GmshDDM on LUMI: first runs of a new solver for large scale time-harmonic flow acoustics problems

P. Marchner[1,2,3], X. Antoine[2], H. Beriot[3] and C. Geuzaine[1]

[1]University of Liège, [2]University of Lorraine, [3]Siemens Digital Industries Software

LUMI User Day, November 6 2023

LIÈGE université

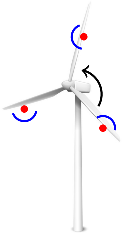
UNIVERSITÉ DE LORRAINE

SIEMENS Ingenuity for life

Industrial context

Predict noise from bodies in motion for the transport/energy industry

Computational (aero)acoustics

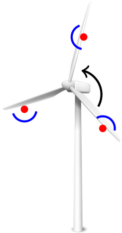


1. Analyze & extract sources
2. Understand sound propagation
3. Find solutions (new material or design)

Industrial context

Predict noise from bodies in motion for the transport/energy industry

Computational (aero)acoustics



1. Analyze & extract sources
2. Understand sound propagation
3. Find solutions (new material or design)

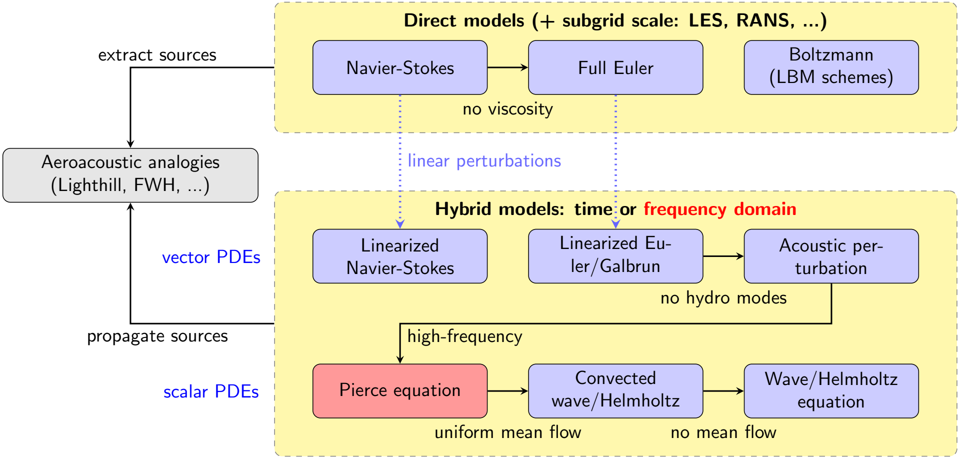
Overall objective

Provide a sound propagation simulation tool

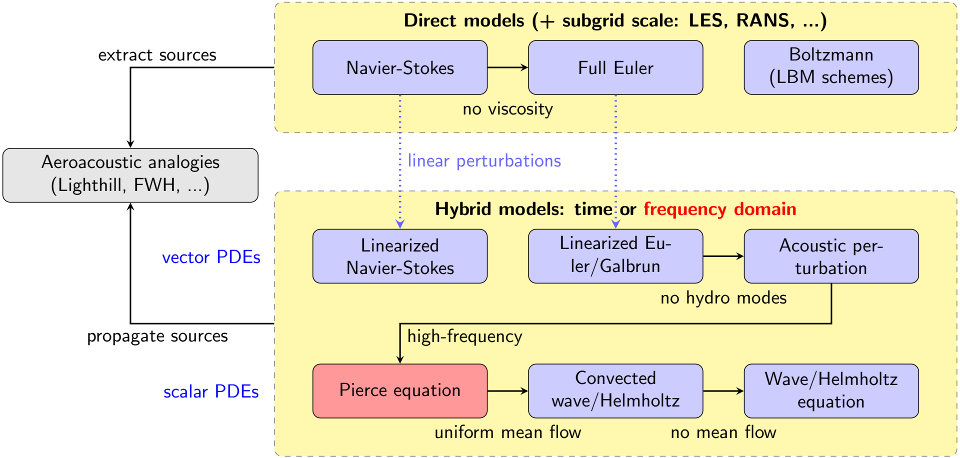
- suitable to modern computer architectures
- applicable to large, complex industrial problems

→ Can serve as basis for optimization routines

Physical models for sound propagation



Physical models for sound propagation



Hybrid model - solve mean flow and acoustic perturbations separately

- We choose the **time-harmonic Pierce Equation**
- Simple but accurate for single tones of turbofan engine intakes and exhausts

Physical model - Pierce Equation

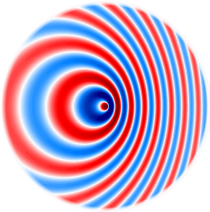
Scalar equation for the acoustic velocity potential u (velocity $\boldsymbol{v} = \nabla u$)

Pierce Equation

$$\rho_0(\boldsymbol{x})\frac{\mathrm{D}_0}{\mathrm{D}t}\left(\frac{1}{\rho_0(\boldsymbol{x})^2 c_0(\boldsymbol{x})^2}\frac{\mathrm{D}_0 u}{\mathrm{D}t}\right) - \nabla\cdot\left(\frac{1}{\rho_0(\boldsymbol{x})}\nabla u\right) = f, \qquad \frac{\mathrm{D}_0}{\mathrm{D}t} := \partial_t + \boldsymbol{v}_0(\boldsymbol{x})\cdot\nabla$$

In frequency domain ($\partial_t \mapsto \mathrm{i}\omega$): Helmholtz-type problem with convection and heterogeneities

Point source in a uniform flow



Mach number $\mathrm{M} = \|\boldsymbol{v}_0\|/c_0 = 0.6$
$\mathrm{M} < 1$ (Subsonic flow)

Physical model - Pierce Equation

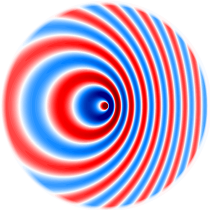
Scalar equation for the acoustic velocity potential u (velocity $\boldsymbol{v} = \nabla u$)

Pierce Equation

$$\rho_0(\boldsymbol{x}) \frac{\mathrm{D}_0}{\mathrm{D}t}\left(\frac{1}{\rho_0(\boldsymbol{x})^2 c_0(\boldsymbol{x})^2}\frac{\mathrm{D}_0 u}{\mathrm{D}t}\right) - \nabla \cdot \left(\frac{1}{\rho_0(\boldsymbol{x})}\nabla u\right) = f, \qquad \frac{\mathrm{D}_0}{\mathrm{D}t} := \partial_t + \boldsymbol{v}_0(\boldsymbol{x}) \cdot \nabla$$

In frequency domain ($\partial_t \mapsto \mathrm{i}\omega$): Helmholtz-type problem with convection and heterogeneities

Point source in a uniform flow



Mach number $\mathrm{M} = \|\boldsymbol{v}_0\| / c_0 = 0.6$
$\mathrm{M} < 1$ (Subsonic flow)

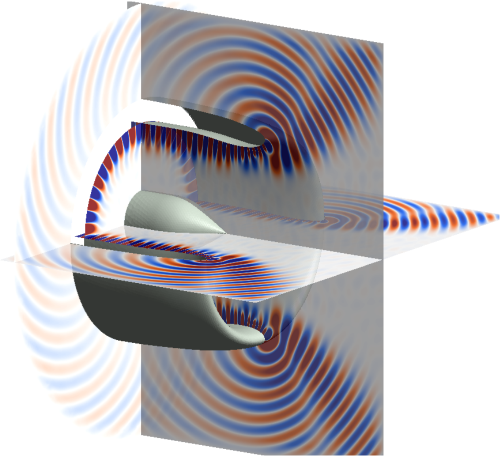
Mathematical difficulties

- Highly oscillatory solution for large ω
- Complex valued, strongly indefinite with ω
- Unbounded domain
- Convection effects

Does not converge with classical iterative methods

Reaching the high frequency limit

State-of-the-art: high-order finite elements (p-FEM) with direct sparse solver (MUMPS)



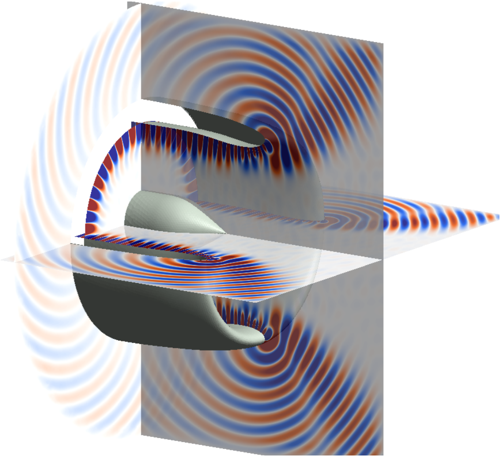
Turbofan intake at $\omega = \omega_{\text{bpf}}$
(≈ 25 wavelengths)

At $\omega = \omega_{\text{bpf}}$: $N_{\text{dofs}} = 10$M, nnz $= 730$M

Direct solver $\rightarrow$ 740 Gb of RAM

Reaching the high frequency limit

State-of-the-art: high-order finite elements (p-FEM) with direct sparse solver (MUMPS)



Turbofan intake at $\omega = \omega_{\text{bpf}}$
(≈ 25 wavelengths)

At $\omega = \omega_{\text{bpf}}$: $N_{\text{dofs}} = 10$M, nnz $=$ 730M

Direct solver $\rightarrow$ 740 Gb of RAM

$\Downarrow$ increase ω?

At $w = 2 \times \omega_{\text{bpf}}$: $N_{\text{dofs}} = 73$M, nnz $=$ 5B

Direct solver $\rightarrow \approx 6$ Tb of RAM

$\mathcal{O}(\omega^3)$ scaling in memory & computational time...

Increasing the upper frequency limit

Scalable domain decomposition method to distribute the memory cost, combining direct and iterative linear solvers

Key ingredients

- High-order finite elements
  - $\rightarrow$ Reduce discretization error (interpolation & dispersion)
  - $\rightarrow$ Increase arithmetic intensity

Increasing the upper frequency limit

Scalable domain decomposition method to distribute the memory cost,
combining direct and iterative linear solvers

Key ingredients

- High-order finite elements
  - → Reduce discretization error (interpolation & dispersion)
  - → Increase arithmetic intensity

Good opportunities for vectorization and multithreading

Increasing the upper frequency limit

Scalable domain decomposition method to distribute the memory cost,
combining direct and iterative linear solvers

Key ingredients

- High-order finite elements
  - → Reduce discretization error (interpolation & dispersion)
  - → Increase arithmetic intensity

  Good opportunities for vectorization and multithreading

- Non-overlapping, substructured optimized Schwarz domain decomposition solver
  - → Sparse direct solver on smaller problems posed on each subdomain
  - → Krylov subspace solver on small number of interface unknowns between subdomains
  - → High-order transmission conditions for quasi-optimal convergence of Krylov solver

Increasing the upper frequency limit

Scalable domain decomposition method to distribute the memory cost,
combining direct and iterative linear solvers

Key ingredients

- High-order finite elements
  - → Reduce discretization error (interpolation & dispersion)
  - → Increase arithmetic intensity

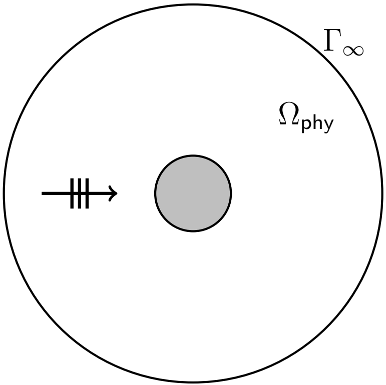
  Good opportunities for vectorization and multithreading

- Non-overlapping, substructured optimized Schwarz domain decomposition solver
  - → Sparse direct solver on smaller problems posed on each subdomain
  - → Krylov subspace solver on small number of interface unknowns between subdomains
  - → High-order transmission conditions for quasi-optimal convergence of Krylov solver

  Good fit for distributed memory cluster architectures

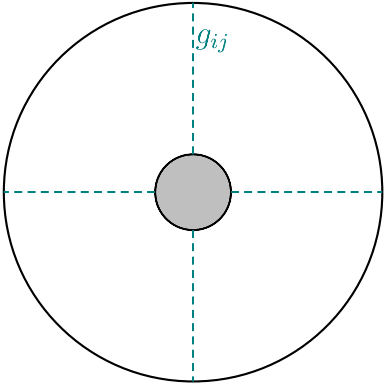
Domain decomposition framework: Non-overlapping optimized Schwarz

Toy example: disk scattering by a plane wave



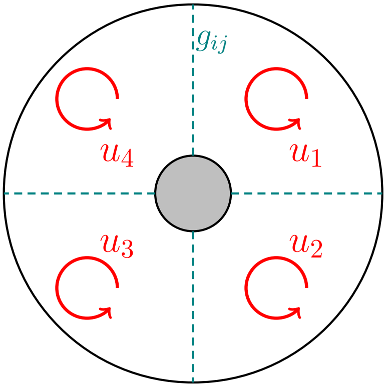
Domain decomposition framework: Non-overlapping optimized Schwarz

Partition $\Omega = \bigcup_{i=0}^{N_{\text{dom}}-1} \Omega_i$ into subdomains



Domain decomposition framework: Non-overlapping optimized Schwarz

Partition $\Omega = \bigcup_{i=0}^{N_{\text{dom}}-1} \Omega_i$ into subdomains



Iterate until convergence

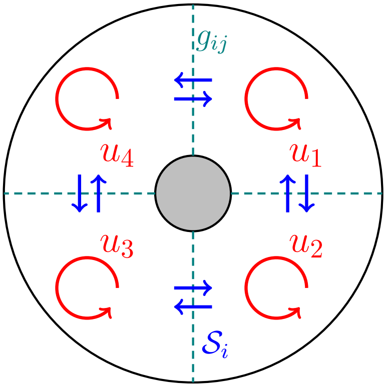
1. Solve the volume subproblems u_i with boundary conditions

Domain decomposition framework: Non-overlapping optimized Schwarz

Partition $\Omega = \bigcup_{i=0}^{N_{\text{dom}}-1} \Omega_i$ into subdomains

Iterate until convergence

1. Solve the volume subproblems u_i with boundary conditions
2. Update the interfaces unknowns $\boldsymbol{g} = (g_{ij}, g_{ji})$ through transmission conditions $(\mathcal{S}_i, \mathcal{S}_j)$ $\Leftrightarrow$ solve surface problem

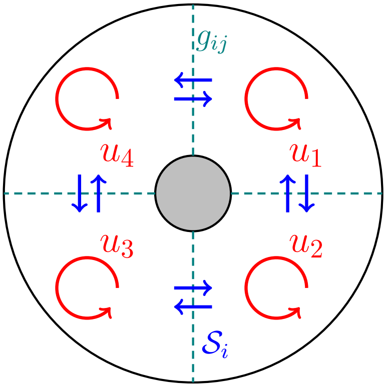


Domain decomposition framework: Non-overlapping optimized Schwarz

Partition $\Omega = \bigcup_{i=0}^{N_{\text{dom}}-1} \Omega_i$ into subdomains

Iterate until convergence

1. Solve the volume subproblems u_i with boundary conditions
2. Update the interfaces unknowns $\boldsymbol{g} = (g_{ij}, g_{ji})$ through transmission conditions $(\mathcal{S}_i, \mathcal{S}_j) \Leftrightarrow$ solve surface problem



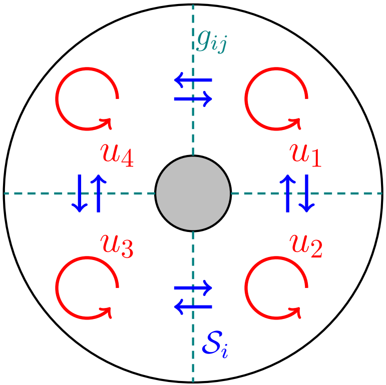
Optimal convergence with the **Dirichlet-to-Neumann** (DtN) operator for the complementary of the subdomain under consideration

Domain decomposition framework: Non-overlapping optimized Schwarz

Partition $\Omega = \bigcup_{i=0}^{N_{\text{dom}}-1} \Omega_i$ into subdomains

Iterate until convergence

1. Solve the volume subproblems u_i with boundary conditions
2. Update the interfaces unknowns $\boldsymbol{g} = (g_{ij}, g_{ji})$ through transmission conditions $(\mathcal{S}_i, \mathcal{S}_j)$ $\Leftrightarrow$ solve surface problem



Optimal convergence with the **Dirichlet-to-Neumann** (DtN) operator for the complementary of the subdomain under consideration

We propose a new family of operators $(\mathcal{S}_i, \mathcal{S}_j)$ that provide **accurate approximations of the DtNs in the high-frequency regime**, based on microlocal analysis

GmshDDM: scalable parallel solver for time-harmonic waves problems

GmshDDM is new C++ library implementing a distributed optimized Schwarz domain decomposition solver with these new high-order transmission conditions

GmshDDM: scalable parallel solver for time-harmonic waves problems

GmshDDM is new C++ library implementing a distributed optimized Schwarz domain decomposition solver with these new high-order transmission conditions

GmshDDM is based on **GmshFEM**, a high-performance finite element library, itself built on top of the **Gmsh** application programming interface

GmshDDM: scalable parallel solver for time-harmonic waves problems

GmshDDM is new C++ library implementing a distributed optimized Schwarz domain decomposition solver with these new high-order transmission conditions

GmshDDM is based on **GmshFEM**, a high-performance finite element library, itself built on top of the **Gmsh** application programming interface

Main features:

- High-order unstructured finite element meshes (1D, 2D, 3D)
- Mesh partitioning using METIS
- Arbitrary order hierarchical basis functions (H^1, H(curl), ...)
- Symbolic specification of weak forms
- Dense linear algebra with Eigen
- Sparse linear algebra with PETSc, sparse LU with MUMPS
- Hybrid MPI/OpenMP parallelization

GmshDDM: scalable parallel solver for time-harmonic waves problems

GmshDDM is new C++ library implementing a distributed optimized Schwarz domain decomposition solver with these new high-order transmission conditions

GmshDDM is based on **GmshFEM**, a high-performance finite element library, itself built on top of the **Gmsh** application programming interface

Main features:

- High-order unstructured finite element meshes (1D, 2D, 3D)
- Mesh partitioning using METIS
- Arbitrary order hierarchical basis functions (H^1, H(curl), ...)
- Symbolic specification of weak forms
- Dense linear algebra with Eigen
- Sparse linear algebra with PETSc, sparse LU with MUMPS
- Hybrid MPI/OpenMP parallelization

Open source under GNU AGPL v3

First GmshDDM runs on LUMI

Preparatory project, followed by project in **LUMI-BE Regular Access** call

- A few (small) teething problems
  - → Cray toolchain can be tricky on some 3rd-party dependencies (e.g. OpenCASCADE) → GNU toolchain
  - → Cray MPI less permissive than OpenMPI (MPI asynch)

First GmshDDM runs on LUMI

Preparatory project, followed by project in **LUMI-BE Regular Access** call

- A few (small) teething problems
  - → Cray toolchain can be tricky on some 3rd-party dependencies (e.g. OpenCASCADE) → GNU toolchain
  - → Cray MPI less permissive than OpenMPI (MPI asynch)
  - → (...and one tricky bug in GmshDDM that drove me crazy!)

First GmshDDM runs on LUMI

Preparatory project, followed by project in **LUMI-BE Regular Access** call

- A few (small) teething problems
  - → Cray toolchain can be tricky on some 3rd-party dependencies (e.g. OpenCASCADE) → GNU toolchain
  - → Cray MPI less permissive than OpenMPI (MPI asynch)
  - → (...and one tricky bug in GmshDDM that drove me crazy!)

- LUMI-C heavily used, but quite stable after latest upgrades (some jobs still fail due to node connection timeout at launch)

First GmshDDM runs on LUMI

Preparatory project, followed by project in **LUMI-BE Regular Access** call

- A few (small) teething problems
  - → Cray toolchain can be tricky on some 3rd-party dependencies (e.g. OpenCASCADE) → GNU toolchain
  - → Cray MPI less permissive than OpenMPI (MPI asynch)
  - → (...and one tricky bug in GmshDDM that drove me crazy!)
- LUMI-C heavily used, but quite stable after latest upgrades (some jobs still fail due to node connection timeout at launch)
- **Excellent performance** overall
  - → Meshing and partitioning on a single LUMI-C node
  - → Variety of runs up to 512 LUMI-C nodes

First GmshDDM runs on LUMI

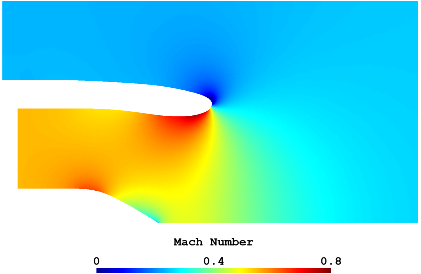
Preparatory project, followed by project in **LUMI-BE Regular Access** call

- A few (small) teething problems
  - → Cray toolchain can be tricky on some 3rd-party dependencies (e.g. OpenCASCADE) → GNU toolchain
  - → Cray MPI less permissive than OpenMPI (MPI asynch)
  - → (...and one tricky bug in GmshDDM that drove me crazy!)
- LUMI-C heavily used, but quite stable after latest upgrades (some jobs still fail due to node connection timeout at launch)
- **Excellent performance** overall
  - → Meshing and partitioning on a single LUMI-C node
  - → Variety of runs up to 512 LUMI-C nodes
  - → Thanks to the LUST team, and in particular to Orian Louant!

3D turbofan intake and exhaust radiation problems

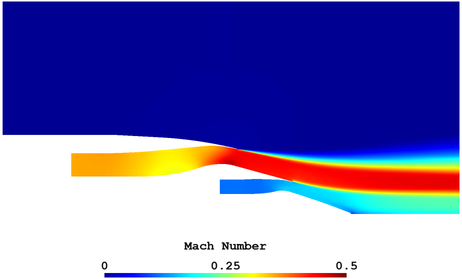
Given a flight configuration (precomputed mean flow, interpolated on the acoustic mesh), predict the radiated noise

Intake typical mean flow



Compute noise from the fan (fixed annular duct mode), at multiples of the blade passing frequency $\omega_{\text{bpf}}/(2\pi) = 1300$ Hz

Exhaust typical mean flow

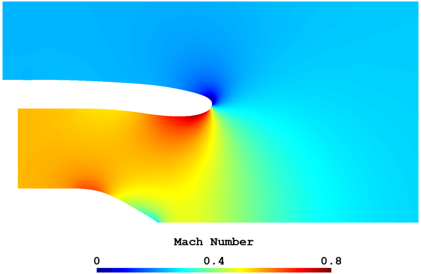


Compute noise from plane wave in the heated core jet at $f > 8000$ Hz.

3D turbofan intake and exhaust radiation problems

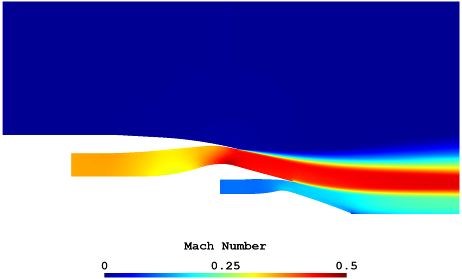
Given a flight configuration (precomputed mean flow, interpolated on the acoustic mesh), predict the radiated noise

Intake typical mean flow



Compute noise from the fan (fixed annular duct mode), at multiples of the blade passing frequency $\omega_{\text{bpf}}/(2\pi) = 1300$ Hz

Exhaust typical mean flow

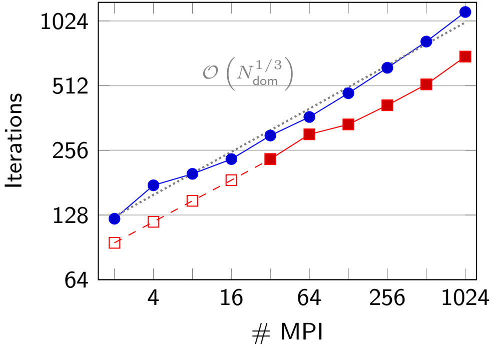


Compute noise from plane wave in the heated core jet at $f > 8000$ Hz.

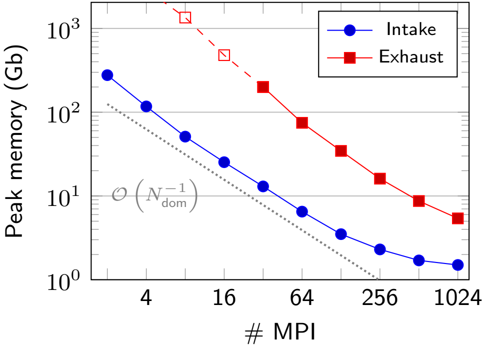
Cylindrical active and passive perfectly matched layers (PMLs) are used to truncate the computational domains

Scalability on small/medium-sized problems

	frequency [Hz]	N_{dofs}	nnz	direct solver est. memory	#tetrahedra
Intake	1300	10M	730M	740Gb	890k
Exhaust	2×7497	86M	6.3B	≈ 10Tb	7.7M



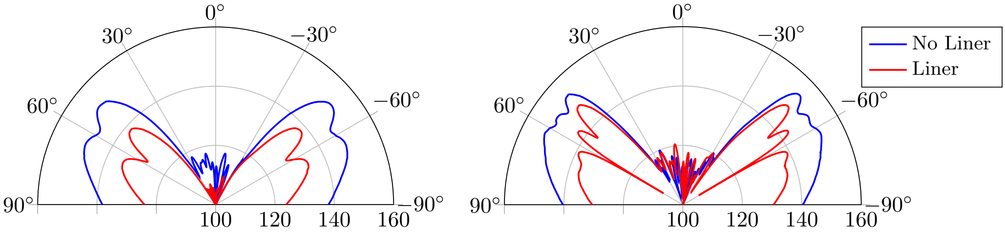
(a) Number of GMRES iterations to reach a relative residual of $r_I = 10^{-6}$.



(b) Peak memory usage over all processes

Engineering results

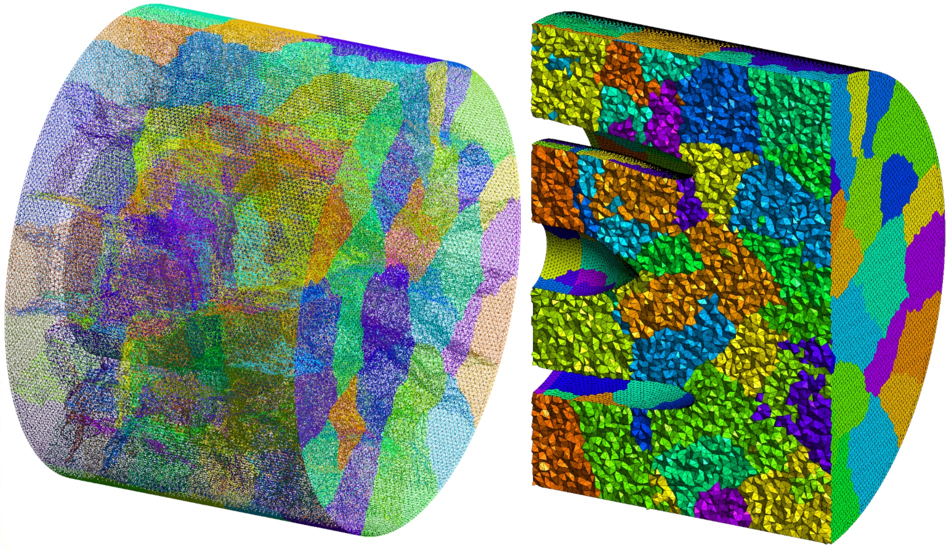
Study of the noise reduction thanks to the acoustic lining



1 BPF   2 BPF

Sound pressure level (dB) near field directivity with and without acoustic lining along a semi-circle of radius 2m centered on the spinner tip in the XY plane.

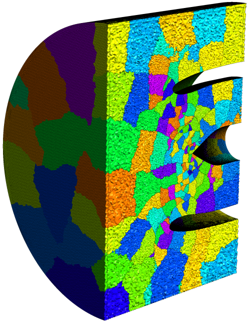
Typical mesh and subdomain partitioning: Intake problem

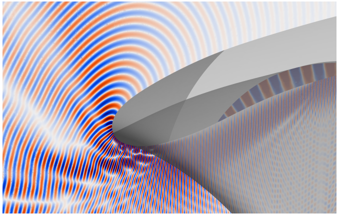


$$N_{\text{dom}} = 512$$

Reaching high frequency: Intake problem at $f = 6500$ Hz (5 BPF!)

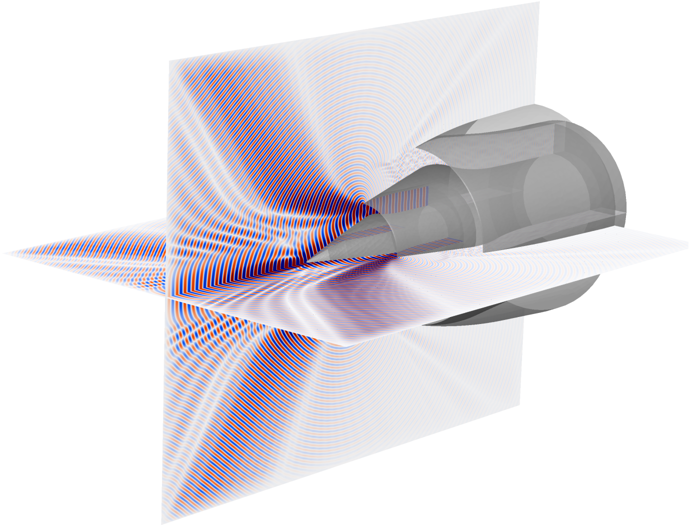
#cores (MPI×threads)	N_{dofs}	nnz	peak mem./process	pre-pro	GMRES	#iter.
1024×64	1.1B	167B	70Gb	24min	3h24min	1293

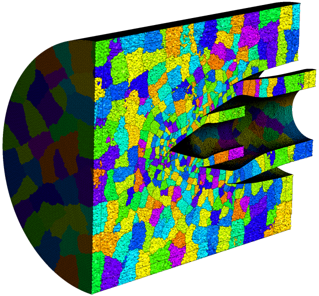


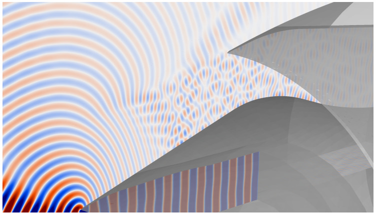


Reaching high frequency: Exhaust problem at $f = 40000$ Hz

#cores (MPI×threads)	N_{dofs}	nnz	peak mem./process	pre-pro	GMRES	#iter.
4096×16	1.3B	96B	18.4Gb	1min	14min	555







Conclusion and perspectives

New distributed memory solver GmshDDM for high frequency flow acoustics

- New quasi-optimal domain decomposition approach
- Performance and scalability validated vs. theoretical bounds
- First LUMI runs allowed to solve large scale problems at unprecedented high frequencies

The only real limitation of the current version of the code is the available number of nodes and the related total amount of memory

Conclusion and perspectives

New distributed memory solver GmshDDM for high frequency flow acoustics

- New quasi-optimal domain decomposition approach
- Performance and scalability validated vs. theoretical bounds
- First LUMI runs allowed to solve large scale problems at unprecedented high frequencies

The only real limitation of the current version of the code is the available number of nodes and the related total amount of memory

Future developments

- GPU acceleration of the iteration phase
  - → Preliminary tests on LUCIA (expected challenge: moving from NVIDIA to AMD GPUs)
- Ongoing work on new applications: electromagnetic waves and elastic waves

References

Open source implementation

- **Gmsh** (`https://gmsh.info`): `https://gitlab.onelab.info/gmsh/gmsh`
- **GmshFEM**: `https://gitlab.onelab.info/gmsh/fem`
- **GmshDDM**: `https://gitlab.onelab.info/gmsh/ddm`

References

Open source implementation

- **Gmsh** (`https://gmsh.info`): `https://gitlab.onelab.info/gmsh/gmsh`
- **GmshFEM**: `https://gitlab.onelab.info/gmsh/fem`
- **GmshDDM**: `https://gitlab.onelab.info/gmsh/ddm`

Mathematical details

- Construction and numerical assessment of local absorbing boundary conditions for heterogeneous time-harmonic acoustic problems. P. Marchner, X. Antoine, C. Geuzaine, H. Bériot, *SIAM Journal on Applied Mathematics* 82(2) pp 476-501, 2022.
- A domain decomposition solver for large scale time-harmonic flow acoustics problems. P. Marchner, H. Beriot, S. Le Bras, X. Antoine, C. Geuzaine, Submitted to *SIAM Journal on Scientific Computing*, 2023. Preprint available on `https://hal.science/hal-04254633`

References

Open source implementation

- **Gmsh** (https://gmsh.info): https://gitlab.onelab.info/gmsh/gmsh
- **GmshFEM**: https://gitlab.onelab.info/gmsh/fem
- **GmshDDM**: https://gitlab.onelab.info/gmsh/ddm

Mathematical details

- Construction and numerical assessment of local absorbing boundary conditions for heterogeneous time-harmonic acoustic problems. P. Marchner, X. Antoine, C. Geuzaine, H. Bériot, *SIAM Journal on Applied Mathematics* 82(2) pp 476-501, 2022.
- A domain decomposition solver for large scale time-harmonic flow acoustics problems. P. Marchner, H. Beriot, S. Le Bras, X. Antoine, C. Geuzaine, Submitted to *SIAM Journal on Scientific Computing*, 2023. Preprint available on https://hal.science/hal-04254633

Thanks for your attention

✉ cgeuzaine@uliege.be

LUMI Cifre ANRT Horizon2020 European Union Funding for Research & Innovation fnrs LA LIBERTÉ DE CHERCHER CÉCI