–Stokes equations based on high-order discontinuous Galerkin discretizations, *J Comput Phys* 348, 2017

Navier–Stokes spatial discretization: weak formulation

Find $\boldsymbol{u}_h \in \mathcal{V}_h^u$, $p_h \in \mathcal{V}_h^p$ such that for all $(\boldsymbol{v}_h, q_h) \in \mathcal{V}_h^u \times \mathcal{V}_h^p$

$$m_{h,u}\left(\boldsymbol{v}_h, \frac{\partial \boldsymbol{u}_h}{\partial t}\right) + c_h(\boldsymbol{v}_h, \boldsymbol{u}_h) + v_h(\boldsymbol{v}_h, \boldsymbol{u}_h) + g_h(\boldsymbol{v}_h, p_h) = f_h(\boldsymbol{v}_h) \ ,$$

$$-d_h(q_h, \boldsymbol{u}_h) = 0 \ .$$

DG discretization of basic operators:

- Convective operator $c_h(\boldsymbol{v}_h, \boldsymbol{u}_h)$: Local Lax–Friedrichs flux
- Viscous operator $v_h(\boldsymbol{v}_h, \boldsymbol{u}_h)$: Symmetric interior penalty method
- Gradient operator $g_h(\boldsymbol{v}_h, p_h)$: Central flux
- Divergence operator $d_h(q_h, \boldsymbol{u}_h)$: Central flux

Computation of integrals: Numerical quadrature with $(k_u+1)^d$ points for viscous and mass matrix terms, $\left(\left\lfloor \frac{3}{2} k_u \right\rfloor + 1\right)^d$ points for convection (non-linearity)

Fehn, Wall, Kronbichler: On the stability of projection methods for the incompressible Navier–Stokes equations based on high-order discontinuous Galerkin discretizations, *J Comput Phys* 348, 2017

Application to under-resolved turbulent flows

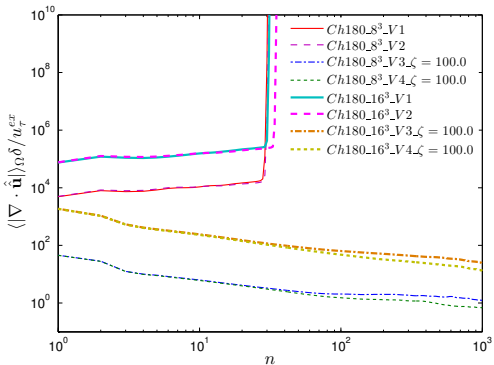
Questions:

- Is this basic discretization approach robust in the under-resolved regime?
- Do we need additional ingredients?

We have:

- DG robust for dominating convective terms, $\frac{h\|\boldsymbol{u}\|}{(k+1)\nu} > 1$
- $k_u = k_p + 1 \rightarrow$ inf–sup stable

Numerical test in underresolved regime with $\frac{\nu}{\|\boldsymbol{u}\|_{\max}} = 0.00029$:
$\rightarrow$ **Severe instabilities**



Robust scheme for all viscosities ν? Energy method

Mathematics help to assess stability of the scheme

$$\frac{1}{2}\frac{\mathrm{d}\|\boldsymbol{u}_h\|^2}{\mathrm{d}t} = \left(\boldsymbol{u}_h, \frac{\partial \boldsymbol{u}_h}{\partial t}\right)_{\Omega_h}$$

Insert PDE: viscous term dissipates energy, pressure gradient cancels due to continuity equation $\rightarrow$ convective term left

$$\frac{1}{2}\frac{\mathrm{d}\|\boldsymbol{u}_h\|^2}{\mathrm{d}t} \leq -\left\langle [\![\boldsymbol{u}_h]\!], \frac{\Lambda}{2}[\![\boldsymbol{u}_h]\!] \right\rangle_{\Gamma_h^{\mathrm{int}}} - \frac{1}{2}(\nabla\cdot\boldsymbol{u}_h, \boldsymbol{u}_h\cdot\boldsymbol{u}_h)_{\Omega_h} + \frac{1}{2}\left\langle [\![\boldsymbol{u}_h]\!]\cdot\boldsymbol{n}, \boldsymbol{u}_h^- \cdot \boldsymbol{u}_h^+ \right\rangle_{\Gamma_h^{\mathrm{int}}}$$

- Lax–Friedrichs (upwind) term dissipates energy
- **Divergence errors** might produce energy
- **Discontinuities of the velocity** in normal direction across interior faces between elements might produce energy
- Both violating terms tend to zero as discretization is refined, but resolution requirement unrealistic for practical flows with $\mathrm{Re} > 10{,}000$

Robustness for under-resolved turbulent flows

- Robust method in under-resolved regime $\mathrm{Re}_h = \frac{h\|\boldsymbol{u}\|}{k\nu} > 1$ must dominate problematic terms. Possible realization by **consistent penalty terms**:

$$m_{h,u}\left(\boldsymbol{v}_h, \frac{\partial \boldsymbol{u}_h}{\partial t}\right) + c_h(\boldsymbol{v}_h, \boldsymbol{u}_h) + v_h(\boldsymbol{v}_h, \boldsymbol{u}_h) + g_h(\boldsymbol{v}_h, p_h) + a_{\mathrm{D}}(\boldsymbol{v}_h, \boldsymbol{u}_h) + a_{\mathrm{C}}(\boldsymbol{v}_h, \boldsymbol{u}_h) = f_h^e(\boldsymbol{v}_h)\ ,$$

$$-d_h(q_h, \boldsymbol{u}_h) = 0\ ,$$

divergence penalty: $a_{\mathrm{D}}(\boldsymbol{v}_h, \boldsymbol{u}_h) = (\nabla \cdot \boldsymbol{v}_h, \tau_{\mathrm{D}} \nabla \cdot \boldsymbol{u}_h)_{\Omega_h}\ ,$

continuity penalty: $a_{\mathrm{C}}(\boldsymbol{v}_h, \boldsymbol{u}_h) = \left\langle [\![\boldsymbol{v}_h]\!] \cdot \boldsymbol{n}, \tau_{\mathrm{C},f} [\![\boldsymbol{u}_h]\!] \cdot \boldsymbol{n} \right\rangle_{\partial\Omega_e \setminus \Gamma_h}\ .$

- Penalty terms mimic properties of $H(\mathrm{div})$ conforming spaces with exact fulfillment of $\nabla \cdot \boldsymbol{u} = 0$ condition, like Raviart–Thomas elements

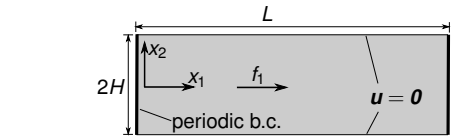
Fehn, Wall, Kronbichler: Robust and efficient discontinuous Galerkin methods for under-resolved turbulent incompressible flows, *J Comput Phys* 372, 2018

Fehn, Kronbichler, Lehrenfeld, Lube, Schroeder: High-order DG solvers for underresolved turbulent incompressible flows: A comparison of L_2 and $H(\mathrm{div})$ methods, *Int J Numer Meth Fluids* 91, 2019

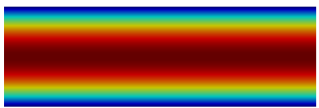
Fehn, Kronbichler, Lube: From anomalous dissipation through Euler singularities to stabilized finite element methods for turbulent flows, *submitted*, 2024, https://doi.org/10.21203/rs.3.rs-4187657/v1

Orr–Sommerfeld stability problem

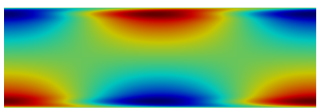
- Geometry, initial condition given by perturbation of magnitude 10^{-5}, $\mathrm{Re} = 7{,}500$



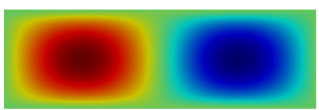
$u_1 \approx U_1$



$u_1 - U_1 \ll u_1$



$u_2 \ll u_1$

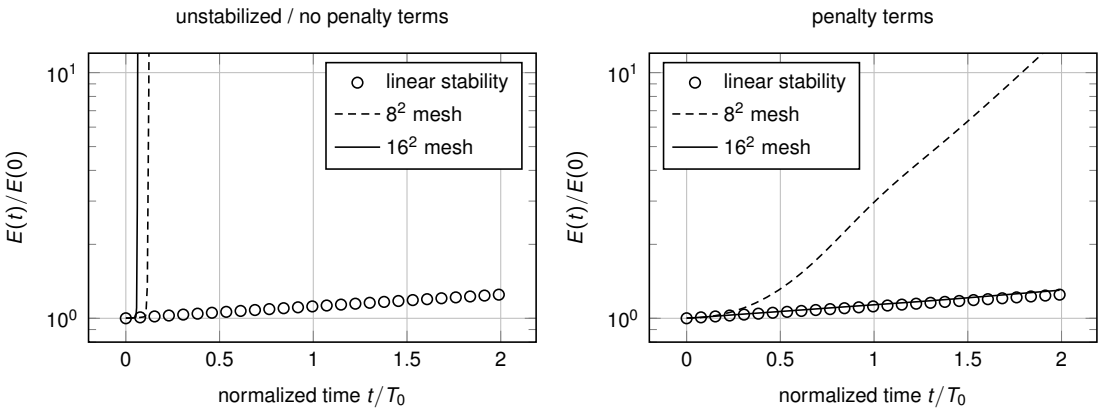


Verification: Orr–Sommerfeld stability problem

Monitored quantities:

- perturbation energy $E = \int_\Omega \|\boldsymbol{u} - \boldsymbol{U}\|^2 \mathrm{d}\boldsymbol{x}$
- linear stability theory: $E(t)/E(t=0) = \exp(2\omega_i t)$

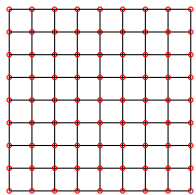
Stability of numerical LES approach for $k = 3$ at two resolution levels



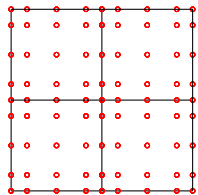
Outline

Efficient discretizations

- Desire 1: Low number of degrees of freedom $\rightarrow$ high accuracy per unknown
- Desire 2: Methods applicable to general geometries
- Challenge: High-Reynolds number incompressible flows develop fine-scale features
  - Need to track solution over long times
  - Need accurate “dispersive” behavior
- Solution: High-order finite element methods
  - Geometrically flexible by use unstructured meshes
  - Hexahedral meshes preferred
  - High-order methods add unknowns inside elements
  - Range of attractive degrees: $p = 3, \ldots, 7$
  - Discontinuous Galerkin especially attractive: upwind fluxes



degree 1, 8^2 mesh



degree 4, 2^2 mesh

Efficient time discretization: Splitting method

BDF of order $J = 1, 2, 3$ within dual splitting method:[1]

1. Explicit convective step

$$\gamma_0 \hat{\boldsymbol{u}} - \sum_{i=0}^{J-1} \alpha_i \boldsymbol{u}^{n-i} = \Delta t \left(\sum_{i=0}^{J-1} \beta_i \nabla \cdot (\boldsymbol{u}^{n-i} \otimes \boldsymbol{u}^{n-i}) + \boldsymbol{f}^{n+1} \right)$$

2. Pressure step $\quad -\nabla^2 p^{n+1} = -\dfrac{\gamma_0}{\Delta t} \nabla \cdot \hat{\boldsymbol{u}}$

   Boundary condition for high-order accuracy in time

$$\nabla p^{n+1} \cdot \boldsymbol{n} = -\left(\frac{\partial \boldsymbol{g}_u(t^{n+1})}{\partial t} + \sum_{i=0}^{J-1} \beta_i \left(\nabla \cdot (\boldsymbol{u}^{n-i} \otimes \boldsymbol{u}^{n-i}) + \nu \nabla \times (\nabla \times \boldsymbol{u}^{n-i}) \right) - \boldsymbol{f}^{n+1} \right) \cdot \boldsymbol{n}$$

3. Projection step $\quad \hat{\hat{\boldsymbol{u}}} = \hat{\boldsymbol{u}} - \dfrac{\Delta t}{\gamma_0} \nabla p^{n+1}$

4. Implicit viscous step $\quad \dfrac{\gamma_0}{\Delta t} \left(\boldsymbol{u}^{n+1} - \hat{\hat{\boldsymbol{u}}} \right) = \nu \nabla \cdot \nabla \boldsymbol{u}^{n+1}$ (Helmholtz-like equation)

[1] Karniadakis, Israeli, Orszag, High-order splitting methods for the incompressible Navier–Stokes equations. *JCP* 97(2), 1991

Test case for accuracy: vortex flow

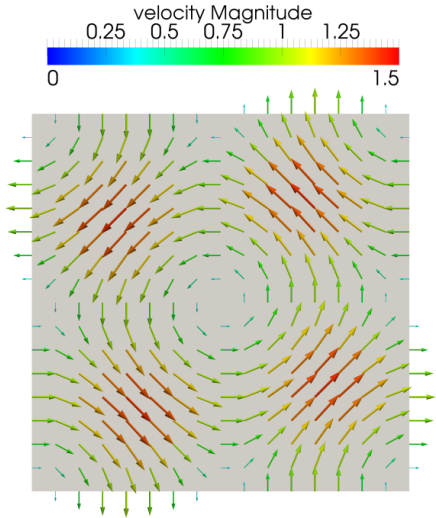
- Analytic solution in $\Omega = (-0.5, 0.5)^2$

$$\boldsymbol{u}(\boldsymbol{x}, t) = \begin{pmatrix} -\sin(2\pi x_2) \\ \sin(2\pi x_1) \end{pmatrix} \mathrm{e}^{-4\nu\pi^2 t}$$

$$p(\boldsymbol{x}, t) = -\cos(2\pi x_1)\cos(2\pi x_2)\mathrm{e}^{-8\nu\pi^2 t}$$

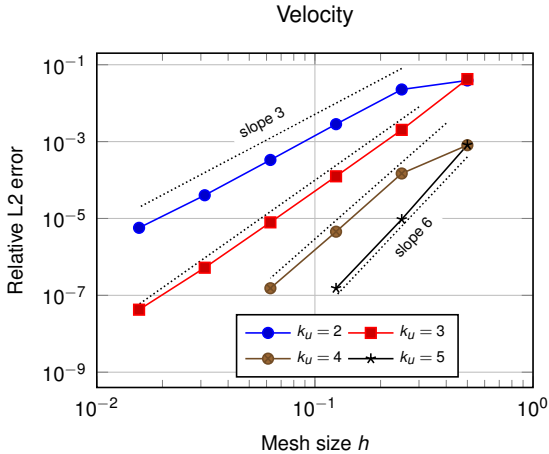
- Initial condition $\boldsymbol{u}(\cdot, t = 0) = \boldsymbol{u}_0$
- Boundary conditions:
  - Dirichlet $\boldsymbol{u} = \boldsymbol{g}_{\boldsymbol{u}}$ on inflow
  - Neumann $(-p\boldsymbol{I} + 2\nu\varepsilon(\boldsymbol{u})) \cdot \boldsymbol{n} = \boldsymbol{h}$ on outflow

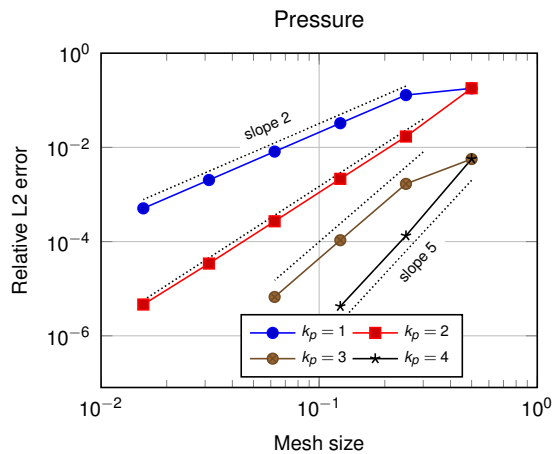
2D vortex flow



Convergence for vortex problem

Vortex problem with 2^2 to 64^2 elements with polynomial degrees $(k_u, k_p) = (2,1), \ldots, (5,4)$, BDF3, $\Delta t = 10^{-4}$

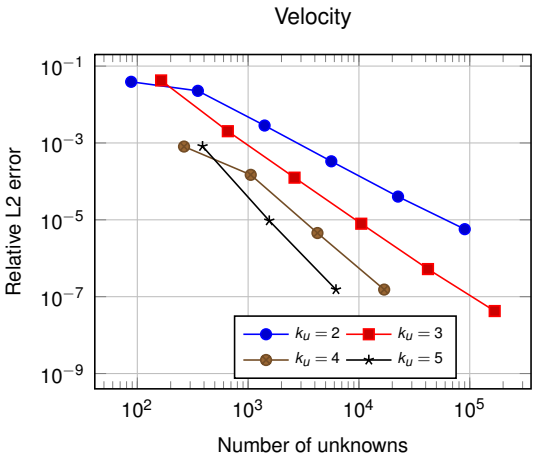


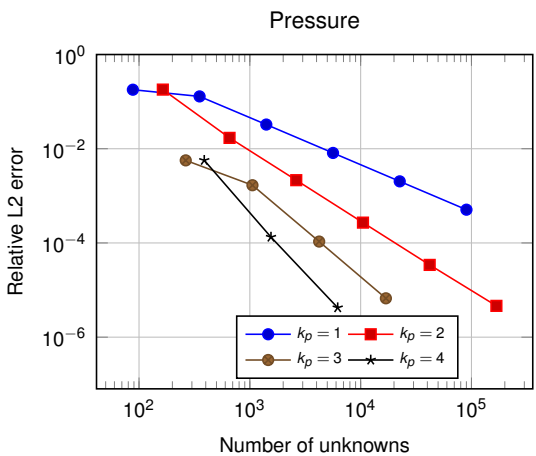


All scenarios converge with optimal order $(k+1)$!

Convergence for vortex problem

Same data as before, but using number of unknowns instead of h





From the number of unknowns to simulation cost

- For smooth problem with good resolution, **high order methods clearly advantageous**
- What about time step size for higher orders?

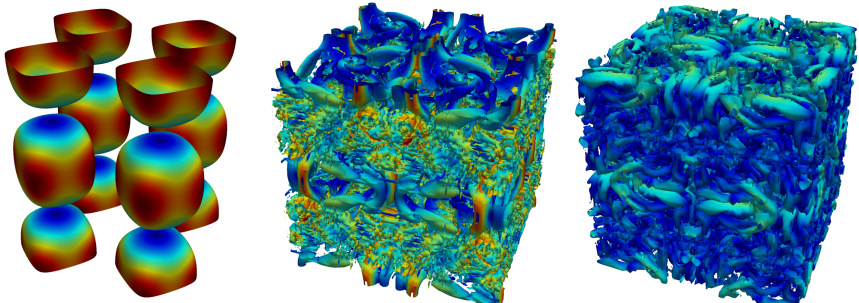
$$\Delta t \lesssim \frac{h}{k^{1.5} \|\boldsymbol{u}\|_\infty}$$

- What about cost per unknown to solve pressure Poisson equation?
- What about convergence rates for realistic problems?

Typical situation in turbulence: Underresolved flows

3D Taylor–Green vortex at $\mathrm{Re} = 1600$, i.e., $\nu = \frac{1}{1600}$

- Periodic box, side length 2π
- Dual splitting scheme, BDF2
- Visualize on 16^3 mesh and $(k_u, k_p) = (7, 6)$ (128^3 effective resolution)
- Iso-contours of q-criterion (value of 0.1) colored by velocity magnitude



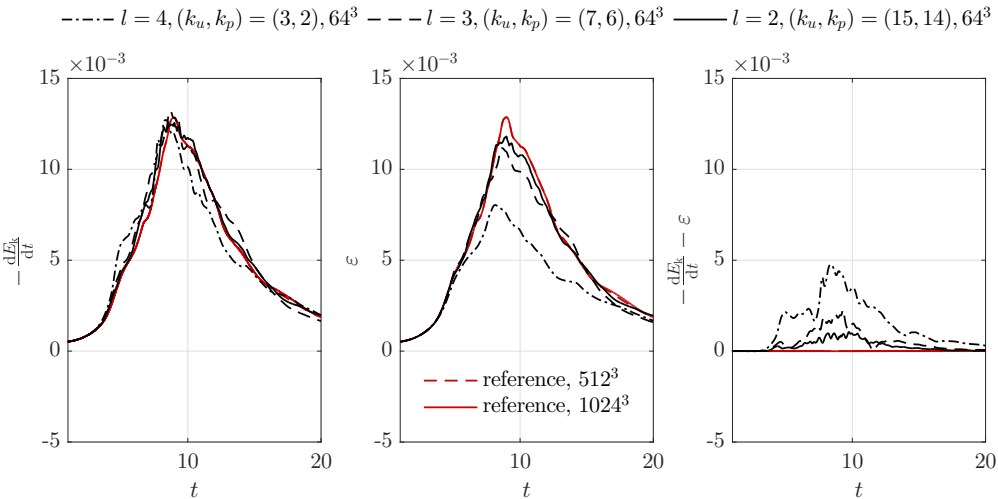
$t = 0$ $t = T/2$ $t = T$

Observed quantities in 3D Taylor–Green vortex

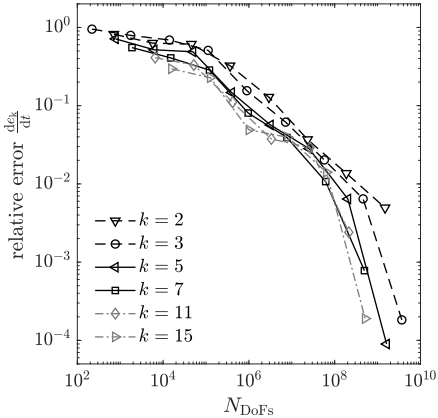
- Kinetic energy $E_\mathrm{k} = \frac{1}{2V_\Omega} \int_\Omega |\boldsymbol{u}_h|^2 \,\mathrm{d}\boldsymbol{x}$ and kinetic energy dissipation rate $\frac{\mathrm{d}E_\mathrm{k}}{\mathrm{d}t} = \frac{E_\mathrm{k}^{i+1} - E_\mathrm{k}^{i-1}}{t_{i+1} - t_{i-1}}$
- Molecular dissipation (dissipation of resolved scales) $\varepsilon = \frac{\nu}{V_\Omega} \int_\Omega \nabla \boldsymbol{u}_h : \nabla \boldsymbol{u}_h \,\mathrm{d}\boldsymbol{x}$
- Numerical dissipation $\varepsilon_\mathrm{num} = -\frac{\mathrm{d}E_\mathrm{k}}{\mathrm{d}t} - \varepsilon$

3D Taylor–Green vortex problem

- Dissipation rates: **divergence and continuity penalty terms** for Taylor–Green vortex problem at $\mathrm{Re} = 1600$, effective resolution 64^3



Taylor–Green vortex: Accuracy over unknowns (DoFs)



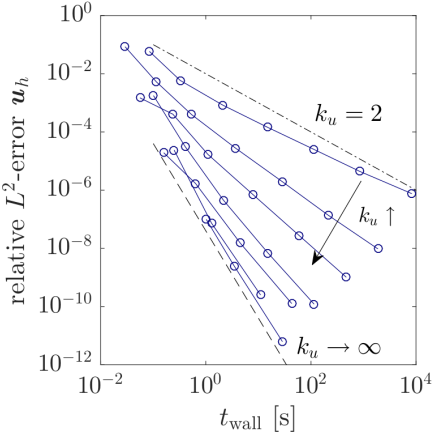
- Higher orders are better, but not much
- Asymptotic regime starts only beyond 10^8 DoFs[2]

[2] Fehn, Wall, Kronbichler: Efficiency of high-performance discontinuous Galerkin spectral element methods for under-resolved turbulent incompressible flows. *Int. J. Numer. Meth. Fluids* 88, 2018

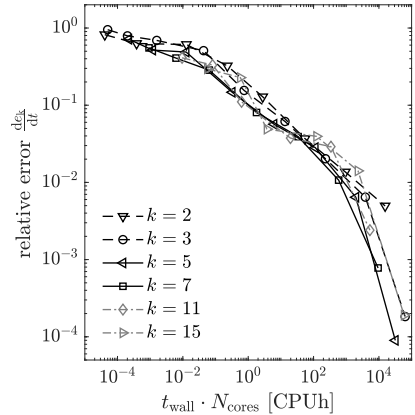
Efficient simulation: error versus compute time

Conclusion: high order brings down resolution requirement by at most a factor of 10 in 3D: need efficient implementation with **similar cost per unknown** as low-order method

2D vortex: **well-resolved**

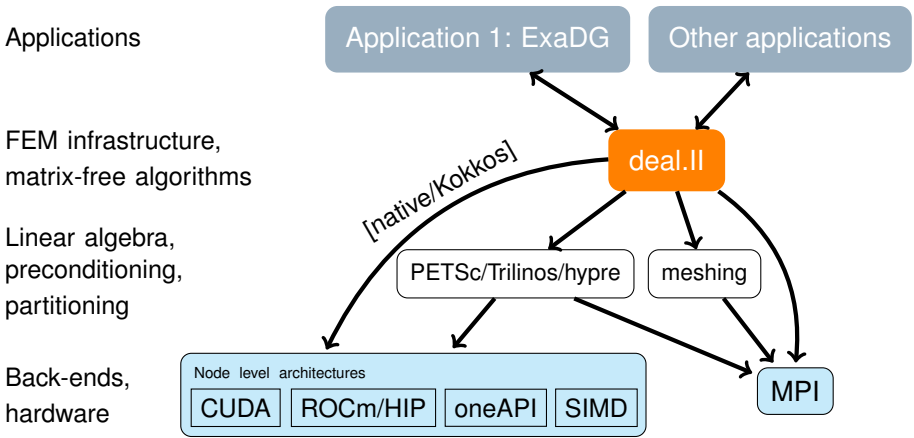


3D Taylor–Green: **under-resolved**



Outline

Design philosophy: Composition of libraries



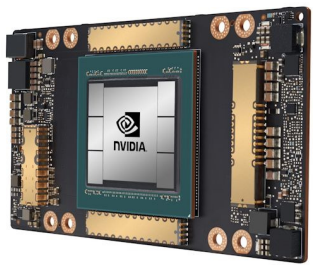
Code activities in my group – library approach

CFD application code development with ExaDG

- Implementation of CFD equations
  - Based on appropriate data structures from deal.II library
- Simulation control based on library components
  - Iterative solvers, multigrid suited for equations at hand
  - Embeds equation evaluator into existing interfaces

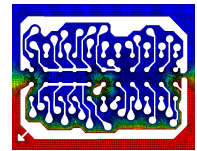
Code development within deal.II finite element library

- Building blocks targeting particular hardware (SIMD, CUDA)
- MPI parallelization of mesh
- Interface to linear algebra libraries like Trilinos/PETSc
- Move common functionality to deal.II library
  - Separation of concerns
  - Forces split of application-specific vs generic FEM toolbox
  - Reuse among many applications with more resources

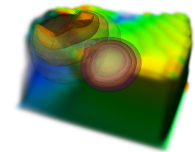


The deal.II finite element library

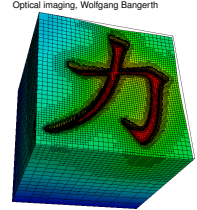
- A C++ software library to ease the development of adaptive finite element codes on HPC systems
- Name origin: Differential Equations Analysis Library
- Homepage: `www.dealii.org`
- Code hosted on `https://github.com/dealii/dealii`
- Provides many continuous, discontinuous, H^{curl} and H^{div} conforming finite elements
- Pre- and post-processing with many formats and external tools
- Linear algebra in deal.II partly implemented directly, partly via PETSc, Trilinos, LAPACK and many other packages
- Parallelization with MPI, shared memory



Parasitic conductivities & impedances, Yuhan Zhou



Optical imaging, Wolfgang Bangerth



Plastic deformation, Joerg Frohne

Efficient implementation: Matrix-free methods

RUHR UNIVERSITÄT BOCHUM RUB

Matrix-based algorithm in finite elements

- Loop through cells and assemble from cell matrices into global sparse matrix
- Apply action of sparse matrix in iterative solver (CG, GMRES, ...)
- Build preconditioner from sparse matrix

Matrix-free algorithm

- Memory access is expensive, computations cheap
- Apply action of discretized operator within iterative solver on the fly
- Redundantly compute FEM integrals
- Preconditioner selection more restricted (ILU not possible, must choose specific ingredients)

Matrix-vector product

matrix-based:

$$\begin{cases} A = \sum_{e=1}^{N_{\text{el}}} G_e^{\mathsf{T}} A_e G_e & \text{(assembly)} \\ v = Au & \text{(matrix-vector product within iterative solver)} \end{cases}$$

matrix-free:

$$v = \sum_{e=1}^{N_{\text{el}}} G_e^{\mathsf{T}} A_e \left(G_e u \right)$$

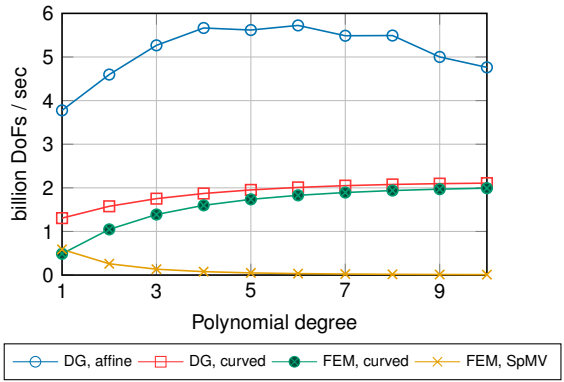
Included in `deal.II` finite element library, `www.dealii.org`

M. Kronbichler, K. Kormann, A generic interface for parallel finite element operator application. *Comput. Fluids* 63:135–147, 2012
M. Kronbichler, K. Kormann, Fast matrix-free evaluation of discontinuous Galerkin finite element operators. *ACM TOMS* 45(3), 29, 2019

Matrix-free vs. matrix-based methods

- Performance of matrix-vector product essential for iterative solvers
- Sparse matrices unsuitable for higher orders $p \geq 2$ on modern hardware due to **memory-bandwidth** limit
- Matrix-free algorithm successful in trading computations for less memory transfer
  - Software: Specify operation at quadrature points
  - Combine with reference cell interpolation matrices
  - Indirect access into vector entries for continuous FEM

Throughput of matrix-vector product (unknowns processed per second) of 3D Laplacian



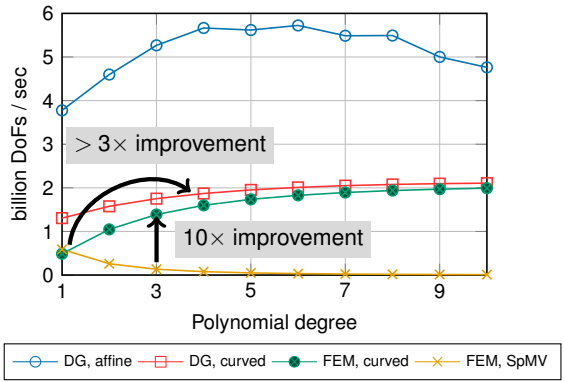
System: 1 node of 2×24 cores of Intel Xeon Platinum 8174 (SuperMUC-NG)
Memory bw: 205 GB/s, arithmetic peak 3.5 TFlop/s

Kronbichler, Kormann: A generic interface for parallel cell-based finite element operator application. *Comput Fluids* 63:135–147, 2012
Kronbichler, Wall: A performance comparison of continuous and discontinuous Galerkin methods with fast multigrid solvers. *SISC* 40(5):A3423–48, 2018
Kronbichler, Kormann: Fast matrix-free evaluation of discontinuous Galerkin finite element operators. *ACM TOMS* 45(3):29/1–40, 2019

Matrix-free vs. matrix-based methods

- Performance of matrix-vector product essential for iterative solvers
- Sparse matrices unsuitable for higher orders $p \geq 2$ on modern hardware due to **memory-bandwidth** limit
- Matrix-free algorithm successful in trading computations for less memory transfer
  - Software: Specify operation at quadrature points
  - Combine with reference cell interpolation matrices
  - Indirect access into vector entries for continuous FEM

Throughput of matrix-vector product (unknowns processed per second) of 3D Laplacian



System: 1 node of 2 × 24 cores of Intel Xeon Platinum 8174 (SuperMUC-NG)
Memory bw: 205 GB/s, arithmetic peak 3.5 TFlop/s

Kronbichler, Kormann: A generic interface for parallel cell-based finite element operator application. *Comput Fluids* 63:135–147, 2012
Kronbichler, Wall: A performance comparison of continuous and discontinuous Galerkin methods with fast multigrid solvers. *SISC* 40(5):A3423–48, 2018
Kronbichler, Kormann: Fast matrix-free evaluation of discontinuous Galerkin finite element operators. *ACM TOMS* 45(3):29/1–40, 2019

Outline

Matrix-free solvers and preconditioners

- Matrix-free approach provides fast operator evaluation (matrix-vector product), but implicit solvers need preconditioners
- For Poisson-type problems, multigrid with selected smoothers works well
  - Point-Jacobi Chebyshev smoothers[3] or overlapping Schwarz smoothers with fast diagonalization techniques[4 5] well-established
- For other operators in Navier–Stokes, situation is less clear
- Analyze **penalty step**:

$$(\boldsymbol{v}_h, \boldsymbol{u}_h)_{\Omega_h} + (\nabla \cdot \boldsymbol{v}_h, \tau_{\mathrm{D}} \nabla \cdot \boldsymbol{u}_h)_{\Omega_h} + \left\langle [\![\boldsymbol{v}_h]\!] \cdot \boldsymbol{n}, \tau_{\mathrm{C},f} [\![\boldsymbol{u}_h]\!] \cdot \boldsymbol{n} \right\rangle_{\partial \Omega_e \setminus \Gamma_h} = (\boldsymbol{v}_h, \widetilde{\boldsymbol{u}}_h)_{\Omega_h}$$

- Similar ingredients needed for augmented-Lagrangian techniques of coupled Navier–Stokes solver
  - But we prefer splitting methods for computational efficiency

[3] Kronbichler, Ljungkvist: Multigrid for matrix-free high-order finite element computations on graphics processors. *ACM Trans. Parallel Comput.* 6, 2019

[4] P. Munch and M. Kronbichler, Cache-optimized and low-overhead implementations of additive Schwarz methods for high-order FEM multigrid computations. *IJHPCA*, 38(3):192–209, 2024

[5] M. Wichrowski, P. Munch, M. Kronbichler, and G. Kanschat. Smoothers with localized residual computations for geometric multigrid method. *SISC*, accepted, 2024

Efficient solvers for penalty step

$$(\boldsymbol{v}_h, \boldsymbol{u}_h)_{\Omega_h} + (\nabla \cdot \boldsymbol{v}_h, \tau_{\mathrm{D}} \nabla \cdot \boldsymbol{u}_h)_{\Omega_h} + \left\langle [\![\boldsymbol{v}_h]\!] \cdot \boldsymbol{n}, \tau_{\mathrm{C},f} [\![\boldsymbol{u}_h]\!] \cdot \boldsymbol{n} \right\rangle_{\partial\Omega_e \setminus \Gamma_h} = (\boldsymbol{v}_h, \widetilde{\boldsymbol{u}}_h)_{\Omega_h}$$

- Matrix system is mostly block-diagonal across elements apart from continuity penalty

$$\begin{pmatrix} M & & & \\ & M & & \\ & & \ddots & \\ & & & M \end{pmatrix} + \begin{pmatrix} A_{\mathrm{D}} & & & \\ & A_{\mathrm{D}} & & \\ & & \ddots & \\ & & & A_{\mathrm{D}} \end{pmatrix} + \begin{pmatrix} A_{\mathrm{C,i}} & A_{\mathrm{C,e}} & & \\ A_{\mathrm{C,e}} & A_{\mathrm{C,i}} & A_{\mathrm{C,e}} & \\ & \ddots & \ddots & \ddots \\ & & A_{\mathrm{C,e}} & A_{\mathrm{C,i}} \end{pmatrix}$$

- Challenge:
  - Choose large-ish parameters τ_{D} and τ_{C} to ensure good conservation
  - Matrix possesses many small eigenvalues related to unconstrained DoFs, large eigenvalues for penalized DoFs via singular matrices A_{D} and A_C
- Our approach: Precondition by **block-diagonal matrix** $\left(M + A_{\mathrm{D}} + A_{C,i}\right)^{-1}$

Efficient solver for element block

Find cheap element-wise matrix inverse expression despite ill-conditioning

$$\left(M + A_{\mathrm{D}} + A_{C,i}\right)^{-1}$$

- Exact inverse would be too expensive
  - For degree $p = 4$, have $125 \times 3 = 375$ degrees of freedom $\rightarrow$ $100\times$ slower than matrix-free operator evaluation
- Choose simple preconditioner M^{-1}
  - Cheap to apply[6]
  - Would work surprisingly well for $(M + A_{\mathrm{D}})^{-1}$ because all eigenvalues to unconstrained modes exactly 1 $\rightarrow$ CG method finds two clusters of eigenvalues and is fast
  - However, $A_{\mathrm{C,i}}$ destroys nice property
- Develop new solver for combined terms

[6] M. Kronbichler, S. Schoeder, C. Müller, W. A. Wall, Comparison of implicit and explicit hybridizable discontinuous Galerkin methods for the acoustic wave equation. *IJNME* 106(9), 2016.

Efficient solver for element block II

Constant-coefficient form of matrix in 2D with Kronecker products:

$$M^{2D} + A_{\mathrm{D}}^{2D} + A_{C,i}^{2D}$$
$$= \begin{pmatrix} M_y \otimes M_x & 0 \\ 0 & M_y \otimes M_x \end{pmatrix} + \begin{pmatrix} S_y \otimes D_x \\ D_y \otimes S_x \end{pmatrix} \begin{pmatrix} S_y^\top \otimes D_x^\top & D_y^\top \otimes S_x^\top \end{pmatrix} + \begin{pmatrix} M_y \otimes G_x & 0 \\ 0 & G_y \otimes M_x \end{pmatrix},$$

with

- M_x, M_y 1D mass matrices,
- S_x, S_y one-dimensional shape values with square root of quadrature weight such that $M_i = S_i S_i^T$,
- D_x, D_y 1D derivatives of shape values times penalty factor, and
- G_x, G_y jump penalty matrix

Strategy: Collect matrix as

$$\begin{pmatrix} M_y \otimes A_x & 0 \\ 0 & A_y \otimes M_x \end{pmatrix} + \begin{pmatrix} S_y \otimes D_x \\ D_y \otimes S_x \end{pmatrix} \begin{pmatrix} S_y^\top \otimes D_x^\top & D_y^\top \otimes S_x^\top \end{pmatrix}$$

Efficient solver for element block III

Aim to cheaply form inverse of matrix

$$\underbrace{\begin{pmatrix} M_y \otimes A_x & 0 \\ 0 & A_y \otimes M_x \end{pmatrix}}_{B} + \underbrace{\begin{pmatrix} S_y \otimes D_x \\ D_y \otimes S_x \end{pmatrix} \begin{pmatrix} S_y^\top \otimes D_x^\top & D_y^\top \otimes S_x^\top \end{pmatrix}}_{CC^\top}$$

- Matrix inversion of

$$\begin{pmatrix} M_y \otimes A_x & 0 \\ 0 & A_y \otimes M_x \end{pmatrix}^{-1}$$

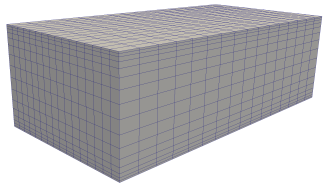
  cheaply formed by fast diagonalization method[7]
- Use Sherman–Morrison–Woodbury formula to account for singular divergence matrix

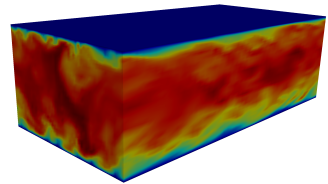
$$\left(B + CC^\top\right)^{-1} = B^{-1} - B^{-1} C \left(I - C^\top B^{-1} C\right)^{-1} C^\top B^{-1}$$

  - All factors in this expression are tensor products or sum of tensor products, with cheap inversion via fast diagonalization method

[7] R. E. Lynch, J. R. Rice, and D. H. Thomas. Direct solution of partial difference equations by tensor product methods. *Numer. Math.*, 6(1), 1964

Evaluation on channel flow



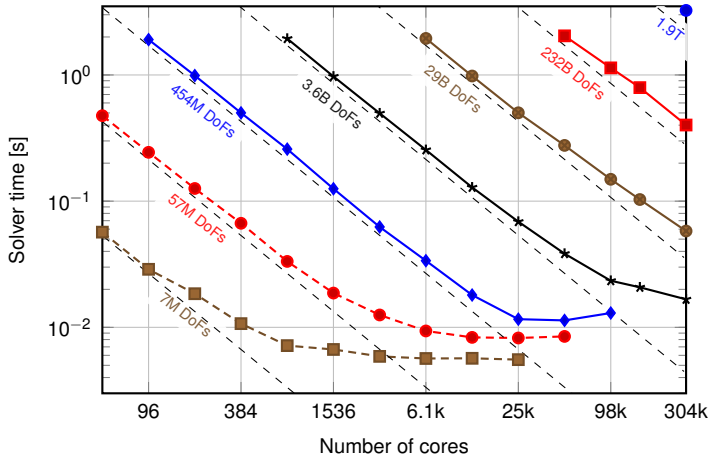


Iteration count for degree $p = 5$, penalty factors $10 \cdot \|\boldsymbol{u}\|$,

- No preconditioner: CG does not converge in 1,000 iterations
- Mass matrix preconditioner: 19.9 iterations/solve over 100k time steps, throughput per CG iteration: **2.8 GDoF/s**
- Proposed preconditioner: 4.7 iterations/solve over 100k time steps, throughput per CG iteration on node with 128 cores (AMD Zen 3): **2.4 GDoF/s**

Extreme-scale parallel multigrid solvers

- Matrix-free ingredients
- Chebyshev smoother
- Mixed precision
- *h*-multigrid (different meshes), *p*-multigrid (different polynomial degrees)
- Tuning for node-level performance and scalability to supercomputer scale



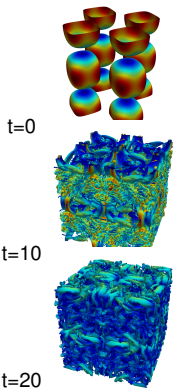
Arithmetic performance of geometric multigrid with 1.9 trillion DoFs on SuperMUC-NG (#64 of top500.org, 11/2024): **5.8 PFlop/s**, 180 GB/s per node

Arndt, Fehn, Kanschat, Kormann, Kronbichler, Munch, Wall, Witte: ExaDG: High-order discontinuous Galerkin for the exa-scale. In H.-J. Bungartz et al. (Eds.), *Software for Exascale Computing – SPPEXA 2016–2019*, 2020

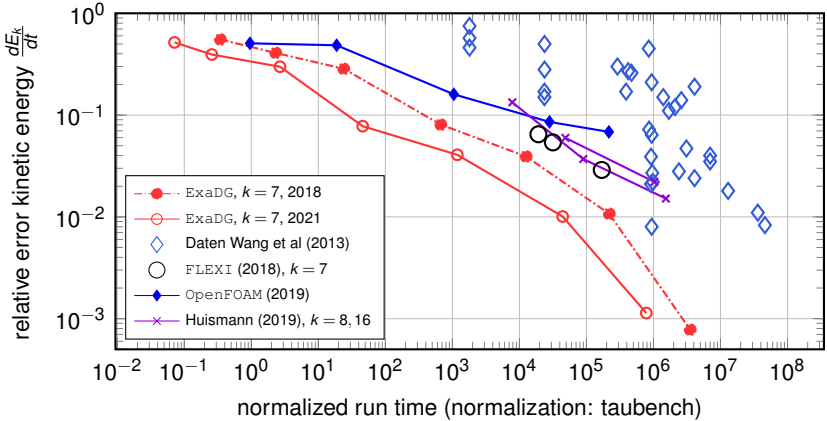
Kronbichler, Ljungkvist: Multigrid for matrix-free high-order finite element computations on graphics processors. *ACM Trans. Parallel Comput.* 6, 2019

Impact of matrix-free algorithms on CFD application

3D Taylor–Green vortex at Re = 1600: iso-contours of q-criterion (value 0.1) colored by velocity magnitude



Result of my research group around **10x faster** than all results of Wang et al. (2013), normalized run time



Wang, Fidkowski et al., High-order CFD methods: current status and perspective, *Int. J. Numer. Meth. Fluids* 72(8), 2013

Fehn, Wall, Kronbichler, Efficiency of high-performance discontinuous Galerkin spectral element methods for under-resolved turbulent incompressible flows, *Int. J. Numer. Meth. Fluids* 88, 2018

Outline

Simulation on complicated geometries: CutFEM

- Hexahedral elements most efficient for sum-factorization algorithms
- However, all-hex mesh generation is challenging
  - Often bad element qualities with bad CFL conditions
- Alternative: Use **unfitted meshes** on interface immersed to domain
- We consider level set description of interface
- Consider problems where near-boundary region is crucial
  - Combine adaptive mesh refinement with high-order methods

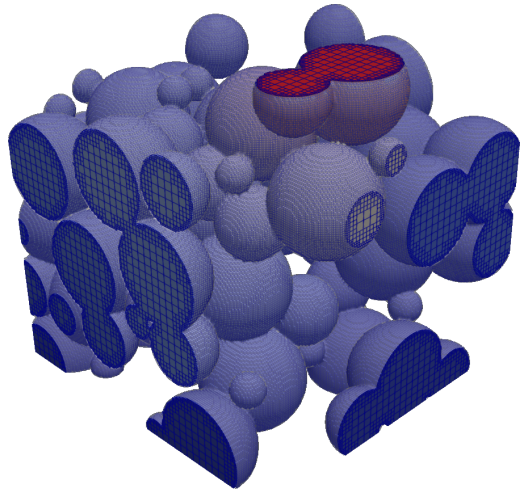
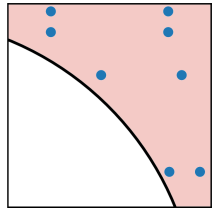


Image by Maximilian Bergbauer

Integration on cut elements and faces

- Cut approach necessitates integration on parts of cells and faces only
- Use approach by Saye[3]
- Sum factorization with cost $\mathcal{O}(k^{d+1})$ per cell relies on tensor product of shape functions and quadrature points
- Impossible on irregular points of cuts
- Must resort to generic evaluation of cost $\mathcal{O}(k^{2d})$ per cell
  - Separate interpolation for each point



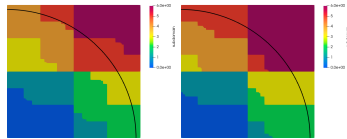
Comparison of approximate computational effort (operations / DoF), where $n_{\text{DoF}} = n_{\text{cells}}(k+1)^d(d+1)$

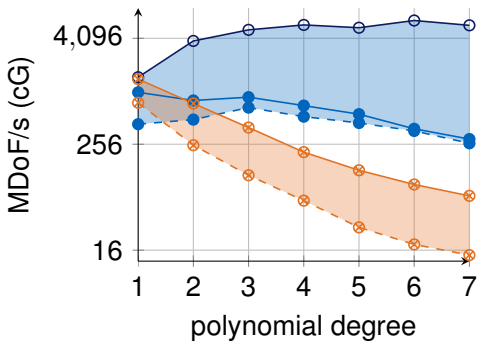
pol. degree $p = k-1$	1	2	3	4	5	6
2D transport, sum fact	60	50	60	60	70	70
2D transport, generic	120	300	450	600	800	1000
3D transport, sum fact	90	80	90	90	100	110
3D transport, generic	300	700	1300	2200	3400	5000

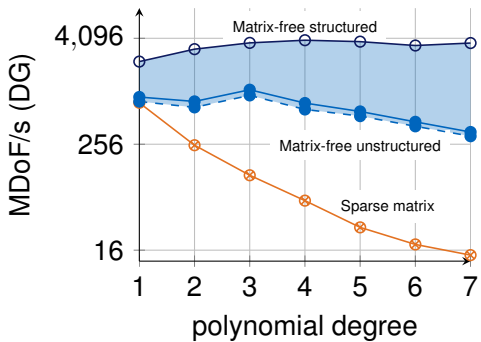
[8] Saye (2015): High-order quadrature methods for implicitly defined surfaces and volumes in hyperrectangles. *SIAM J. Sci. Comput.* 37(2):A993–A1019

Matrix-free evaluation for CutFEM

- Sum factorization on regular elements
- Generic integration on cut cells
- Domain: 2D circle / 3D ball immersed



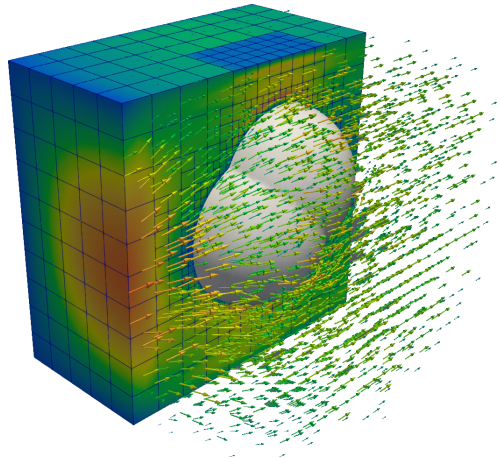




M. Bergbauer, P. Munch, W. A. Wall, M. Kronbichler, High-performance matrix-free unfitted finite element operator evaluation, arXiv preprint arXiv:2404.07911, 2024

CutFEM on complicated geometries

- Fluid simulations on cut geometries promising
- Ghost penalty stabilization to account for small-cut problem
- BDF time integration
- Solvers: Multigrid solvers in pressure Poisson and momentum operator
- Challenge: Need direct solver of cut elements
  - block Jacobi or overlapping Schwarz with element-wise patches
- Work with M. Bergbauer, W. A. Wall



Outline

Summary

- High-order methods attractive in fluid dynamics
- Correct mathematical ingredients essential: $H(\mathrm{div})$ conforming solutions
- Mathematical developments must be combined with good implementations to deliver full potential
- Matrix-free finite element algorithms good target for high performance computing
- Iterative solvers need specific preconditioners