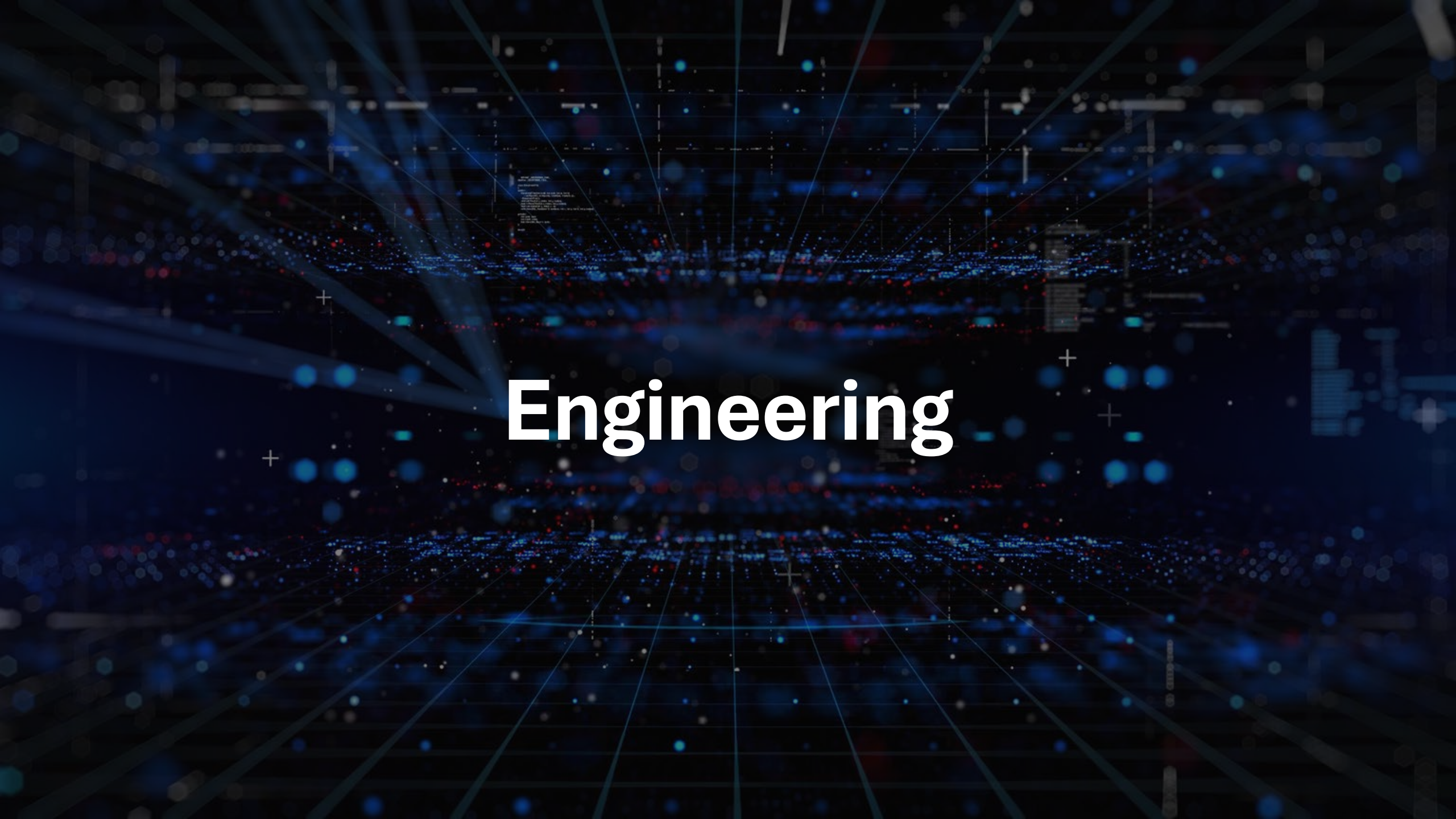




Engineering

AN EFFICIENT MULTIGRID SOLVER FOR FINITE ELEMENT METHODS ON MULTI-GPU SYSTEMS

Chris Richardson, Joe Dean, Igor Baratta, Garth Wells,
Adrian Jackson

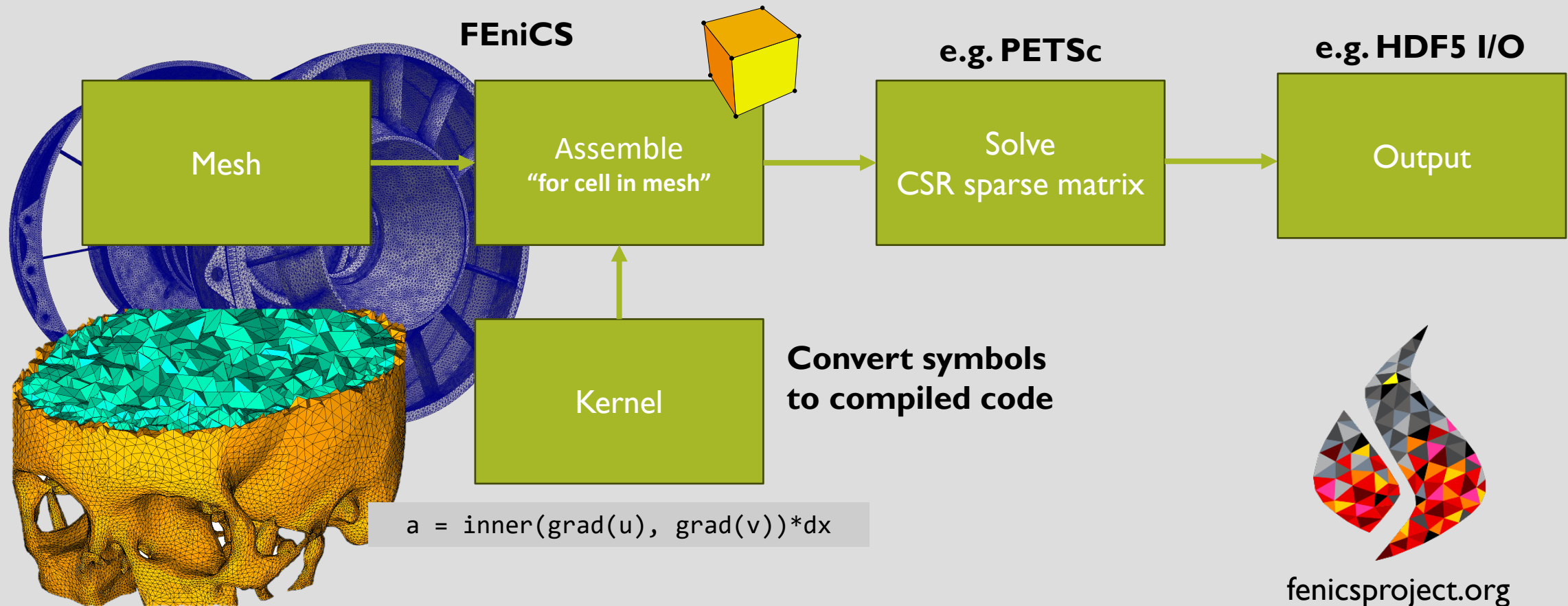
cnr12@cam.ac.uk



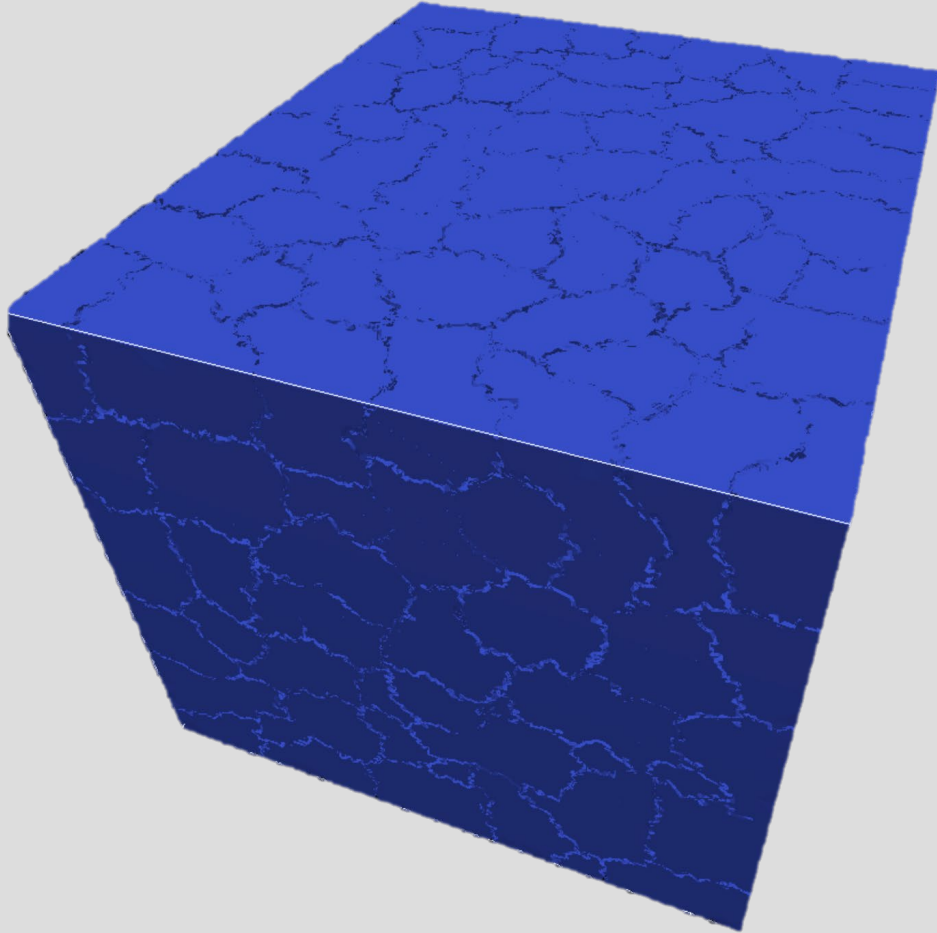
UNIVERSITY OF
CAMBRIDGE



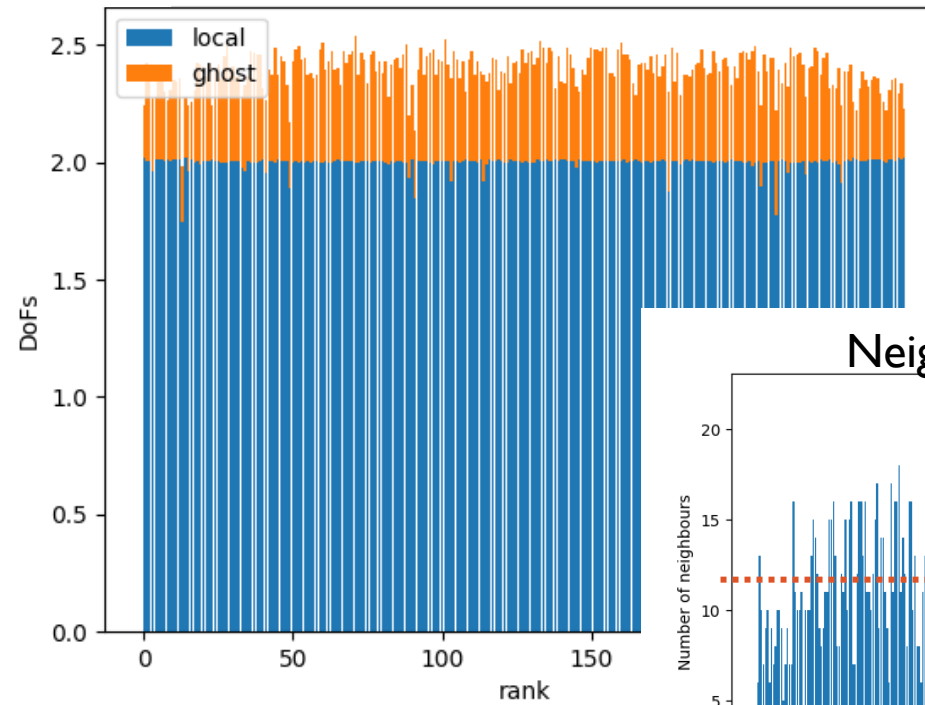
FENICS FINITE ELEMENT ON HPC



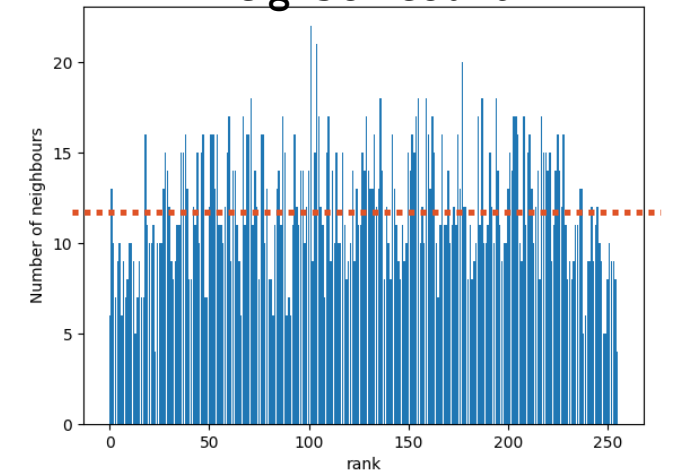
PARTITIONING AND HALO



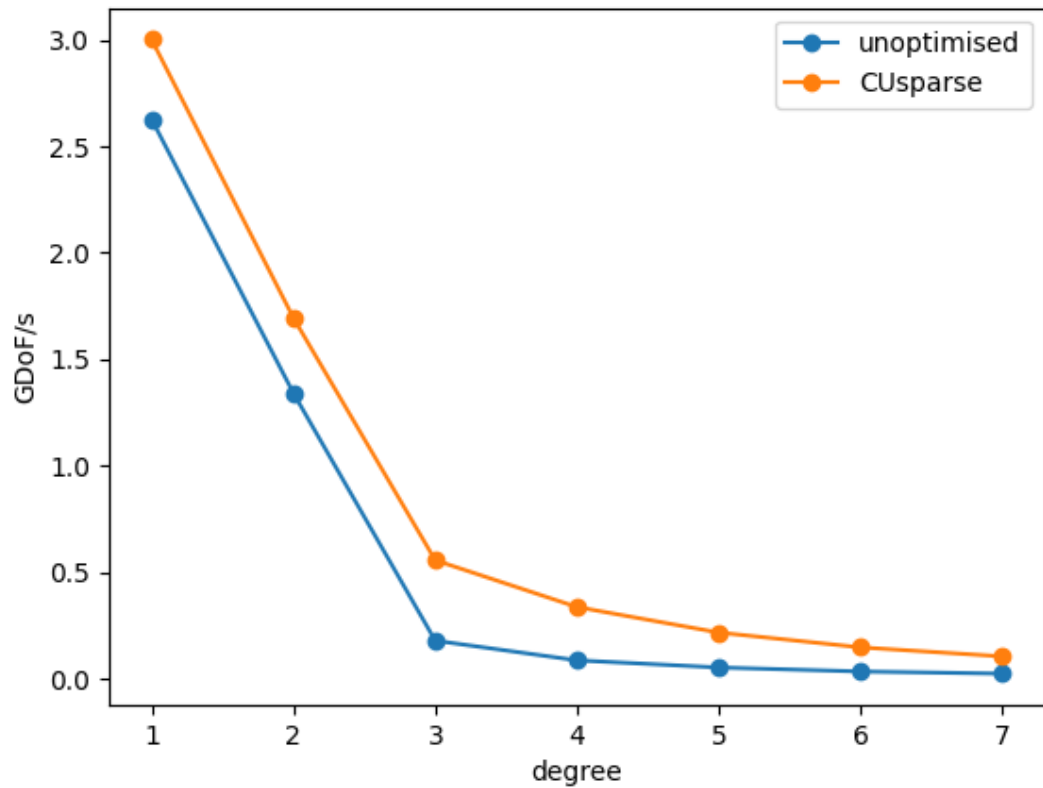
$\times 10^7$ Degrees of freedom per GCD



Neighbor count

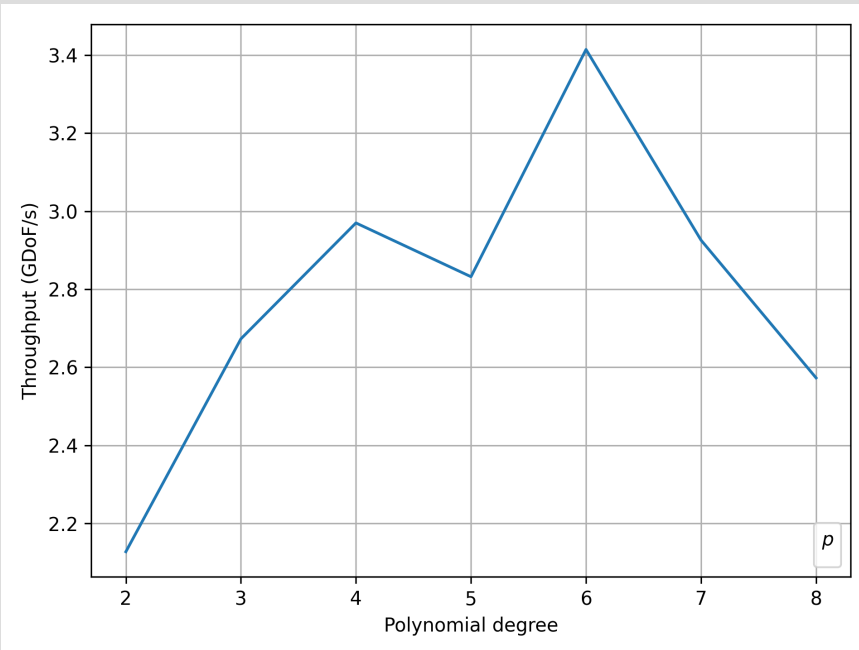


THROUGHPUT: SPARSE MATRIX ON GPU



Compute time is very slow, and memory usage high, for increasing polynomial degree

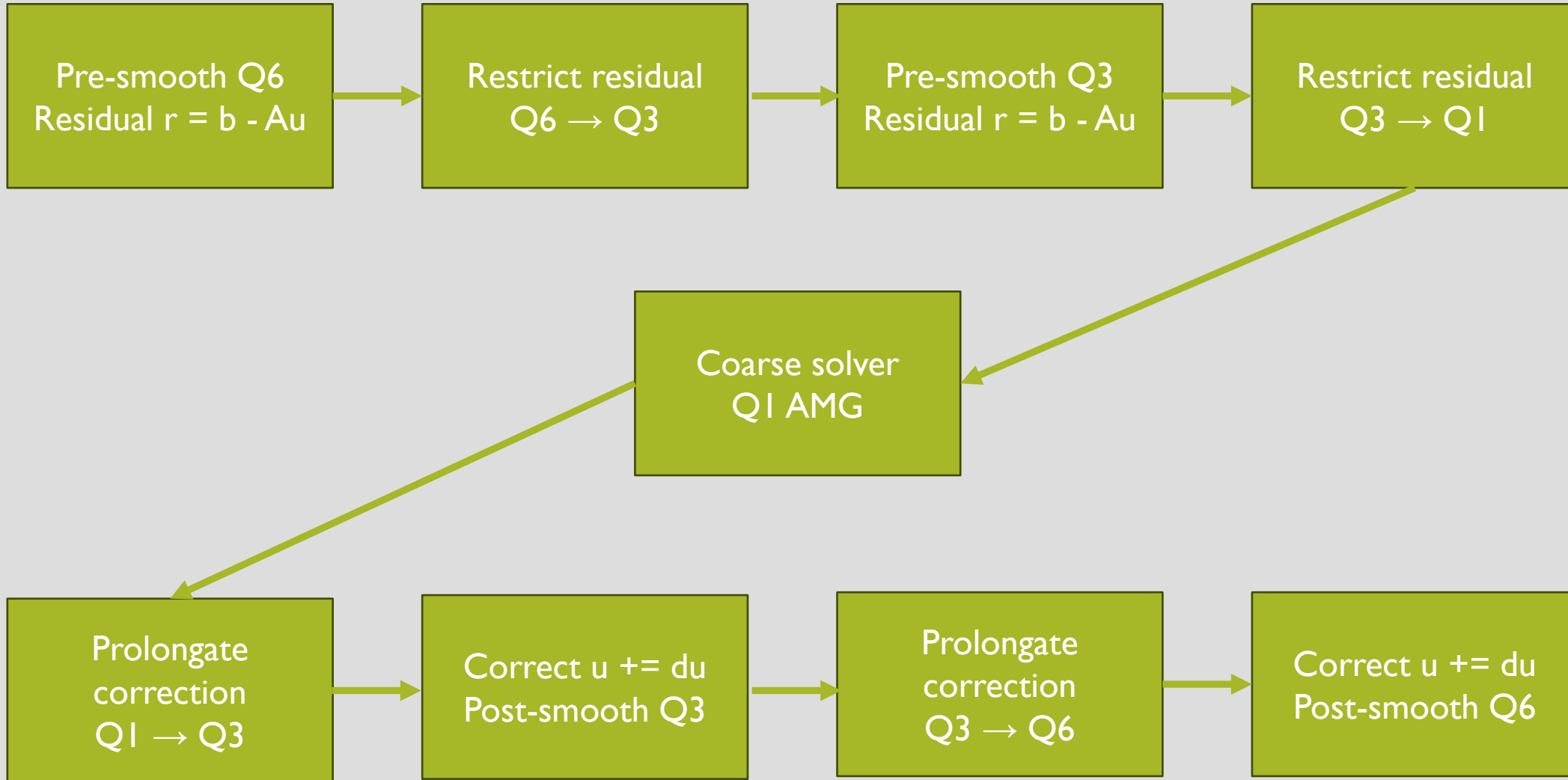
THROUGHPUT: MATRIX-FREE ON GPU



- Replace CSR Matrix with “Matrix-free” action
- Performance throughput peaks around $p=6$
- Memory usage is much less than CSR

Max throughput for matrix-free hexahedra
Laplace operator on LUMI-G MI250x GCD

MULTIGRID CYCLE



Coarse level “Q1”

Fine level “Q6”

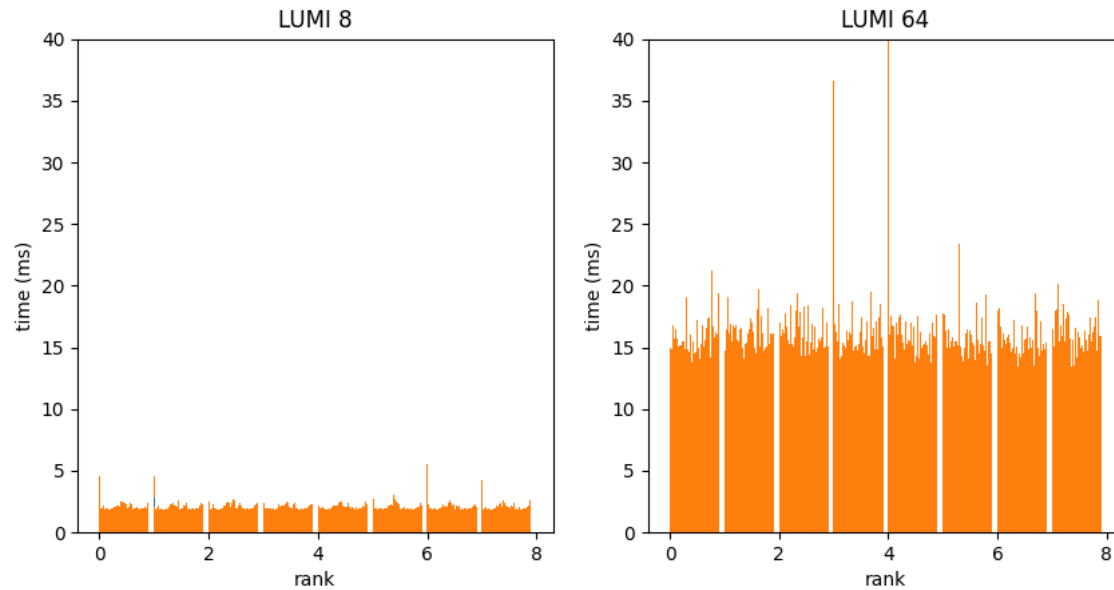
Mid level “Q3”

Implemented **GPU** Kernels

- Laplace Operator Q6, Q3
- Prolongation/Restriction
- Geometry pre-computation
- “Matrix” Diagonal operator

- AXPY
- Pointwise multiplication
- Inner product
- Pack/unpack for MPI

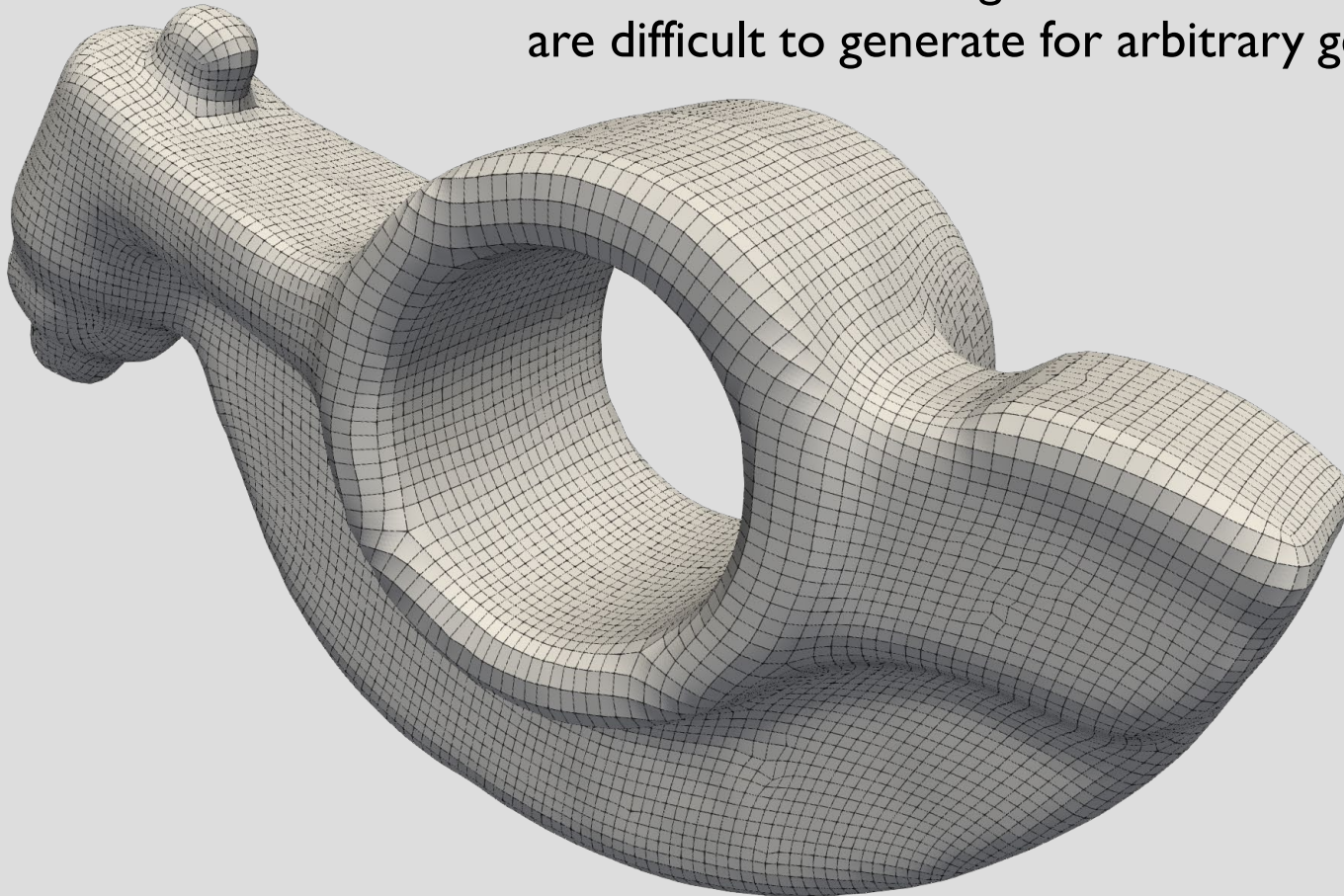
HALO NEIGHBOR UPDATE TIMINGS



Scalability is limited by hardware:
Neighbor update gets slower as
problem size increases (even
though number of neighbours is
constant)

MESH GEOMETRY

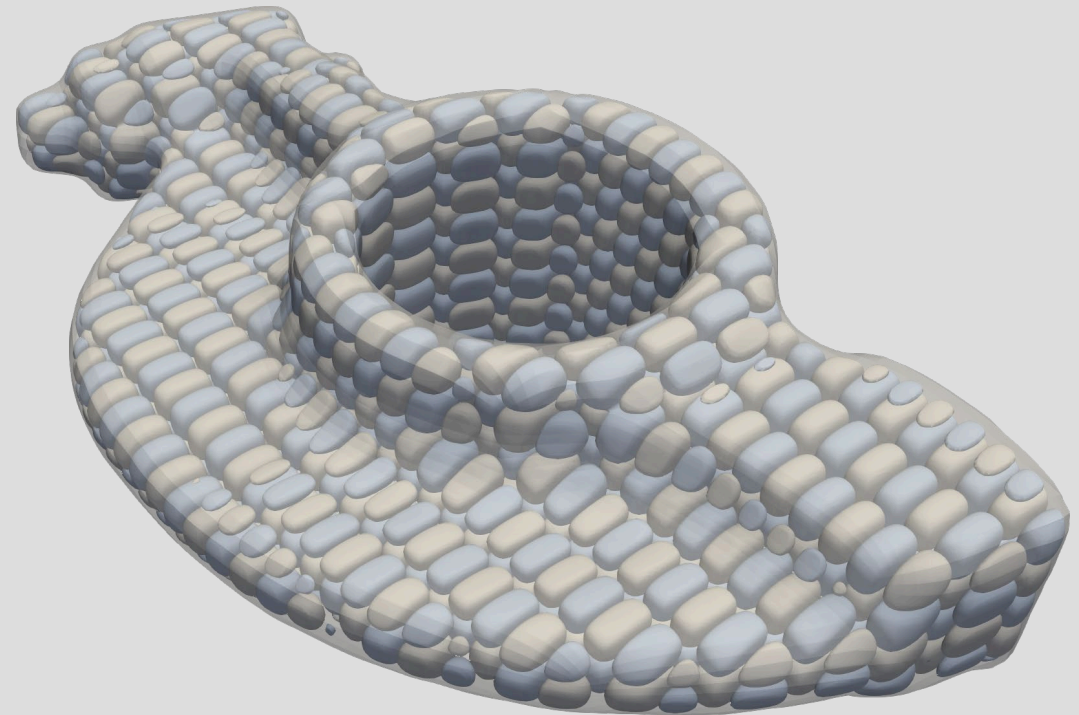
“Matrix-free” GPU algorithm works best with hexahedral meshes – but these are difficult to generate for arbitrary geometries



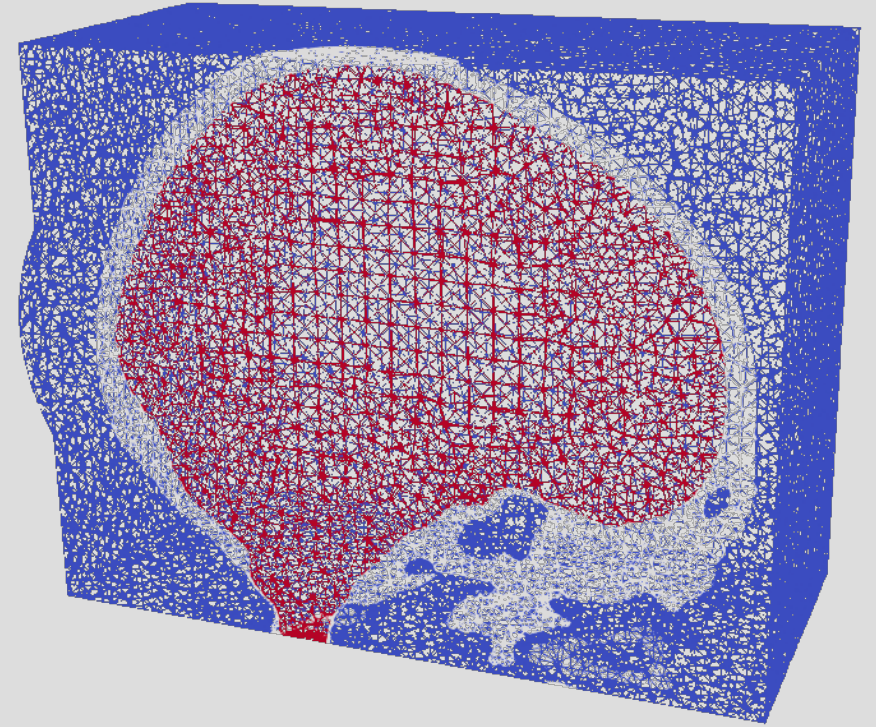
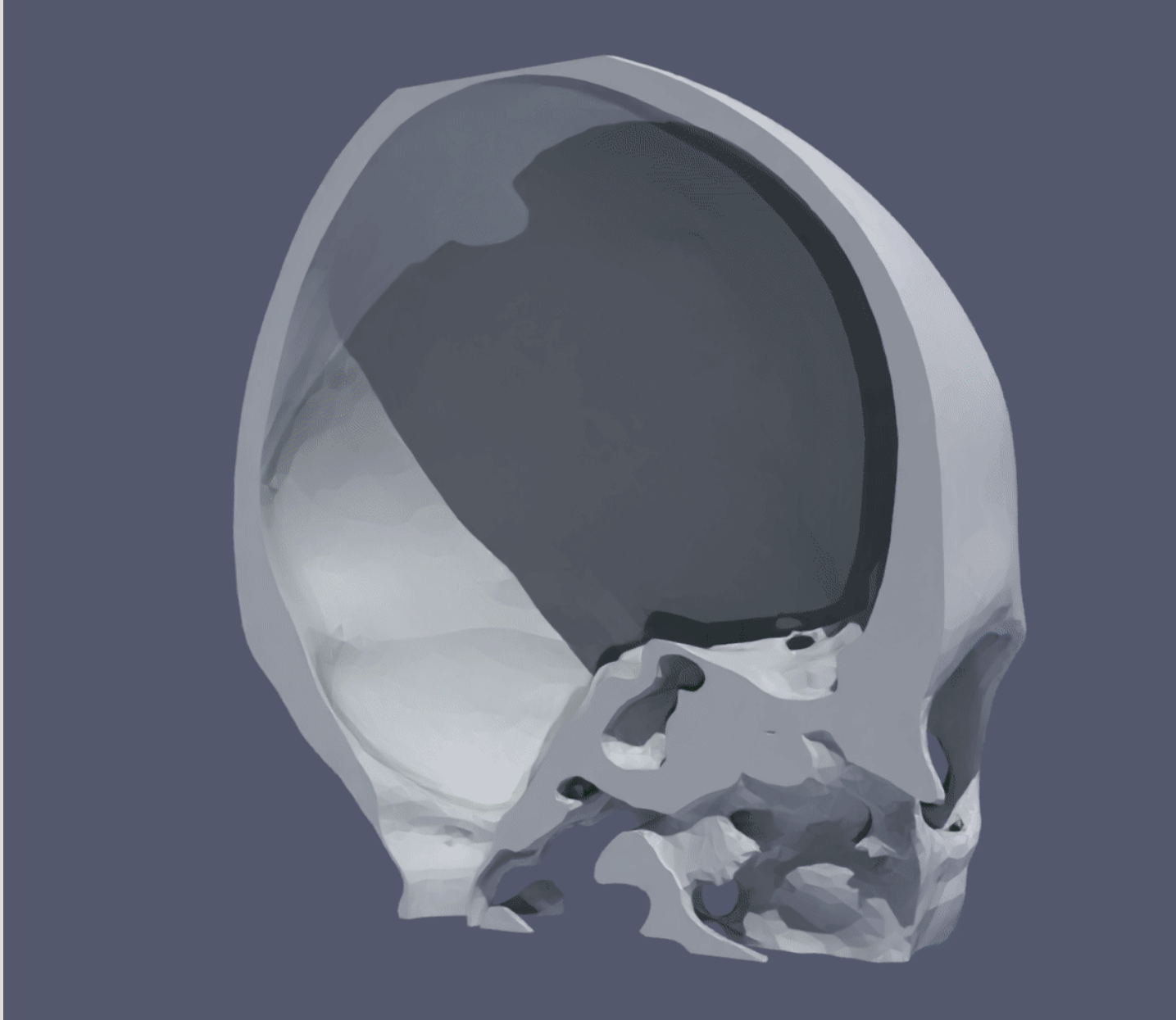
**Weak scaling of Poisson problem on a cube mesh
from 8 to 1024 GCDs on LUMI-G**

Efficiency drops, but we can do
problems on over 1000 GCDs

10M DoFs/GCD



Work by our MPhil student (Sam Russo) using tetrahedral mesh



SUMMARY

- GPU work made possible with EuroHPC development access
- Working within an existing Open Source framework
- Efficient FEM is possible at $p=6$ with p-multigrid and matrix-free methods
- Scaling up depends on efficient neighbor update over network
- More general meshes (tetrahedra, or mixed cell-types) are harder to make efficient
- Future work will address mesh types, and more equations (elasticity, EM, wave, fluid)
- Code example available at <https://github.com/Wells-Group/pmg-dolfinx-lumi>



UNIVERSITY OF
CAMBRIDGE

| epcc |





EuroHPC – User days

Sonic Fan Large Eddy Simulation – EHPC-EXT-2024E01-056

Beurville Allan,

Boudet Jérôme, Clair Vincent, Giauque Alexis, Duchaine Florent

30th September 2025

Context

- growth of air traffic leads to more **pollution** and **noise emission** caused by airplanes,
- modern design of **Ultra-High By-pass Ratio** fan engines has reduced jet and **tonal noise** while increasing efficiency,
- however flow in such configuration is **transonic**, leading to **other noise sources** and **affects noise propagation**,
- **broadband noise** is also significant issue.

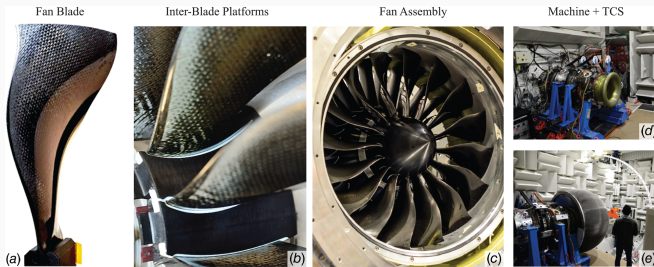


Modern industrial low-speed composite fans and open-test-case
ECL5/CATANA (Pagès *et al.*, 2022)

The ECL5 configuration

European **open** project CATANA:

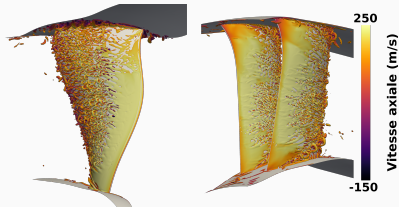
- development of **next generation UHBR** turbofan engines,
- at the École Centrale de Lyon,
- a composite fan stage as interdisciplinary benchmark for **aeroelastic** and **aeroacoustic** interactions,
- **experimental**, and **numerical** dataset.



Rotor blades, rotor assembly and facility (Schneider *et al.*, 2024).

This numerical project – Sonic Fan Large Eddy Simulation

- last year of PhD thesis dedicated to **nominal speed** (100Nn),
- **transonic** flow, **shocks** and acoustic sources,
- simulating the 360° ECL5 **heterogeneous** fan/OGV stage



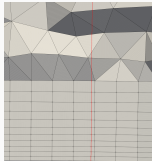
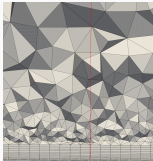
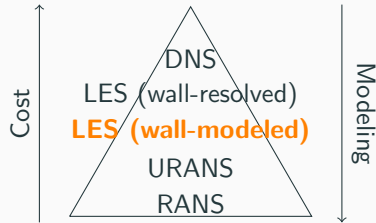
Iso-contour of the Q-criterion colored by axial velocity (sector configuration, IRENE TGCC)

Scientific purpose of a full-stage configuration :

- take into account **blade-to-blade variations** (e.g. stagger angle) : **heterogeneous configuration**,
- **remove** the **periodic boundary condition** allowing the acoustic to **develop in the azimuthal direction**.

Numerical approach

- approach: **large eddy simulation** (LES),
- CFD solver : **AVBP**.  CERFACS, **high performance parallel** LES solver for **compressible flows**.



Dimensionless velocity:

$$u^+ = \begin{cases} \frac{1}{\kappa} \ln y^+ + A & y^+ > 11.45 \\ y^+ & y^+ < 11.45 \end{cases}$$

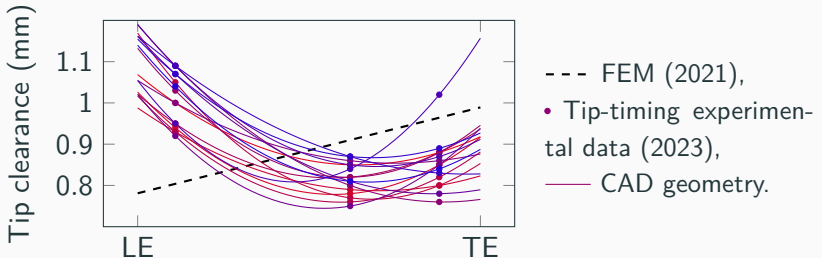


Wall resolved
resolution.

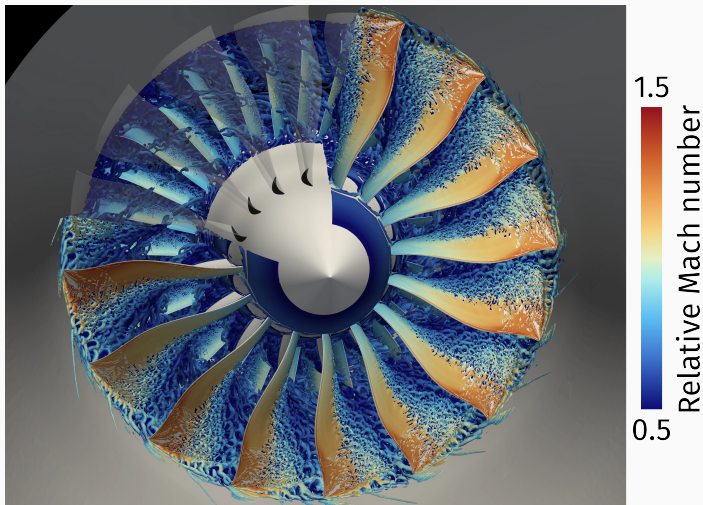
Wall modeled
resolution.

Numerical parameters

- Wall modeled with $y^+ \sim 30$,
 $x^+ \sim 150$,
- Hybrid mesh (prisms et tetrahedrons),
- 1.65 billion cells,
- partitionning: $110 \times$ AMD EPYC 7763,
- mesh cut-off frequency one chord away from the blades: 15 kHz (direct aeroacoustic propagation),
- stagger angle and tip-clearance adjusted for each blade using experimental data: heterogeneous configuration.

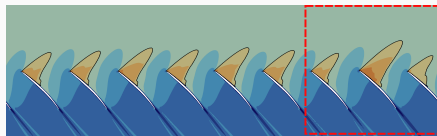


Results overview – Instantaneous flow



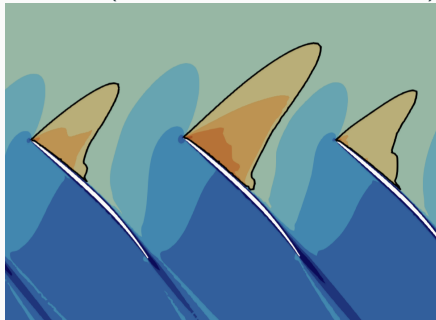
Isocontour of the Q-criterion colored by relative Mach number (360° configuration).

Results overview – Average flow

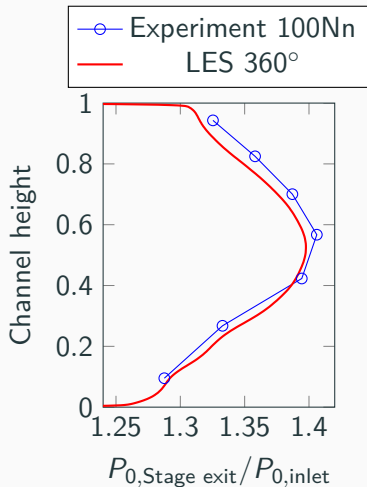


0.4 1.6

Time-averaged relative Mach number (at 80% of the span, rotor)

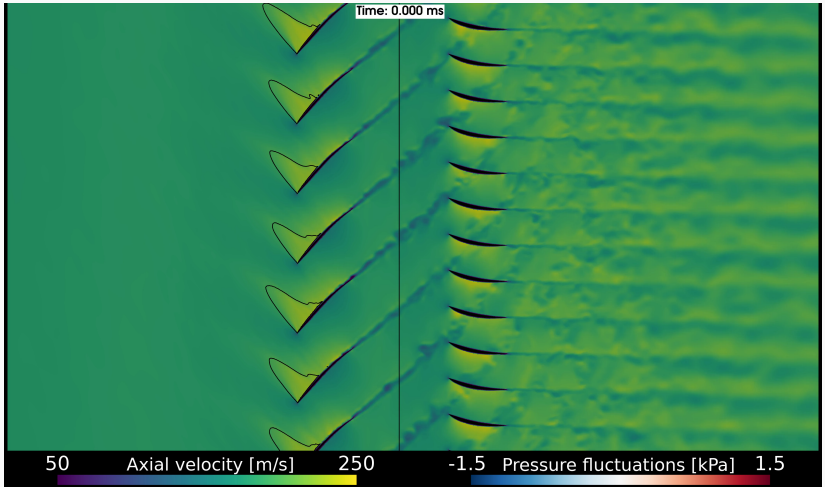


Zoom around blades 6, 7 and 8.



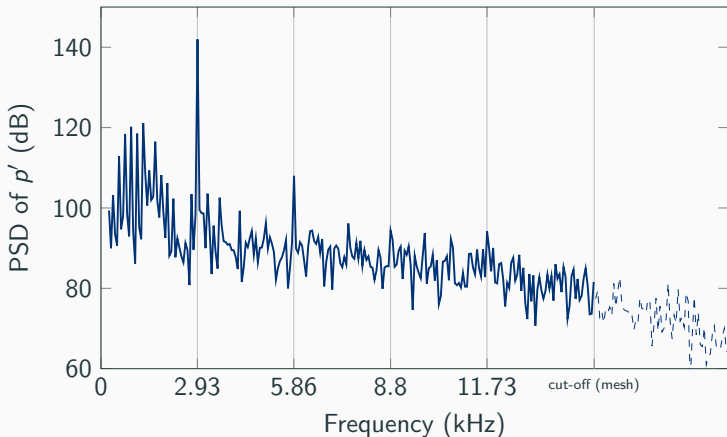
Total pressure ratio at the stage exit.

Results overview – Acoustics



Axial velocity, then instantaneous pressure fluctuation in the stage
(80% of the span).

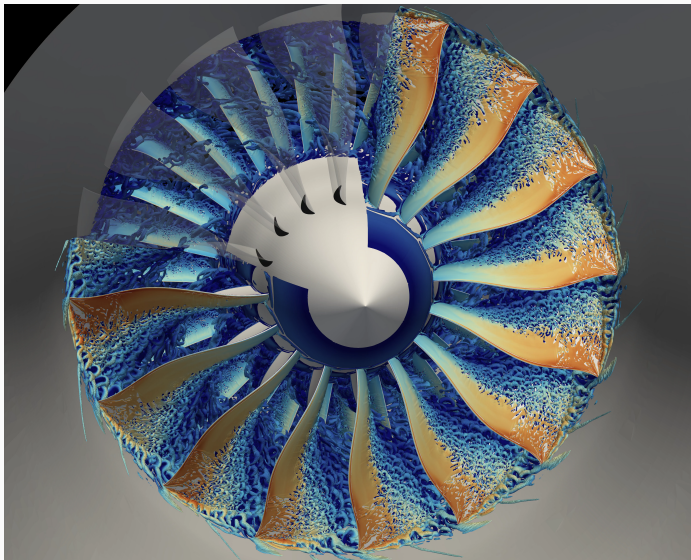
Results overview – Acoustics



Power spectral density of pressure fluctuations one chord upstream of the rotor blades.

- **comparison with experiment**,
- investigate further the **heterogeneity** of the configuration,
- compare results with **the periodic angular sector**,
- perform further **modal acoustic analysis**,
- **share** the produced dataset.

Acknowledgment



Isocontour of the Q-criterion colored by relative Mach number.

Computational resources

Computational cost:

- numerical time step: 1.6×10^{-8} s,
- physical time of one rotation: 5.4×10^{-3} s,
- iterations per rotation: 330,000,
- convergence time: 4 rotations,
- total time simulated on LUMI-C: 14 rotations,
- total CPU usage: **40,000,000** CPUh.

Storage: more than 100 TB of data generated.

Enabling Ginkgo as Numerics Backend in nekRS Employing A Loosely-Coupled Configuration File Concept

Yu-Hsiang M. Tsai¹, Mathis Bode², Hartwig Anzt^{1,3}

mail: yu-hsiang.tsai@tum.de

¹Technical University of Munich

²Forschungszentrum Jülich

³Innovative Computing Laboratory, the University of Tennessee

Outline

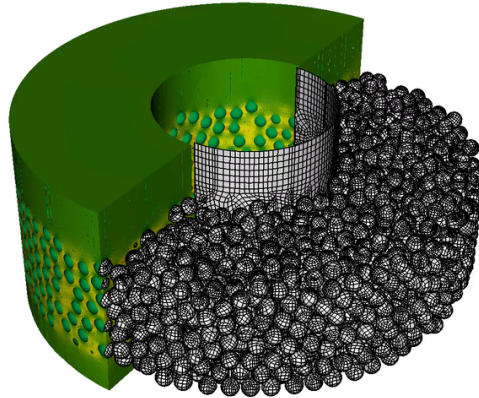
- Background
- Ginkgo integration and approach
- Ginkgo performance in nekRS
- Conclusion

Background

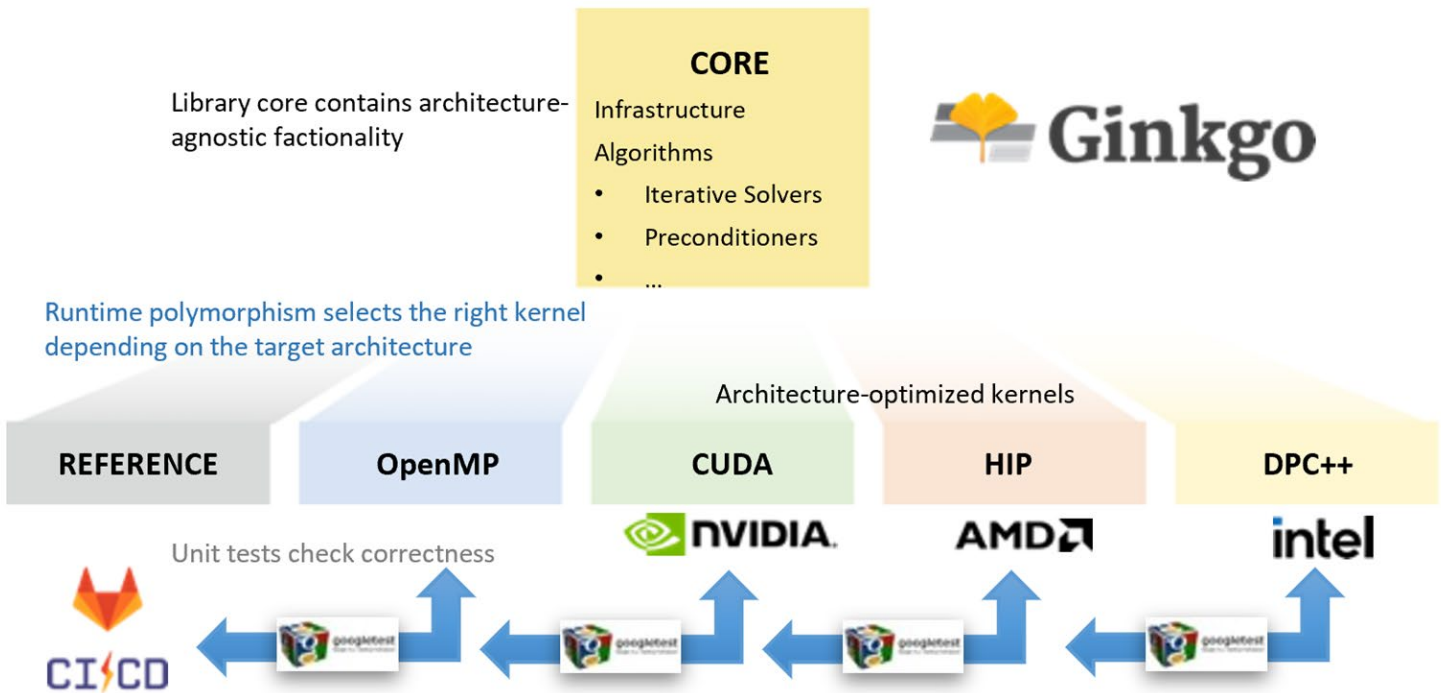
nekRS

nekRS is a fast and scalable computational fluid dynamics (CFD) solver targeting HPC applications.

nekRS is relying on the OCCA portability layer to enable hardware portability.



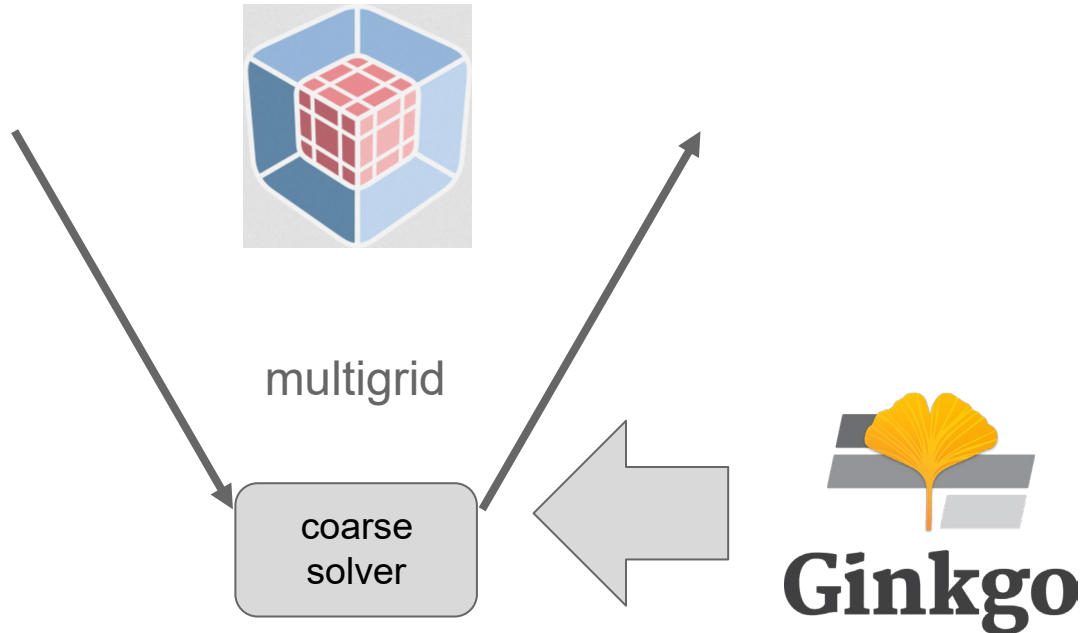
Ginkgo - portable and high performance



Integration

Ginkgo - another option in nekRS

nekRS wants to have more solver options to solve the coarse problem.



Ginkgo - many solver options

		OMP	CUDA	HIP	SYCL
Basic	SpMV	✓	✓	✓	✓
	SpMM	✓	✓	✓	✓
	SpGeMM	✓	✓	✓	✓
	BiCG	✓	✓	✓	✓
Krylov solvers	BiCGSTAB	✓	✓	✓	✓
	CG	✓	✓	✓	✓
	CGS	✓	✓	✓	✓
	GCR	✓	✓	✓	✓
	GMRES	✓	✓	✓	✓
	FCG	✓	✓	✓	✓
	FGMRES	✓	✓	✓	✓
	IR	✓	✓	✓	✓
	IDR	✓	✓	✓	✓
	Chebyshev	✓	✓	✓	✓
	MinRES	✓	✓	✓	✓
	Block-Jacobi	✓	✓	✓	✓
Preconditioners	Gauss-Seidel/SSOR	✓	✓	✓	✓
	ILU/IC	✓	✓	✓	✓
	Parallel ILU/IC	✓	✓	✓	✓
	Parallel ILUT/ICT	✓	✓	✓	✓
	ISAI	✓	✓	✓	✓

		OMP	CUDA	HIP	SYCL
Batched	Batched BiCGSTAB	✓	✓	✓	✓
	Batched CG	✓	✓	✓	✓
	Batched GMRES	✓	✓	✓	✓
	Batched ILU	✓	✓	✓	✓
	Batched ISAI	✓	✓	✓	✓
	Batched Block-Jacobi	✓	✓	✓	✓
AMG	AMG preconditioner	✓	✓	✓	✓
	AMG solver	✓	✓	✓	✓
	Parallel Graph Match	✓	✓	✓	✓
Sparse direct	Symbolic Cholesky	✓	✓	✓	✓
	Numeric Cholesky	✓	✓	✓	✓
	Symbolic LU	✓	✓	✓	✓
	Numeric LU	✓	✓	✓	✓
	Sparse TRSV	✓	✓	✓	✓
Utilities	On-Device Matrix Assembly	✓	✓	✓	✓
	MC64/RCM reordering	✓	✓	✓	✓
	Wrapping user data	✓	✓	✓	✓
	Logging	✓	✓	✓	✓
	PAPI counters (SDE)	✓	✓	✓	✓
	Config file support	✓	✓	✓	✓

✓ MPI Support

✓ Single-GPU Support

Solver selection

There is no perfect solver to solve all kinds of problem correctly and fast.

We have some rules to follow, but applications still need to try several solvers with different settings to find a suitable solver.



Simple approach - compile every time

```
gko::solver::Cg<>::build() ...
```



change and recompile

```
gko::solver::Gmres<>::build() ...
```



change and recompile

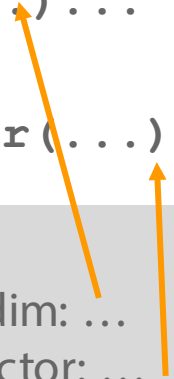
```
gko::solver::Ir<>::build() ...
```

Option approach - massive option

list all kinds of solvers with corresponding options

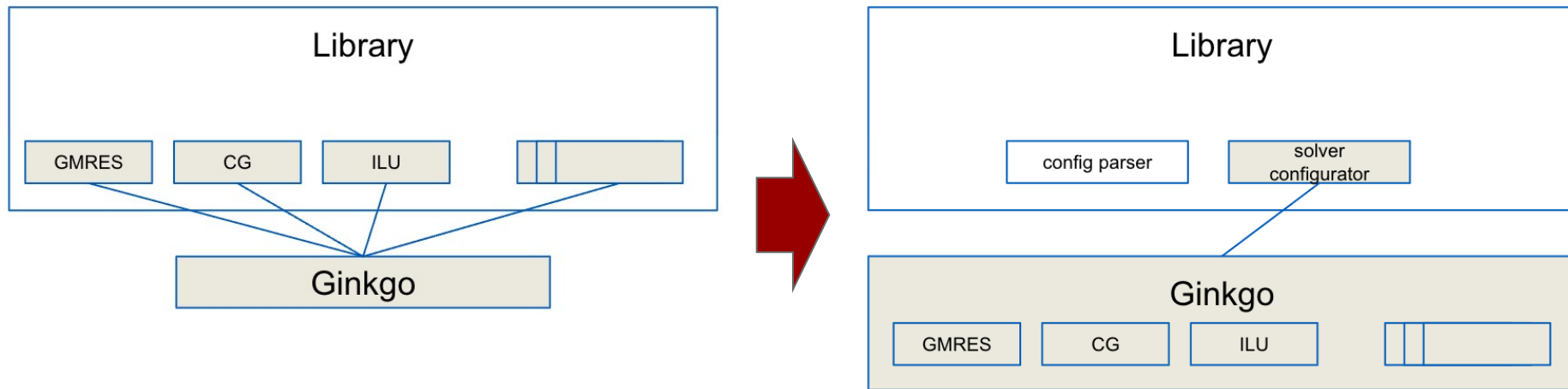
```
if (solver == "GinkgoCg") {  
    gko::solver::Cg<>::build() ...  
} else if (solver == "GinkgoGmres") {  
    gko::solver::Gmres<>::build().with_krylov_dim(...) ...  
} else if (solver == "GinkgoIr") {  
    gko::solver::Ir<>::build().with_relaxation_factor(...) ...  
} else ...
```

preconditioner? multigrid?



solver:..
(gmres_)krylov_dim: ...
(ir_)relaxation_factor: ...
...

New approach - file based config



Configuration example

CG

```
{
  "type": "solver::Cg",
  "criteria": [
    {"type": "Iteration",
     "max_iters": 4},
    {"type": "ImplicitResidualNorm",
     "reduction_factor": 1e-4}
  ]
}
```

hierarchy design

AMG

```
{
  "type": "solver::Multigrid",
  "multigrid_level": [
    {"type": "multigrid::Pgm",
     "deterministic": true}
  ],
  "criteria": [
    {"type": "Iteration",
     "max_iters": 1}
  ],
  "coarse_solver": {...}
}
```

New approach - benefit

- reusable
- hierarchy structure
- clear parameter scope
- easy for experiments
- maintained by Ginkgo directly
- no limitation on file format via internal data structure

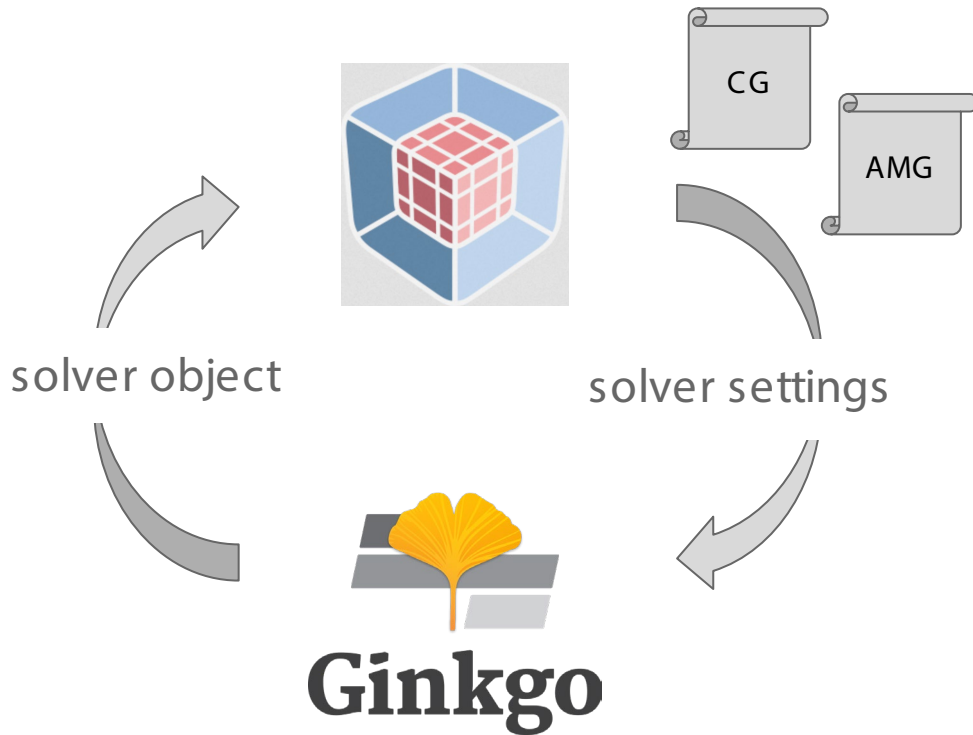


Ginkgo - easy configuration

We can configure a solver via a file

```
[PRESSURE]  
preconditioner = multigrid  
coarseSolver = ginkgo
```

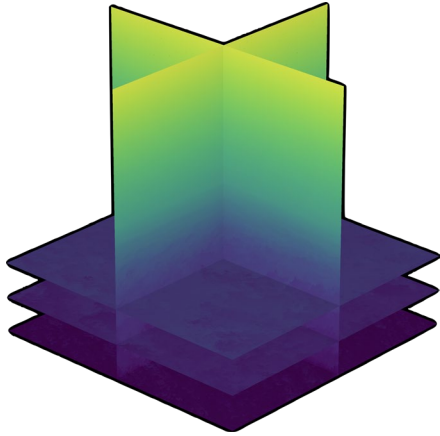
```
[GINKGO]  
configFile = config.json
```



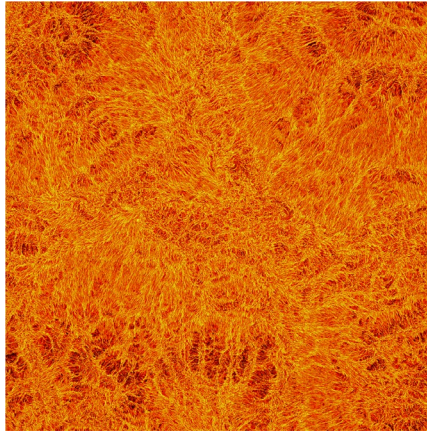
Performance

Experiments

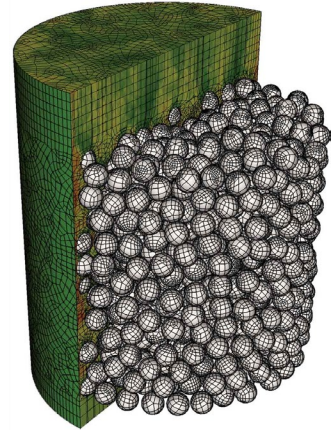
We use the GABLS (GEWEX Atmospheric Boundary Layer Study), RBC (Rayleigh Bénard convection), PB (Pebble Bed) benchmark problems.



GABLS



RBC



PB

PB image from Merzari, Elia, et al. "Cardinal: A low-length-scale multiphysics simulator for pebble-bed reactors."

Experiments

We consider several solver configurations:

- smoother from nekRS
- CG from Ginkgo
- AMG with CG as coarse-grid solver from Ginkgo

We mainly run the experiments on Jureca, which contains 4 Nvidia A100 GPUs per node.



Problem detail

GABLS

RBC

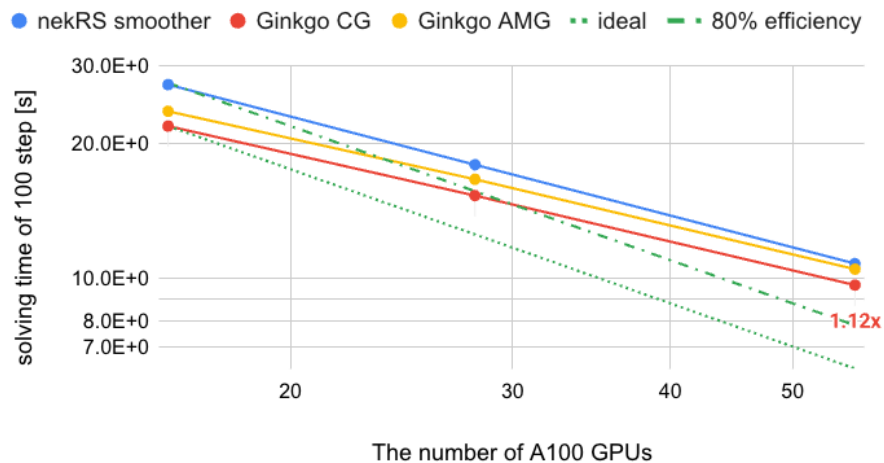
PB

	polynomial order	the total number of elements	global problem size from nekRS to Ginkgo
GABLS_64	8	262,144	266,240
GABLS_128	8	2,097,152	2,113,536
RBC_40	7	2,160,000	2,182,500
RBC_80	9	2,160,000	2,182,500
ann3344	7	1,080,003	1,238,974
cyl11k	8	3,575,076	4,096,583

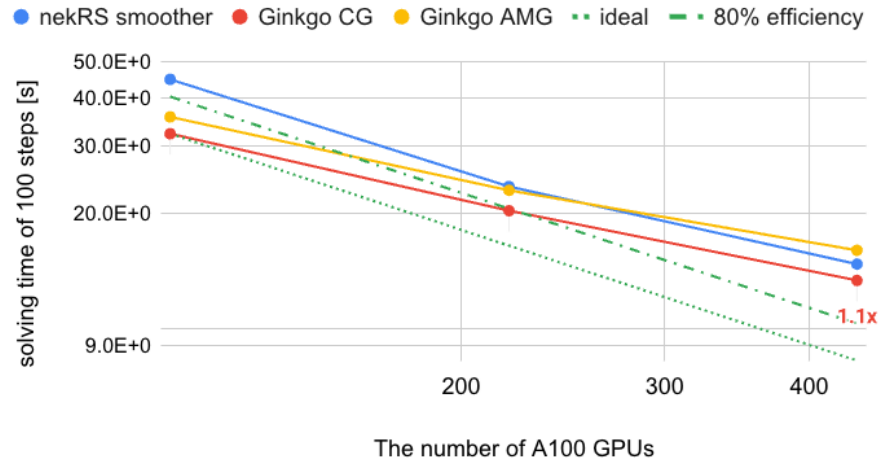
Performance - GABLS

Ginkgo CG performs the best and gives around 1.12x speedup

GABLS_64 on JURECA: Strong Scalability

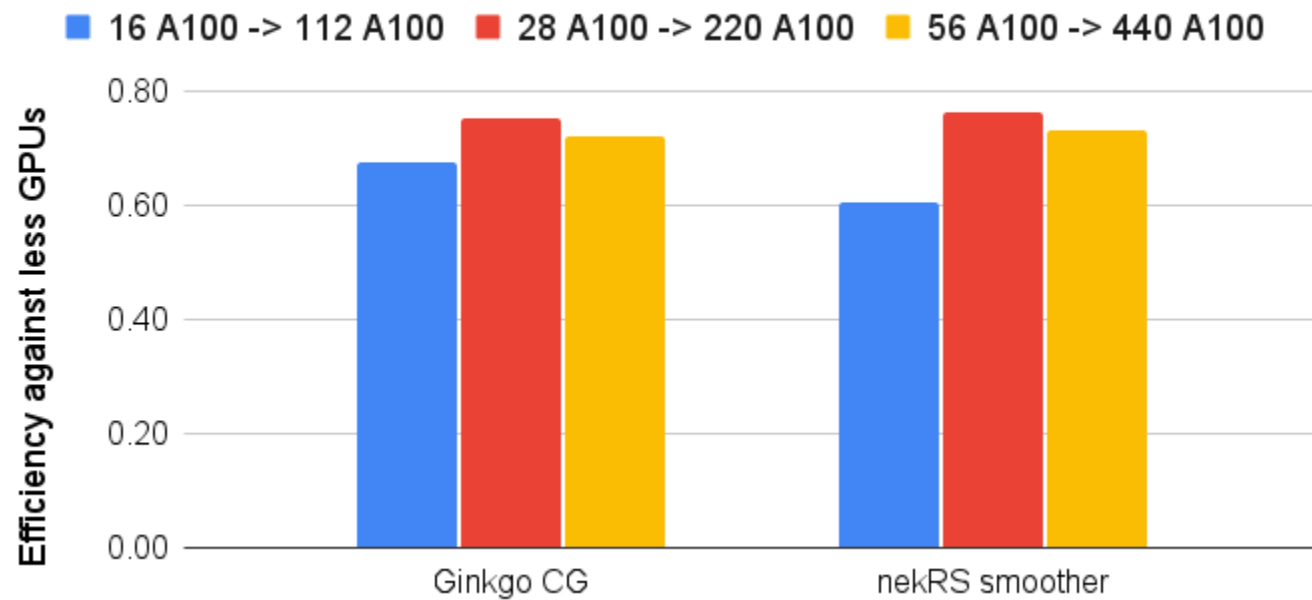


GABLS_128 on JURECA: Strong Scalability



Performance - G A B L S

GABLS on JURECA: Weak Scaling Efficiency

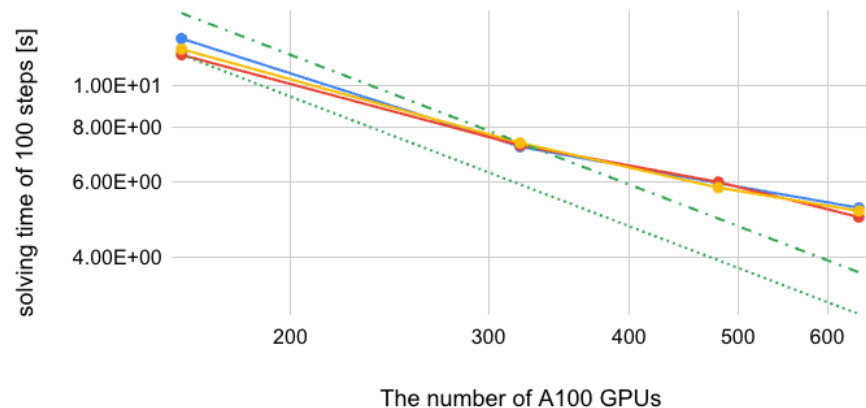


Performance - R B C

Ginkgo CG performs the best and gives around 1.15x speedup

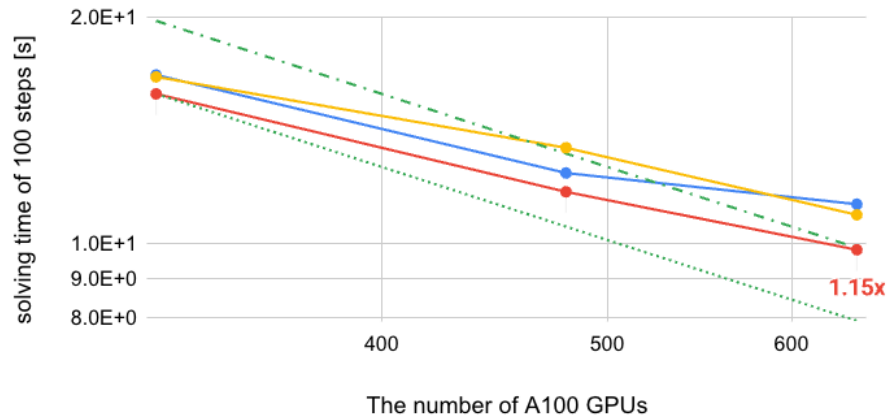
RBC_40 on JURECA: Strong Scalability

● nekRS smoother ● Ginkgo CG ● Ginkgo AMG --- ideal --- 80% efficiency



RBC_80 on JURECA: Strong Scalability

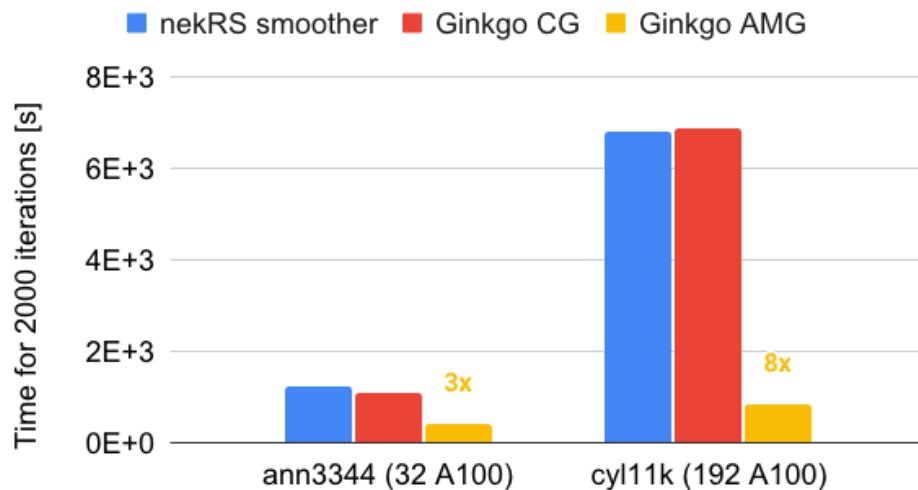
● nekRS smoother ● Ginkgo CG ● Ginkgo AMG --- ideal --- 80% efficiency



Performance - P B

Ginkgo AMG outperforms others and gains 3x and 8x speedup.

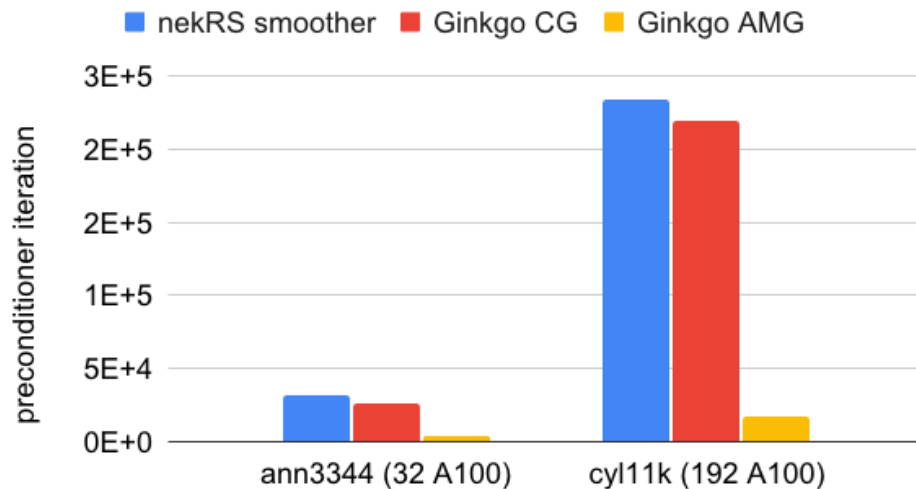
Pebble bed cases on JURECA: Performance



Convergence - P B

The performance benefit is from the better convergence on challenging problems.

Pebble bed cases on JURECA: Convergence



Acknowledgement

This research is supported by the Inno4Scale project under Inno4scale202301-099. Inno4Scale has received funding from the European HighPerformance Computing Joint Undertaking (JU) under grant agreement No 101118139. The JU receives support from the European Union's Horizon Europe Programme. The authors acknowledge the Gauss Centre for Supercomputing e.V. (www.gausscentre.eu) for funding this project by providing computing time on the GCS Supercomputers at Jülich Supercomputing Centre (JSC).

Conclusion

- Integrate Ginkgo as an another option to nekRS
- Provide a reusable approach to configure Ginkgo solver in other libraries
- Perform nekRS with Ginkgo on large scale machine JURECA
- Ginkgo CG gives up to 1.15x speedup on RBC and GABLS problems
- Ginkgo AMG gives up to 8x speedup on challenging PB problems

accLB: A High-Performance Lattice Boltzmann Code for Multiphase Turbulence on Multi-Gpu Architectures

A. Montessori(1), Marco Lauricella(2), Aritra Mukherjee(3), Luca Brandt(3,4), Sauro Succi,(5), Daulet Izbassarov (6)

1) Department of Civil, Computer Science and Aeronautical Technologies Engineering, Roma Tre University

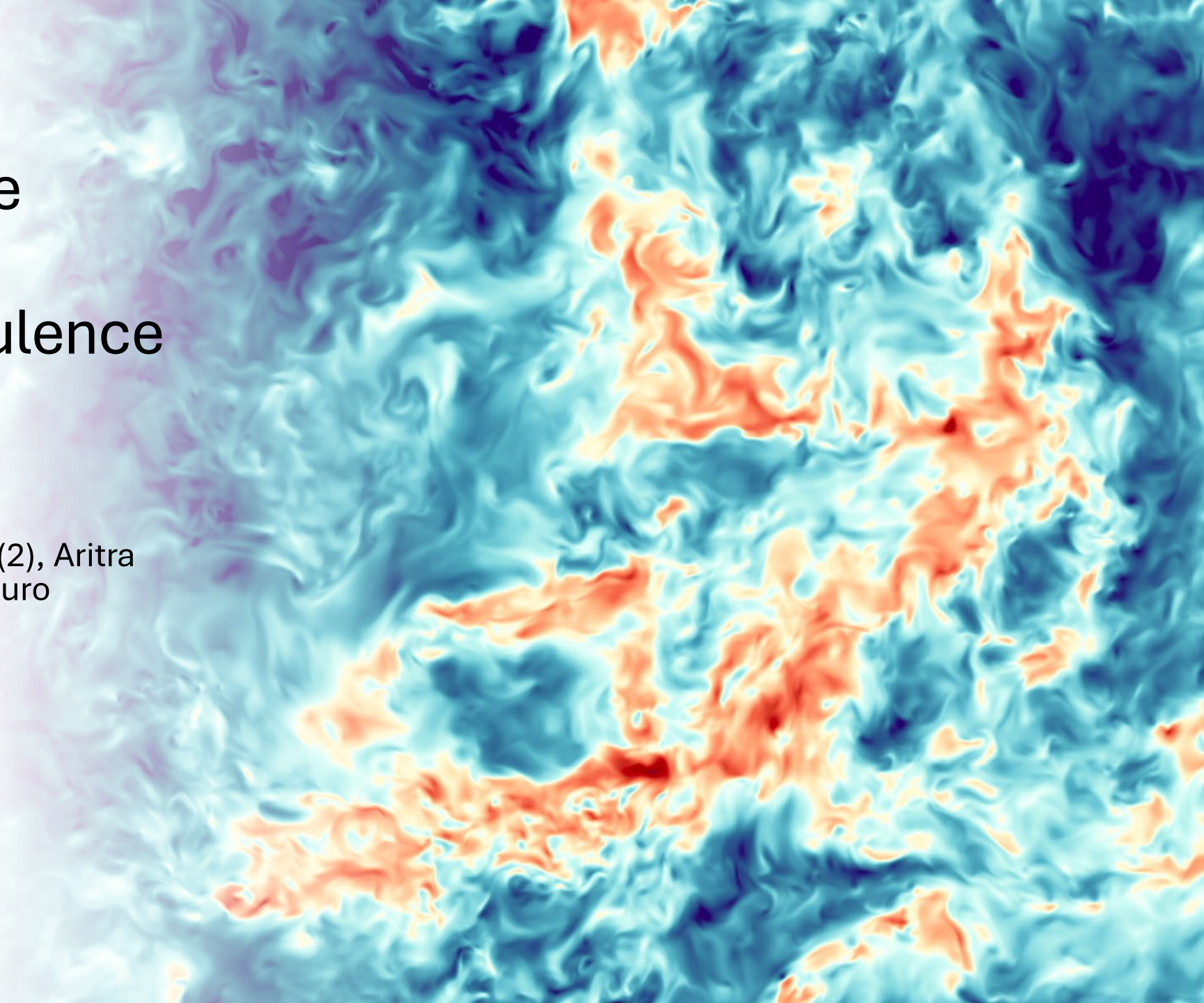
2) Istituto per le Applicazioni del Calcolo

3) Department of Energy and Process Engineering, Norwegian University of Science and Technology

4) Dipartimento di Ingegneria dell'Ambiente, del Territorio e delle Infrastrutture, Politecnico di Torino

5) Istituto Italiano di Tecnologia

6) Finnish Meteorological Institute



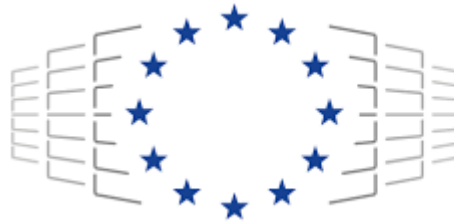
Acknowledgments



Ministero
dell'Università
e della Ricerca



Grant (MOBIOS) ID: 2022N4ZNH3 -CUP: F53C24001000



EuroHPC
Joint Undertaking



LEONARDO
CINECA

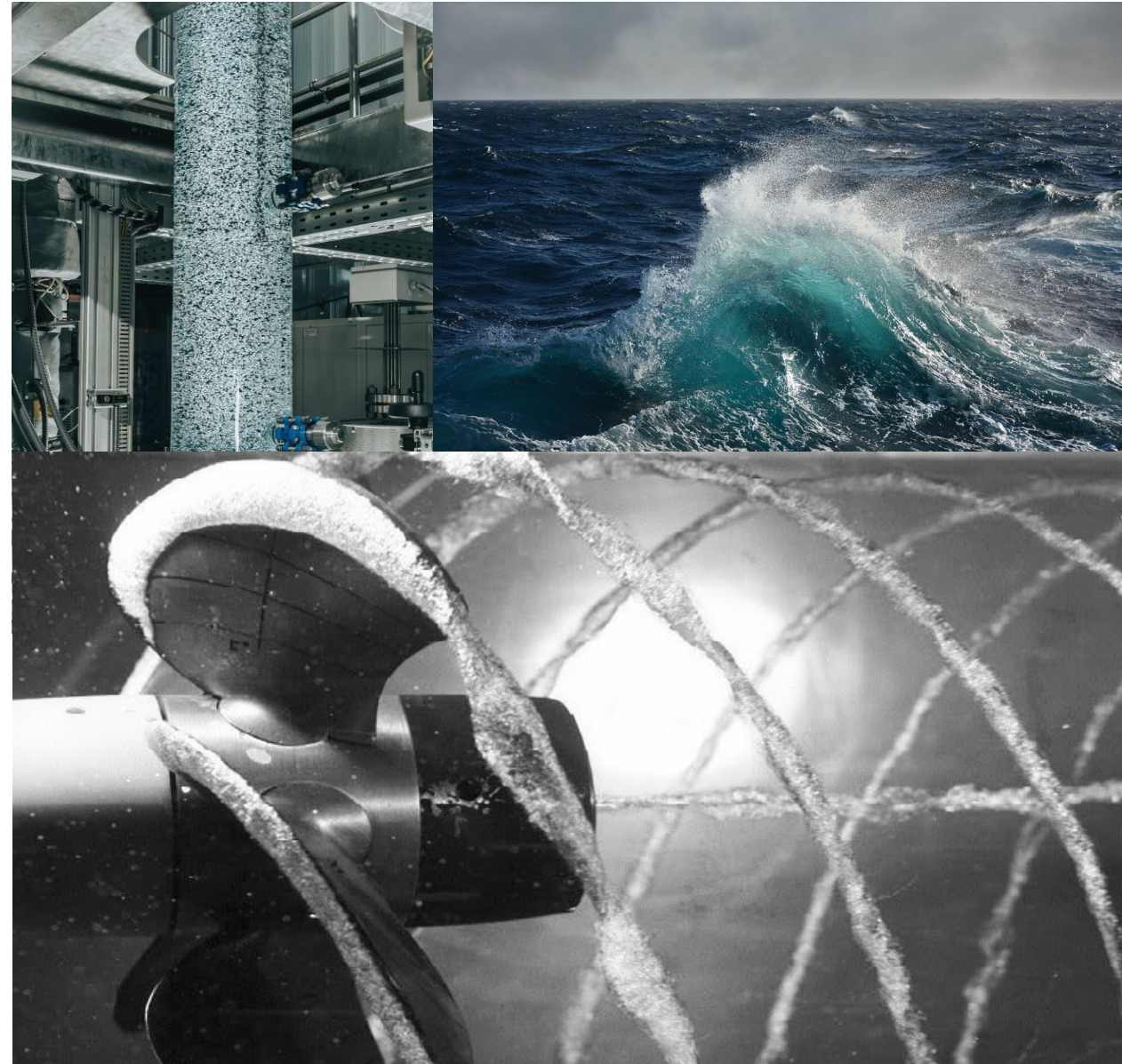
ISCRA B project DODECA (HP10BUBFIL)



Research Council of Finland, Grant number: 354620

Motivation & Context

- Multiphase turbulence is key in many applications: from ocean-atmosphere exchange to industrial mixing processes and cavitation.
- Intrinsic multiscale and multiphysics nature modifies energy transfer across scales and alters small-scale dynamics
- HPC numerical simulations and accurate multiphase models play a fundamental role in uncovering the underlying physics



accLB: a high-performance computing code for multiphase flows simulations

- **accLB**: a high-performance Fortran-based lattice Boltzmann (LB) solver tailored to multiphase turbulent flows on multi-GPU architectures.
- accLB couples a **conservative** phase-field formulation of the **Allen–Cahn equation** with a **thread-safe, regularized LB method** to capture complex interface dynamics.
- It employs **MPI** for distributed memory parallelism and **OpenACC** for GPU acceleration, achieving excellent portability and scalability on leading pre-exascale systems such as *Leonardo(CINECA, Italy)* and *LUMI (Finland)*.

Physics of Fluids

ARTICLE

pubs.aip.org/aip/pof

Thread-safe multiphase lattice Boltzmann model for droplet and bubble dynamics at high density and viscosity contrasts

Cite as: Phys. Fluids **37**, 072111 (2025); doi: [10.1063/5.0271706](https://doi.org/10.1063/5.0271706)

Submitted: 19 March 2025 · Accepted: 18 May 2025 ·

Published Online: 9 July 2025



Marco Lauricella,¹ Adriano Tiribocchi,^{1,2} Sauro Succi,³ Luca Brandt,⁴ Aritra Mukherjee,⁵ Michele La Rocca,⁶ and Andrea Montessori^{6,a)}



Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 00 (2025) 000–000

Procedia
Computer Science
www.elsevier.com/locate/procedia

Proceedings of the Third EuroHPC user day

accLB: A High-Performance Lattice Boltzmann Code for
Multiphase Turbulence on Multi-Gpu Architectures

Marco Lauricella^a, Aritra Mukherjee^b, Luca Brandt^{b,c}, Sauro Succi^{a,d,e}, Daulet
Izbassarov^f, Andrea Montessori^g

Computational Framework: LB

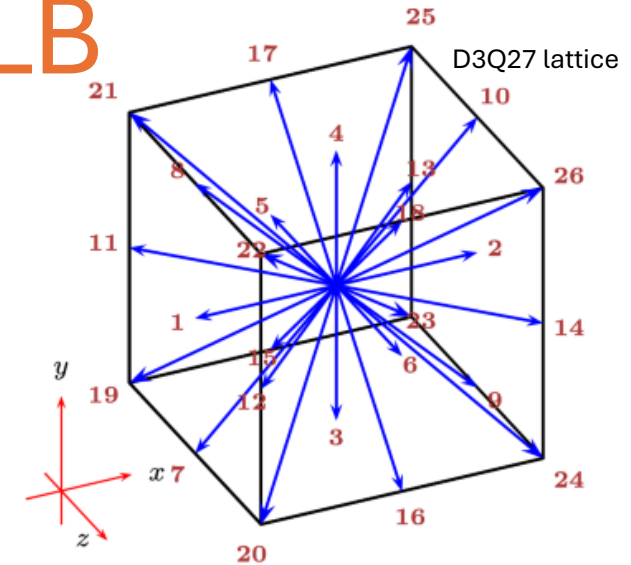
a) The lattice Boltzmann equation

Free streaming along lattice directions

$$f_i(\mathbf{x} + \mathbf{c}_i \Delta t, t + \Delta t) = f_i(\mathbf{x}, t)$$

Collision and forcing

$$f_i(\mathbf{x} + \mathbf{c}_i \Delta t, t + \Delta t) = f_i(\mathbf{x}, t) + \omega(f_i^{\text{eq}} - f_i) + (1 - \frac{\omega}{2})S_i$$



b) Macroscopic fields reconstruction via statistical moments of the distribution

$$p^*(x_\alpha, t) = \sum_i f_i(x_\alpha, t), \quad u(x_\alpha, t) = \sum_i f_i(x_\alpha, t) c_{i\alpha} + \frac{1}{2} \sum_i S_i c_{i\alpha}, \quad a_{neq, \alpha\beta}^2 = \sum_i (c_{i\alpha} c_{i\beta} - c_s^2 \delta_{\alpha\beta}) (f_i - \overline{f_i^{eq}}).$$

Pressure

Linear momentum

Momentum flux tensor

c) Multiscale expansion(Chapman-Enskog) ensure that solving the LB equation in time and space is equivalent of solving a set of PDE for mass and momentum conservation → The hydrodynamic limit of the LBE is NSE

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0$$

$$\frac{\partial(\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u} \mathbf{u}) = -\nabla p + \nabla \cdot (\rho \nu [\nabla \mathbf{u} + (\nabla \mathbf{u})^\top]) + \mathbf{F}^s$$

$$\mathbf{F}^s = \nabla \cdot (\sigma(\mathbf{I} - \mathbf{n} \otimes \mathbf{n}))$$

Computational Framework: Conservative Allen-Cahn

$$\frac{\partial \phi}{\partial t} + \overset{\text{Advection}}{\mathbf{u} \cdot \nabla \phi} = \overset{\text{Diffusion}}{D \nabla^2 \phi} - \overset{\text{Interfacial compression}}{\kappa \nabla \cdot [\phi(1 - \phi) \mathbf{n}]}$$

- The interface dynamics is captured through the evolution of a phase field, ϕ , governed by the conservative Allen-Cahn equation
- The interface tracking equation is dynamically coupled to the Navier-Stokes equations through the advection term.
- To enhance the accuracy of the model in the presence of strong advection, the advection term is discretized using a fifth-order Weighted Essentially Non-Oscillatory (WENO-5) scheme.

Allen, S. M., & Cahn, J. W. (1979).

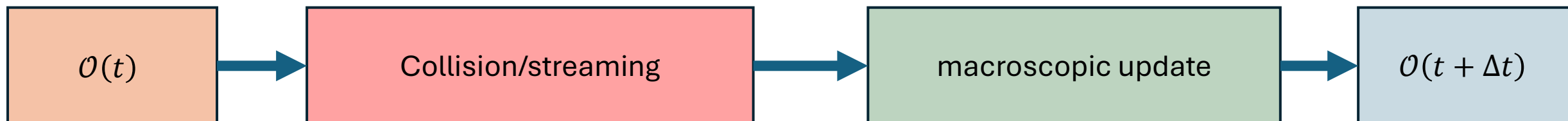
A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening., *Acta Metallurgica*, 27(6), 1085–1095.

Sun, Y., Beckermann, C. (2007).

Sharp interface tracking using the phase-field equation.
Journal of Computational Physics, 220(2), 626–653.

Implementation details

- **Full memory decomposition with MPI:** 3D Cartesian domain decomposition with halo exchange between neighboring subdomains.
- **GPU offloading with OpenACC Fortran:** Each MPI rank mapped to one GPU. OpenACC directives ensure fine-grained parallelism and portability across GPU architectures (both NVIDIA and AMD GPUs).
- **Scalability:** Designed for large-scale heterogeneous HPC clusters
- **Thread-safe:** Collision/streaming use macroscopic fields as input and update distributions; macroscopic update does the reverse.

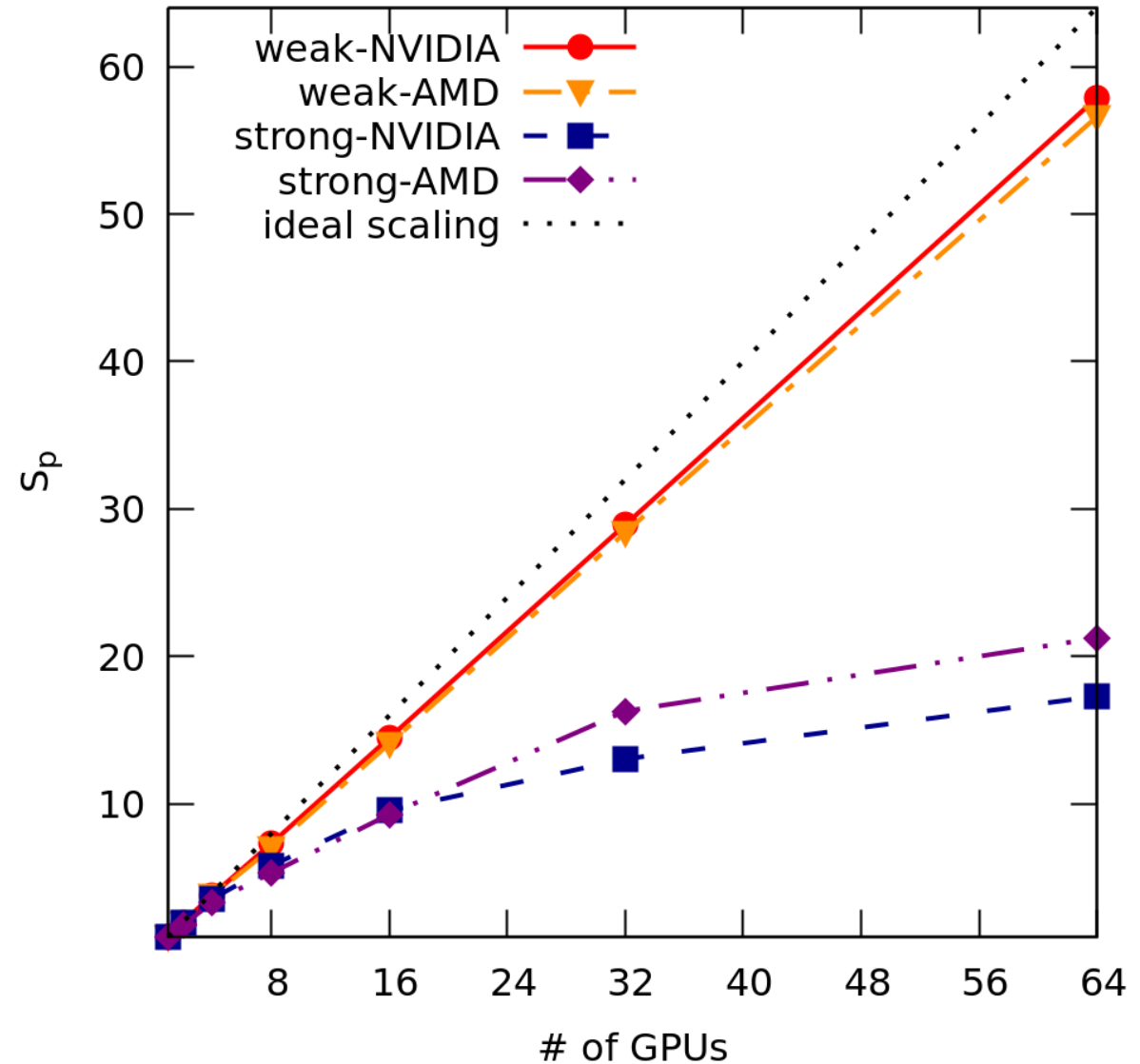


Performances

- **Benchmark case:** 3D droplet dynamics simulation
- **Scaling test:** up to 64 MPI ranks (1 rank $\leftrightarrow$ 1 GPU)
- **Strong scaling:** cubic box with $512 \times 512 \times 512$ lattice points.
- **Weak scaling:** cubic box with $512 \times 512 \times 512 \times (\text{\#GPUs})$ lattice points.

$\text{GLUPS} = \frac{L_x L_y L_z}{10^9 t_s}$	$S_p = \frac{\text{GLUPS}_p}{\text{GLUPS}_s}$
Billions of lattice updates per seconds	speedup

t_s = wall clock time



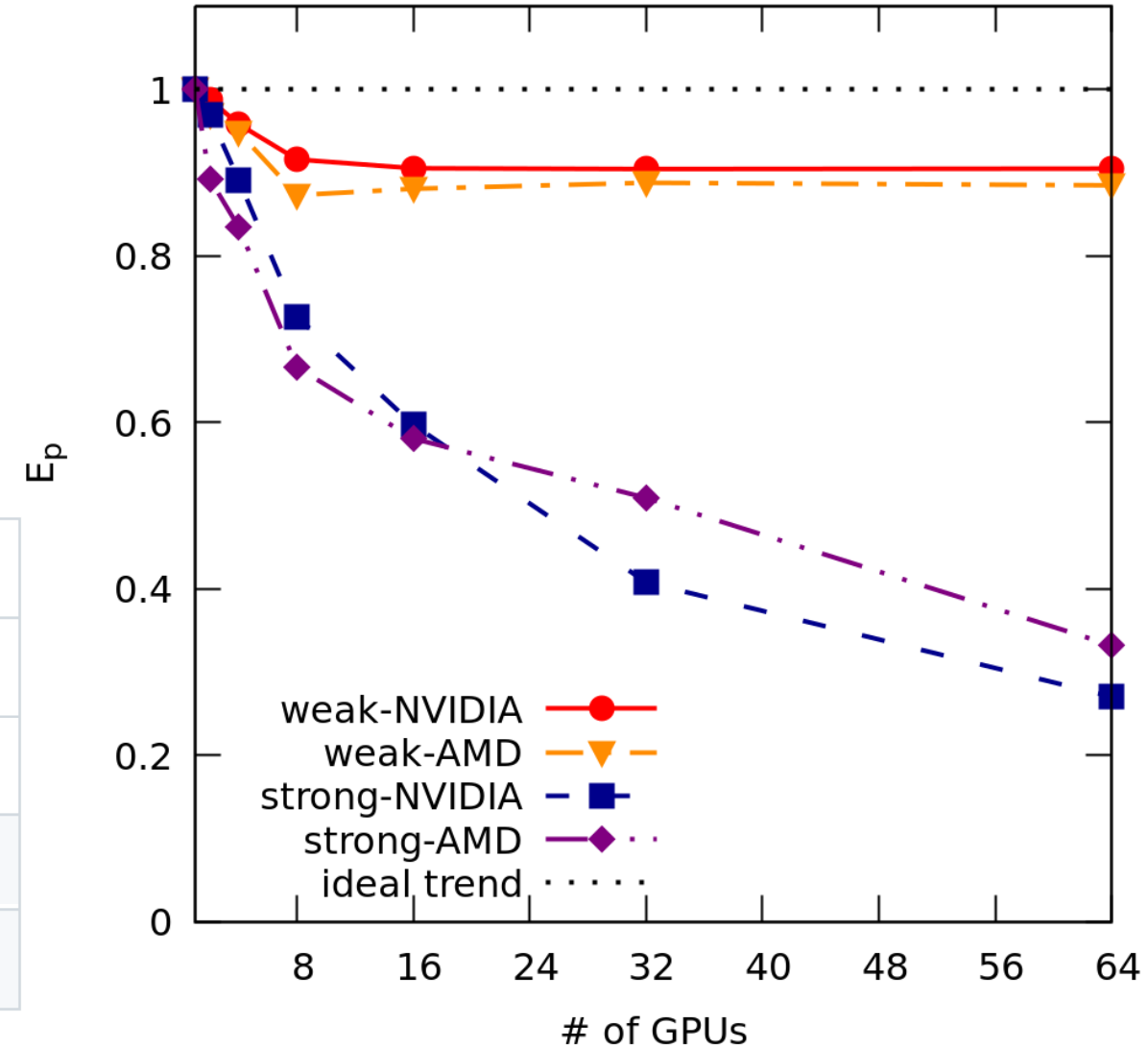
Performances

Parallel efficiency: $E_p = \frac{S_p}{n_p}$

Weak scaling with parallel efficiency close to 0.9.

Weak scaling: on Nvidia A100 and AMD MI250X

GPU	Nprocs	MPI decomp.	GLUPS
● NVIDIA A100 64GB	1	1 × 1 × 1	1.60
● NVIDIA A100 64GB	64	1 × 1 × 64	92.63
● AMD MI250X 128GB	1	1 × 1 × 1	1.16
● AMD MI250X 128GB	64	1 × 1 × 64	65.67



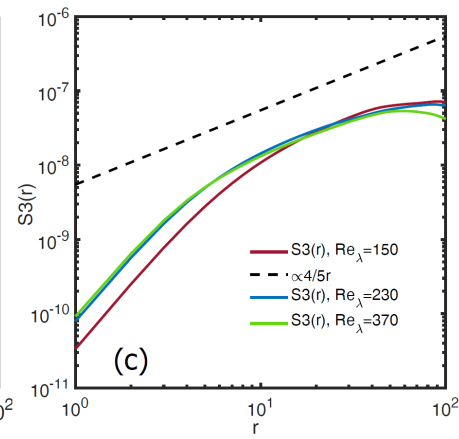
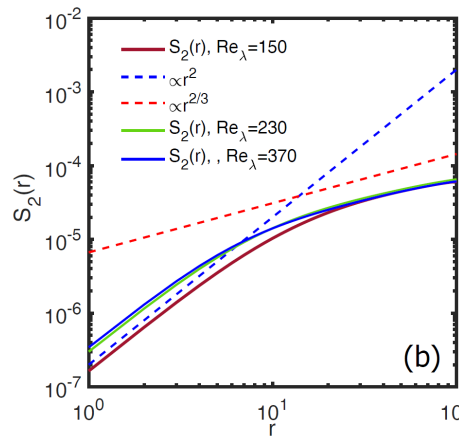
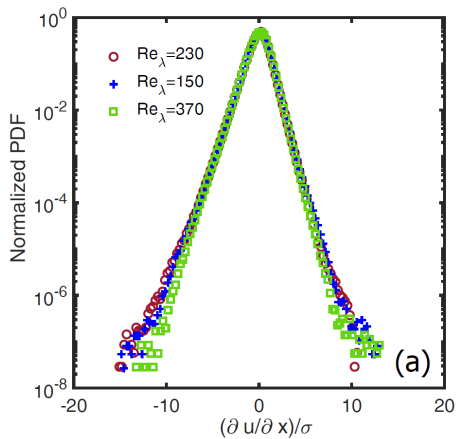
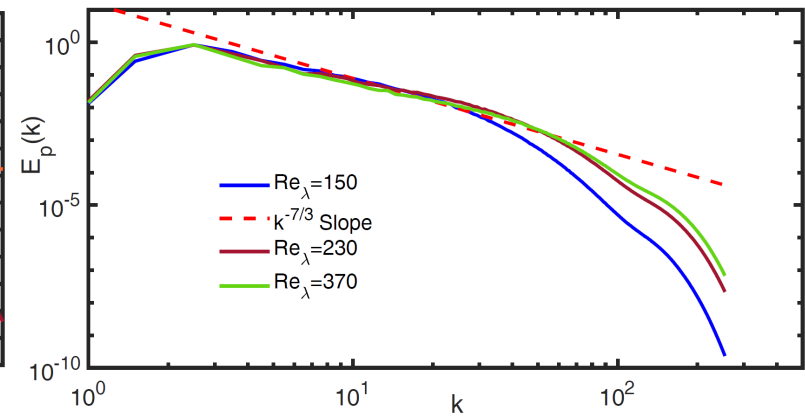
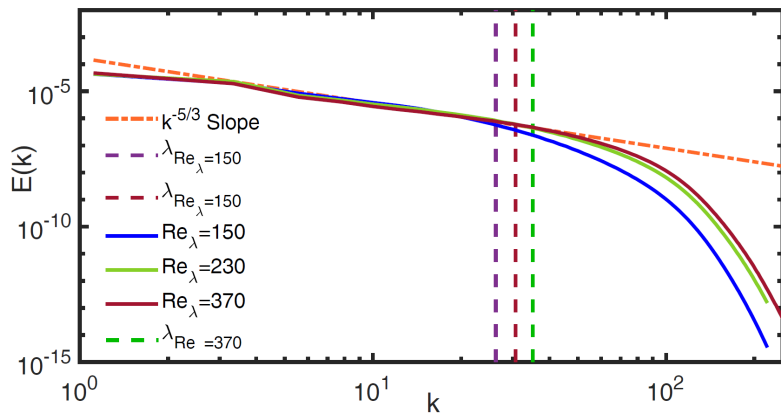
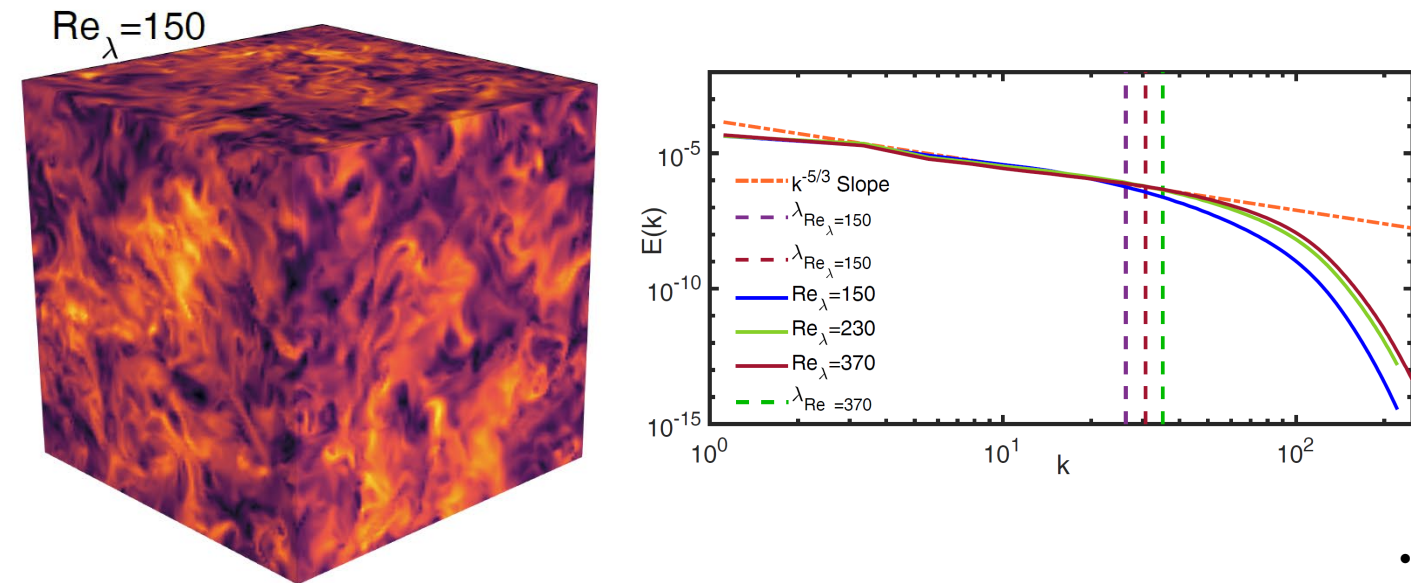
Simulation Setup

- We performed bubble-laden HIT simulations across a range of bubble volume fractions (ϕ_B), viscosity, and density ratios.
- The focus is on the impact of gas-liquid interfaces on higher-order turbulence statistics and their connection to the turbulence modulation.
- Cubic, tri-periodic box (512^3 computational nodes)
- The homogeneous isotropic turbulent state is maintained through large-scale energy injection via the ABC (Arnold–Beltrami–Childress) forcing, applied only on the bulk liquid phase.

$$\mathbf{f}(\mathbf{x}) = A \begin{pmatrix} \sin(k_f z) + \cos(k_f y) \\ \sin(k_f x) + \cos(k_f z) \\ \sin(k_f y) + \cos(k_f x) \end{pmatrix}$$

- Three Taylor-Reynolds number ($Re_\lambda = \frac{u' \lambda}{\nu}$) have been considered $Re_\lambda = 65, 86, 114$ (150,230 and 370 single component case).

Code validation: Single-phase HIT



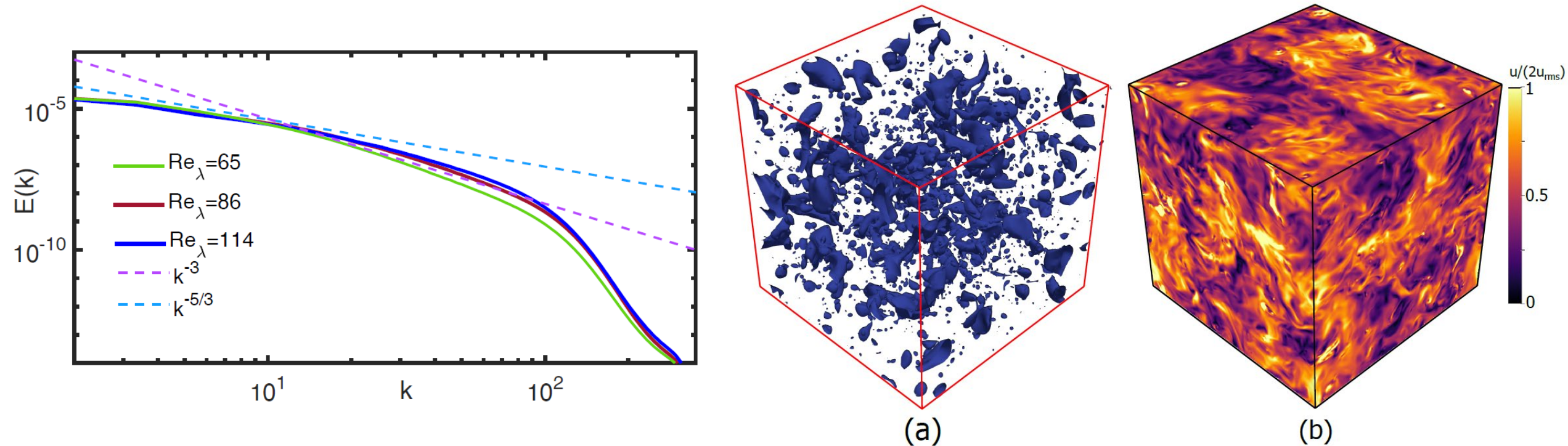
$$S_2(r) = \langle [\delta_\parallel(x, r)]^2 \rangle.$$

$$S_3(r) = \langle [\delta_\parallel(x, r)]^3 \rangle.$$

- Instantaneous velocity field in a homogeneous single-phase turbulent flow at $Re_\lambda = 150$.
- In upper left panel, the energy spectrum for two different Re_λ , namely 150, 230 and 370, is reported.
- In upper right panel, the pressure spectra for the same Re_λ follow the $-7/3$ scaling law within the inertial range.
- Lower Panels: the normalized PDF of the longitudinal velocity gradients and S_2 and S_3 compared with two theoretical scaling laws, namely r^2 in the dissipation range and the Kolmogorov scaling $r^{2/3}$ and $r^{4/5}$, in the inertial range.

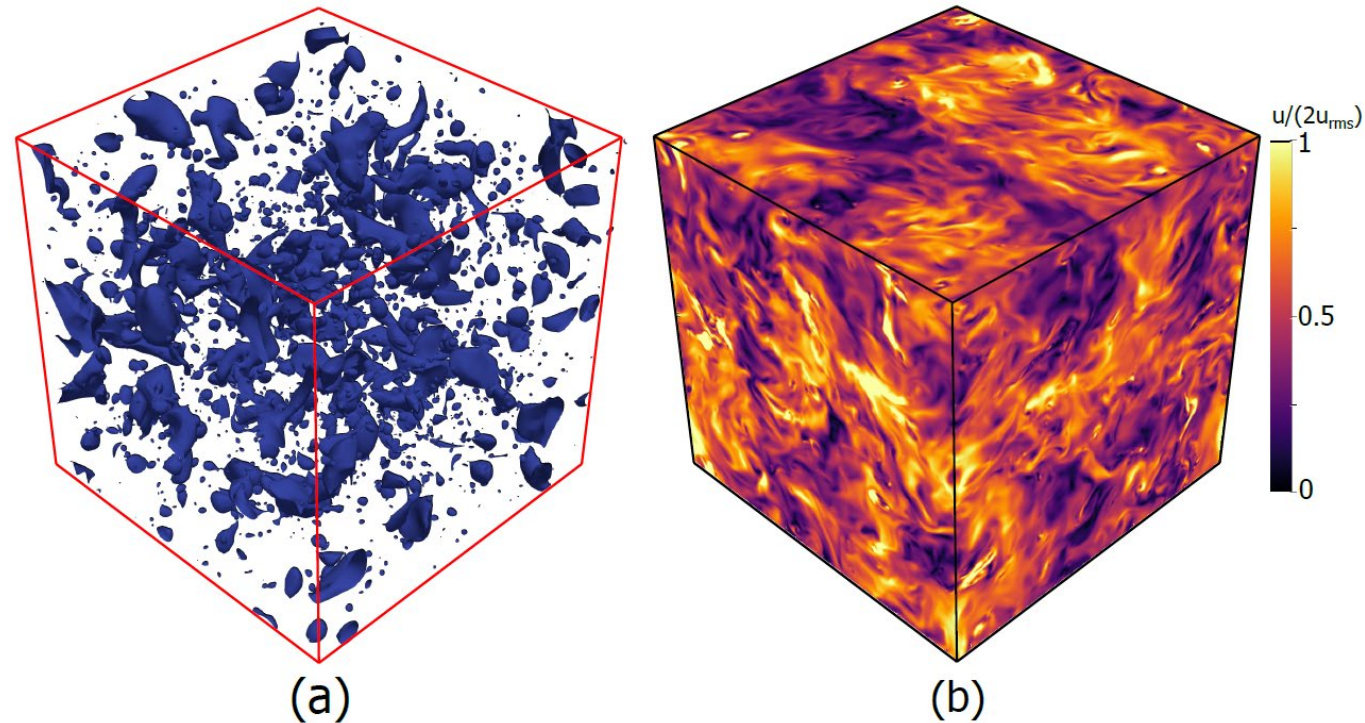
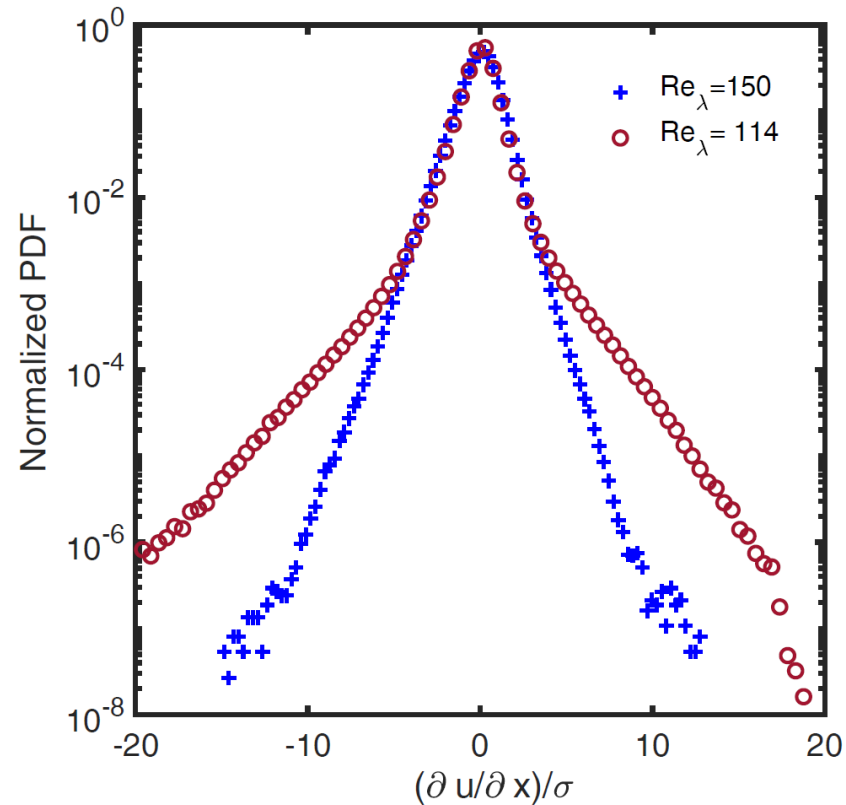
$$\delta_\parallel(x, r) = [u(x + r) - u(x)]r/|r| \quad \text{longitudinal velocity increment at separation } r$$

Bubble-laden HIT: energy spectra



Energy spectra for bubble-laden HIT at different Re_λ . The transition from $-5/3$ to -3 scaling, observed experimentally in [28], is captured in the intermediate wavenumber range. (Right) Instantaneous snapshots of the bubble field panel (a) and velocity field (b) at $Re_\lambda = 114$ and 6% bubble volume fraction. Snapshots are taken at the statistical steady state.

Bubble-laden HIT: Intermittency



Comparison of velocity increment PDFs between single-phase and bubble-laden HIT. The enhanced tails in the latter case indicate increased intermittency induced by bubble dynamics. (Right) Instantaneous snapshots of the bubble field panel (a) and velocity field (b) at $Re_\lambda = 114$ and 6% bubble volume fraction. Snapshots are taken at the statistical steady state.

Conclusions

- In this work we have presented accLB, a high-performance, thread-safe Lattice Boltzmann solver tailored for the multiphase turbulent flows on modern GPU-based architectures.
- The code architecture exploits hybrid parallelism via MPI and OpenACC, offering excellent portability and scalability across heterogeneous hardware platforms.
- Benchmark tests on pre-exascale systems Leonardo and LUMI demonstrate strong and weak scaling performance up to 64 parallel GPUs, with sustained throughput exceeding 150 GLUPS in large-scale runs.
- In the single-phase case, we recover the classical energy and pressure spectra scalings and turbulence invariants across scales.
- For the bubble-laden scenarios, our simulations capture the emergence of a -3 scaling in the energy spectrum, along with amplified small-scale intermittency.
- These findings confirm that accLB is a reliable tool to investigate turbulence modulation in complex multiphase systems.