A fast coupling interface for a high-order acoustic perturbation equations solver with finite volume CFD codes to predict complex jet noise

Abstract

In this project, our aim was to develop a fast interface for a high-order acoustic perturbation equations solver, APESolver, to couple with finite volume Navier-Stokes equations codes that are widely used in the UK Turbulence, Applied Aerodynamics and ARCHER communities. Driven by more stringent design requirements, the demand for multi-physics modelling capabilities is growing fast. Well-established modelling tools for individual disciplines are in need of running simultaneously while exchanging coupling data efficiently. APESolver, as part of the Nektar++ software framework (www.nektar.info), is an open-source p-order polynomial spectral element code, which solves the linear Acoustic Perturbation Equations (APE) to obtain flow-induced acoustic fields in time and space. It is ideal for capturing sound wave dilatation and scattering around complex geometry with high accuracy, such as jet noise emitted from civil aircraft, where source terms can be determined from solutions of compressible Navier-Stokes (N-S) equations using well-established finite volume CFD codes. Since the sound source data is volumetric and time-dependent, a fast, parallel and memory efficient coupling is essential and the best way to achieve this is via MPI.

To couple with APESolver, we chose two mainstream finite volume N-S codes: HYDRA, and OpenFOAM based on their popularity and availability within the community (either proprietary or free). The N-S codes chosen also feature cell-vertex and cell-centred control volumes, allowing our coupling implementation to work with both types of connectivity as commonly used within the CFD community.

1.1 Introduction

The world's civil aircraft fleet is almost doubling in size every two decades, creating unprecedented impact on community due to noise generated by these aircraft. Driven by economic growth, major airports' plan for expansion can only make the problem worse, if the noise issue is not fundamentally tackled. According to ACARE (Advisory Council for Aviation Research and innovation in Europe) FlightPath 2050 targets, the perceived noise emission from flying aircraft should be reduced by 65% relative to levels in 2000.

The jet noise research community has made encouraging progress in recent years, thanks to advances in HPC and growing utilisation of HPC resources. Computational approaches are now seen increasingly promising in modelling isolated simple jets [1], but more needs to be done to understand installed jets, where jet streams interact with aircraft wing, flaps and pylon, causing sound waves reflecting and scattering. Due to this added complexity, deploying a more traditional FW-H surface integral method, which only takes flow information on a surface enclosing noise sources to project values at an observer location, may be challenging, and resolving a full acoustic propagation field is more accurate and straightforward, despite being more expensive.

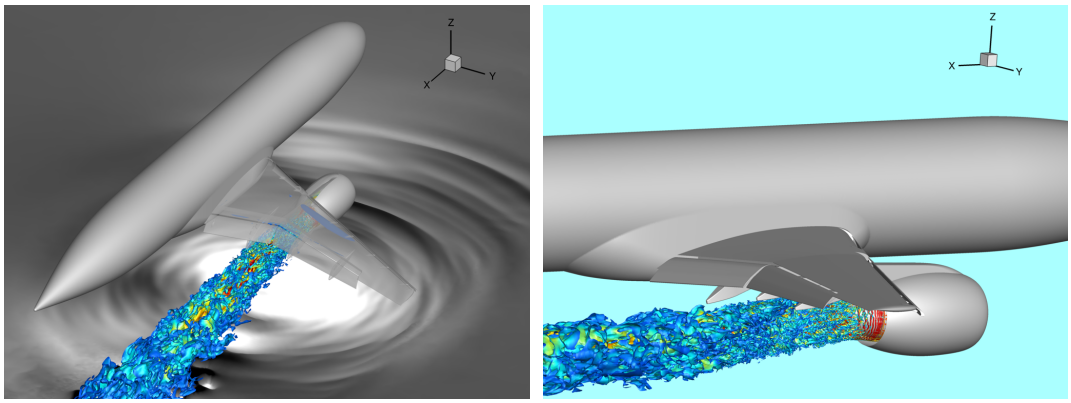


Figure 1: Jet wing/flap/pylon interaction.

Solving the Acoustic Perturbation Equations (APE) not only provides a full propagation field, but more crucially has the ability to filter out pseudo noise caused by near-field hydrodynamic signals [2]. From a computational perspective, APE requires 3D volume source data, which no doubt, if using a file-based coupling, will significantly slow down both the flow and acoustic codes' parallel performance. Hence, the motivation to develop an on-the-fly transparent coupling between flow and acoustic codes is strong. It is worth noting that such an interface is not limited to flow-acoustic coupling, and in fact once an API library is created it can be applied to a wide range of flow-flow (e.g. combustor to turbine flows) and flow-solid (e.g. conjugate heat transfer) couplings.

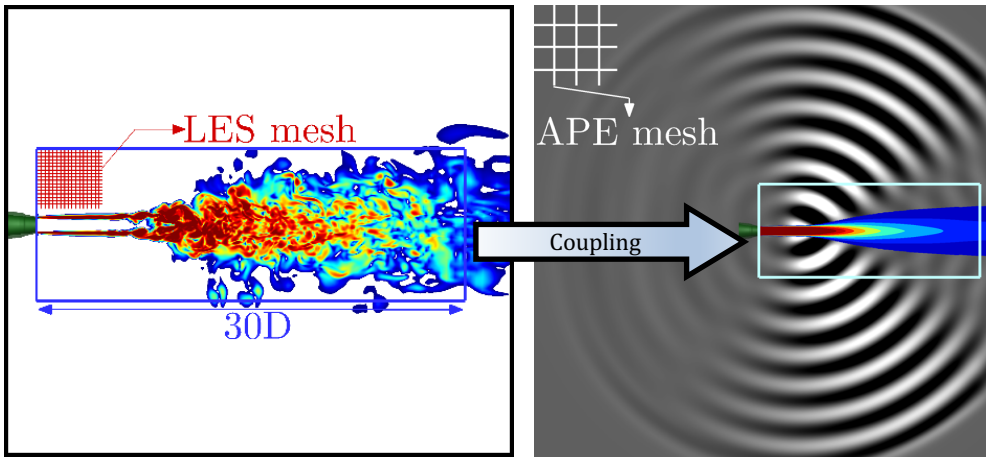


Figure 2: Coupling APE with captured noise sources

1.2 Objectives and Success Metrics

To achieve our project aim, we proposed four main objectives:

i) To implement coupling functionalities between HYDRA and APESolver within the source codes using multiple MPI_COMM_WORLD communicators and a low-level interpolation library CWIPI.

This involves additional coding for data preparation and exchange at given points of a solution procedure. If successful, both codes should be running at speeds that are no less than 85% of their standalone speeds. The volumetric data should be passed correctly, and existing file-based coupling results will be used to validate.

Outcome: The implementation of the library in HYDRA was successful thanks to the receiving subroutines of the CWIPI library being already implemented in the APESolver. The coupling implementation was validated successfully using the existing file-based methodology applied to a cylinder vortex shedding case. Even though the test was only made with meshing containing hexahedral elements, further tests extended to other types of elements should be straightforward.

ii) To implement the same functionalities as i) for OpenFOAM coupled with APESolver.

Implementation this time will only be added within the cell-centred CFD code. The APESolver side's previous implementation will be reused. The same success metrics as i) will be used.

Outcome: The implementation of CWIPI in OpenFOAM was successful using rhoPimpleFOAM. The same cylinder vortex shedding case was used for the validation of the implementation.

iii) As a main objective, to create a library of API subroutine/functions that extracts implementations (in i) and ii)) from the CFD codes and APESolver to form a wrapper layer so that the coupling is transparent to both sides.

If successful, the new interface library should contain all intended functionalities and should produce the same results as those in i) and ii). Speed wise there should be no change, but memory consumption may see a little increase due to additional function allocations. The library will work with other codes in similar principles.

Outcome: An API wrapper library has been created to contain as much as possible the implementation of coupling, in Fortran (using HYDRA's implementation as a reference), and in C++ for OpenFOAM.

iv) To make the project's outcome available to the communities and to produce

detailed documentation for users and developers.

The interface library itself will be made available on Github as open-source free software. The modified codes in delivering objectives i) and ii) will be made into a new branch and available according to their respective licenses.

Outcome: The modified codes (under their original license agreements) and the implemented API wrapper libraries are made available on Github.

1.3 Coupling Interface Mechanism

To send the source information from the LES codes to Nektar++ in an efficient way, the MPI capabilities have to be used to take advantage of the parallel communication between the different CPU's. In this work, the communication between the codes is achieved via the third-party low-level interpolation library called CWIPI [3], which is developed at ONERA. In Lackhove et al. [4], a detailed explanation of the implementation of CWIPI in Nektar++ and its application on combustion noise is given. The present work extends those capabilities of Nektar++ for its application on jet noise propagation using HYDRA and OpenFOAM. CWIPI makes use of the Multiple Program-Multiple Data (MPMD) mode of the "extended" MPI-1 standard used for launching different executables that share the same MPI_global_communicator.

Two different ways for the implementation of CWIPI in the two solvers have been used in this project. In the first one, the required CWIPI subroutines/functions were hardcoded and embedded in the source codes (Figure 3). Within the two LES source codes additional data information (such as the connectivity table and nodal/cell centre positions) had to be prepared for the correct exchange of the acoustic sources. However, the main goal of the project was the extraction of the coupling functionality from the LES source codes, so that it could be easily implemented by other users into their own solvers and/or OpenFOAM versions. Figure 4 shows the final implementation of the coupling interface that works as a wrapper layer between the LES/Nektar++ codes and CWIPI.

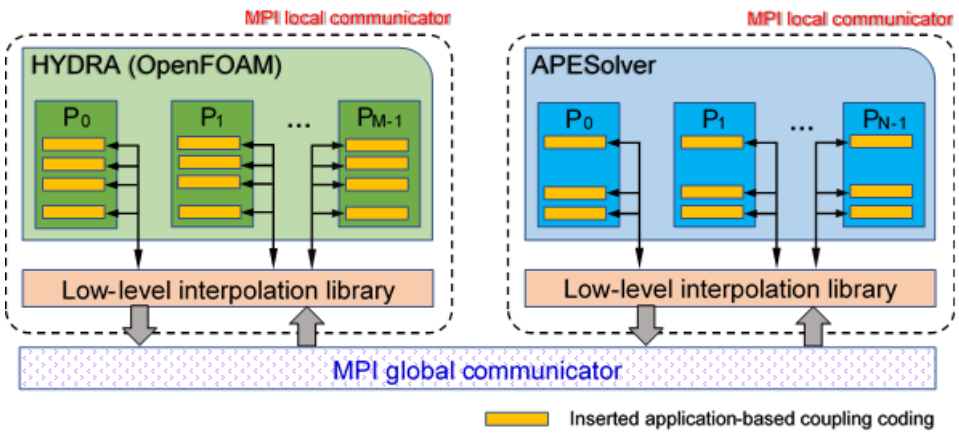


Figure 3: Diagram of coupling procedure with extensive modification to source codes

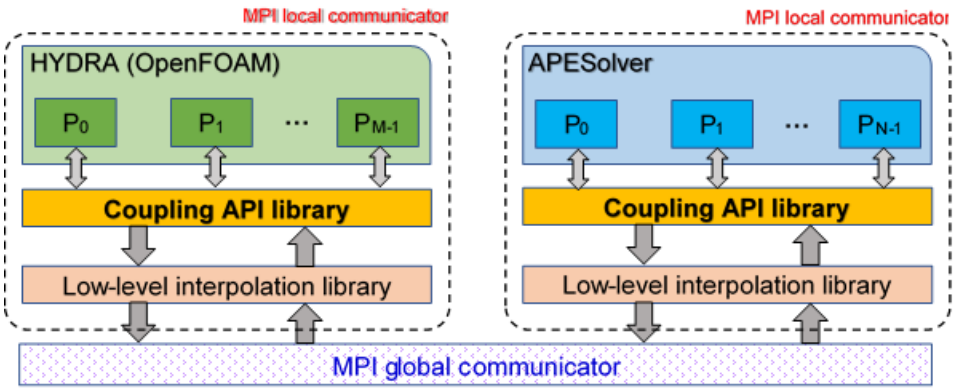


Figure 4: Diagram of coupling procedure with the common API library interface

The implementation of the API library interface minimises code modification from the user. Only minimum changes remain which are CWIPI requirements for the communication between the two codes. Depending on their functionality these can be classified in three different categories:

1. **Initialization and termination**: the initialization of the coupling between the codes requires the modification of the mpi_init and mpi_comm_rank subroutines, so that the two applications have the same MPI_COMM_WORLD communicator. The only modification required to the codes is the addition of the cwipi_init subroutine that assigns a local communicator to each application so that they can be identifiable but still work under the same global communicator. The termination of the coupling requires the replacement of the mpi_finalize subroutine by the cwipi_finalize subroutine.

2. **Definition of coupling characteristics and mesh domains:** since the coupling implemented allows for the use of different grids in each application, a definition of a coupling environment is needed for the two solvers to identify the regions in which the interpolation of the flow variables is going to take place. In the present project, these definitions have been implemented in two different ways.

   2.1 Hardcoding them into the source codes:

   Before the partitioning of the mesh is transferred to CWIPI, a bunch of preliminary subroutine calls to specify the coupling characteristics:

   - Local and distant coupling name identifiers.
   - Definition of number of variables to send and name of those variables.
   - Type of solver: cell center/vertex, static/dynamic mesh, 2D/3D.
   - Synchronization of local and distant solvers.
   - Acquisition of distant integer tag to identify the MPI send calls.

   Then, the mesh is sent to CWIPI via a call to cwipi_define_mesh. This subroutine requires the information of the number and position of vertices and the connectivity table. Once the partition of the two meshes is transferred, CWIPI begins to locate the nodes/cell centers between the two application to prepare the interpolation of the flow variables in a later step. This is done by calling the cwipi_locate subroutine.

2.2 Creating a wrapping layer:

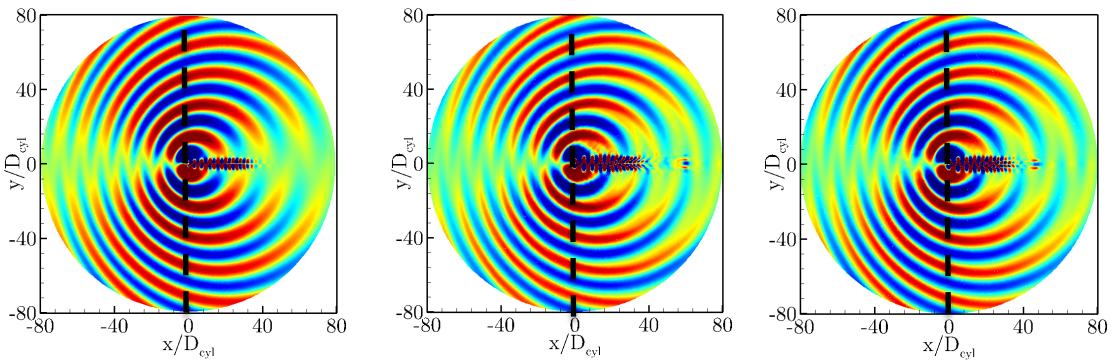
All the above subroutine calls where brought together into a single library that only requires one call into the source code. This considerably simplifies the procedure of further application.

3. **Interpolation and exchange of data:** these subroutines are used to gather and transfer the flow variables data that to the APESolver. They are equivalent to the mpi_send, mpi_receive and mpi_wait subroutines. The first subroutine needed is called cwipi_issend and it requires the number and names of variables sent and the array with the data of the acoustic sources. It also contains an exchange request integer that, together with the cwipi_wait_issend subroutine, allows to detect if the other application has correctly received the data information.

1.4 Test cases

For the validation of the LES/APE coupling methodology three different cases have been tested using both HYDRA and OpenFOAM solvers coupled with Nektar++. This section summarizes some of the results obtained for each case. More detailed results can be found in Moratilla-Vega et al. [5].

The first case is a 2D cylinder immersed in a crossflow at low Mach and Reynolds numbers. Since the mesh requirements are not very high, the file coupling methodology that was already developed in [6] is used for comparison. The acoustic field, represented by pressure perturbation, is presented in Figure 5 for the standalone DNS simulation, the file-coupling approach and the MPI parallel interface method. Figure 6 shows the comparison between the cases, for the pressure distribution along the black dashed lines shown in Figure 5. The agreement between the three cases is encouraging.



a) DNS b) APE-MPI coupling c) APE-File coupling

Figure 5: Acoustic field for the CFD and CFD/APE simulations

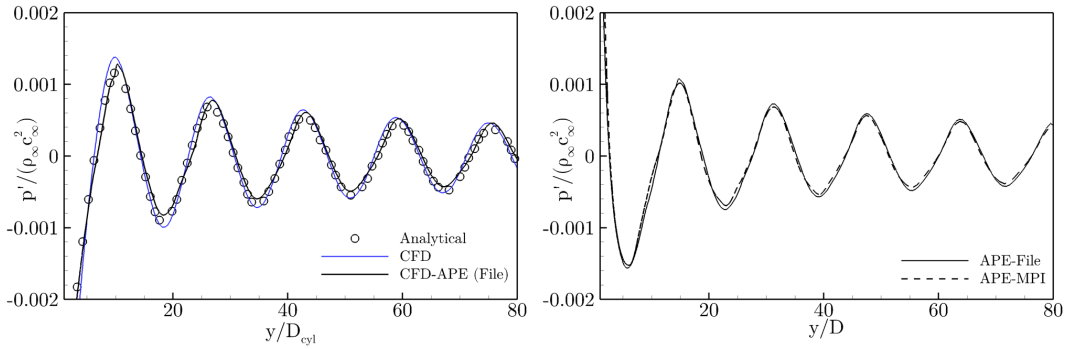


Figure 6: Coupling APE with captured noise sources

To demonstrate the different efficiencies between the traditional file-coupling approach and the present MPI Interface-coupling method, this case was run in different combination of processors on ARCHER to obtain the scalability of the standalone and coupled applications. The scalability of the standalone applications it was found to be very good. However, when using the file-based method, a flattening trend is observed after using 48 nodes. This poor scalability is caused by the limited I/O speed of the disk system of the cluster. Conversely, the results found with the MPI Interface method are quite satisfactory. Even tough the size of the application is smaller than in a realistic scenario, there is an insignificant penalty observed in the transmission of information between the codes. Due to the great efficiency difference between the file and MPI coupling approaches, for the remaining cases only the MPI Interface method was used.

Table 1: Parallel speedup comparison for running standalone and coupled cases.

No. of Processors	Ideal	Nektar++ Standalone	HYDRA Standalone	OpenFOAM Standalone	File Coupling	MPI Coupling
14	1	1.00	1.00	1.00	1.00	1.00
28	2	1.86	1.80	1.90	1.54	1.94
56	4	3.69	3.4	3.60	2.26	3.87
112	8	7.39	6.60	7.20	2.91	7.61
224	16	15.41	13.04	12.29	3.40	16.48
448	32	36.84	25.6	-	3.62	36.81
672	48	58.65	37.92	-	3.74	57.61

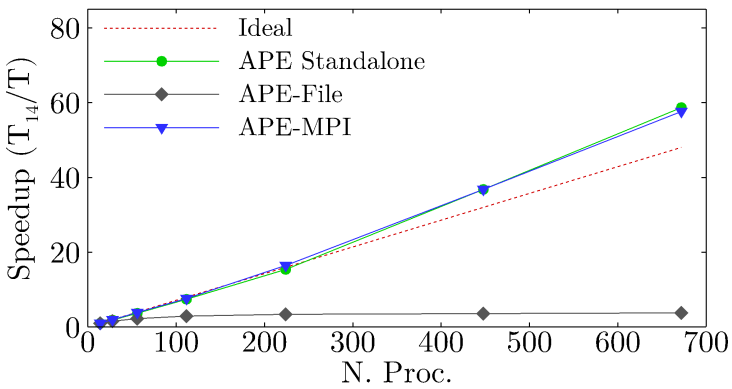


Figure 7: Coupling APE with captured noise sources

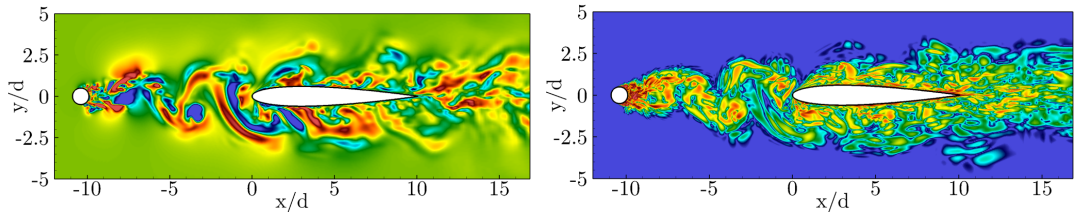


Figure 8: Contours of instantaneous spanwise vorticity and velocity fields obtained with HYDRA.

The second case considered for the validation of the methodology was a rod wake-airfoil interaction case, which is considered a benchmark for CAA codes validation. The case is run in a 2.5D LES mesh and a 2D APE mesh. The Mach number of the case is still low, but the Reynolds number is much higher, giving a more realistic turbulent flow. The instantaneous flow obtained with HYDRA is shown in Figure 8 through contours of spanwise vorticity and velocity fields.

The acoustic propagation results obtained with the MPI Interface coupling approach using HYDRA and Nektar++ are presented in Figure 9. The expected tonal dipole component of this configuration appears in the simulation. Furtermore, the agreement with the experiments for the noise at an observer at $y/D_j = 185$ are satisfactory.

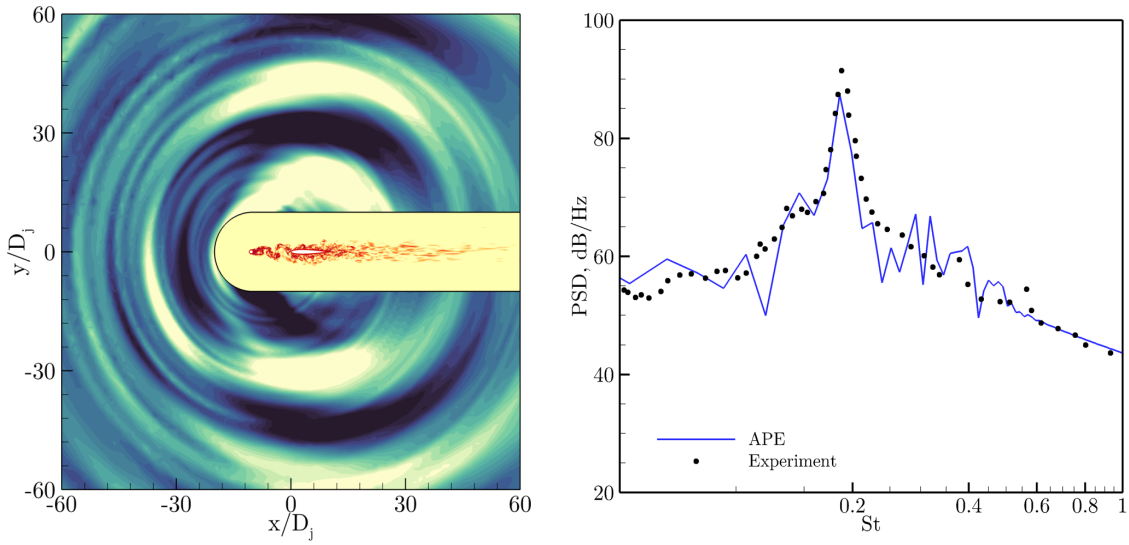


Figure 9: Contours of pressure perturbation (left) and power spectral density (right). Results obtained with HYDRA-Nektar++ coupled implementation.

The validate the coupling methodology in a realistic 3D turbulent jet configuration, a high Mach high Reynolds number jet case was used. Results are presented here for the OpenFOAM-Nektar++ coupling approach using the developed API interface. Due to the much bigger sizes of the meshes the number of points that had to be exchanged between the solvers is much higher than in the two previous cases. Therefore, the transfer of information from OpenFOAM to Nektar++ had a bigger overhead time. Nonetheless, preliminary results have shown that the decrease in speed compared to the standalone run of each solver is approximately 10%. Figure 10 shows the preliminary acoustic field obtained

with the coupled OpenFOAM-Nektar++ configuration. There is a clear improvement of the present methodology over the noise propagation with the standalone OpenFOAM, which has a considerably higher numerical dissipation due to the second-order discretization scheme used.

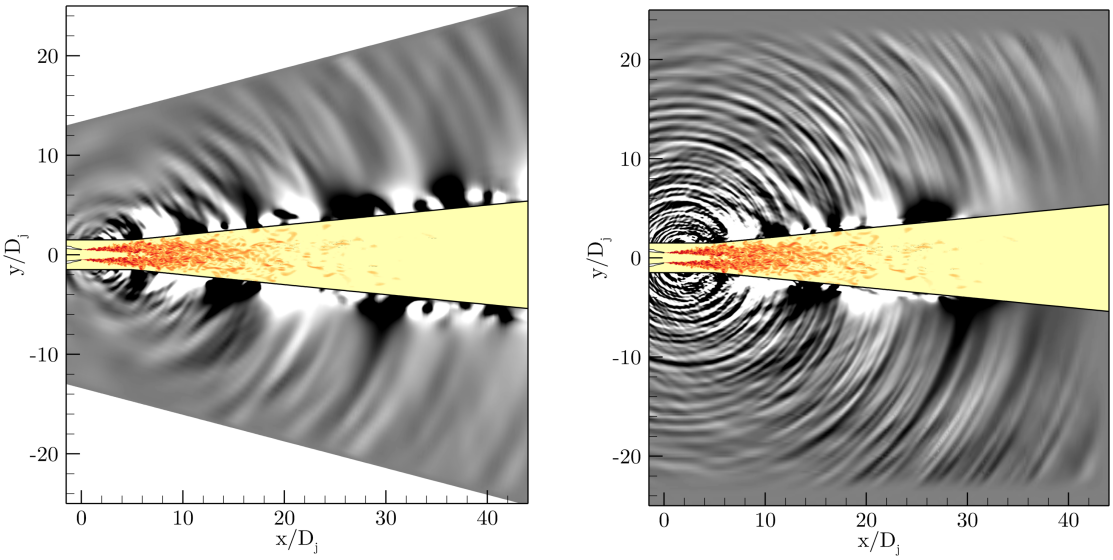


Figure 10: Acoustic field for the standalone OpenFOAM and the coupled OpenFOAM/Nektar++ cases.

1.5 Conclusion

An LES-APE coupling strategy for the prediction of noise in jets and other configurations has been presented with a new parallel coupling interface implemented. Three different codes have been used in the project: two flow solvers (HYDRA and OpenFOAM) and an acoustic propagator (APESolver of Nektar++). The coupling between the codes is done using a parallel-interface external library called CWIPI. Two different implementation of the parallel-interface has been carried out. In the first one the subroutines required for the exchange of data were hard coded into the two solvers. In the second implementation, most of the necessary subroutines were wrapped in an API layer that is invisible to the two codes. With this second approach few modifications to the source codes are needed. The efficiency of the coupling technique has been tested and compared against the standalone APE code and a traditional file-based coupling approach that was presented in [6]. The scalability test shows promising results as the overhead time of the coupling approach is minimum. The LES-APE methodology is also tested for two configurations that are related to the installed jet problem. First, a rod-airfoil interaction case is presented giving the expected noise prediction when compared with the experiment. It also demonstrates the possibility of using unstructured meshes when complex geometries are involved. Finally, a 3D turbulent jet case at Re=1,000,000 is also studied as part of the tests and results met expectations. As for further enabled studies, an installed jet-flat plate case is being studied while more complex and realistic jet-wing-flap configurations are expected to be examined thereafter.

1.6 Acknowledgment

This work was funded under the embedded CSE programme of the ARCHER UK National Supercomputing Service (http://www.archer.ac.uk). The authors would like to thank EPSRC for the computational time made available on the UK supercomputing facility ARCHER via the UK Turbulence Consortium (EP/R029326/1 for the allocation period 2018-2022). The support provided by Kilian Lackhove (Technische Universität Darmstadt) for the implementation of CWIPI is gratefully acknowledged.

1.7 References

[1] Lele SK, Nichols JW (2014) A second golden age of aeroacoustics? *Phil.Trans. R. Soc. A*, 372:20130321.

[2] Ewert R, Schröder W (2003) Acoustic perturbation equations based on flow decomposition via source filtering, *J. Comp. Phys.*, 188(2):365-398.

[3] Refloch, A., et al., (2011), ‘‘CEDRE Software,” *Tech. rep.*, *AerospaceLab*.

[4] Lackhove, K., Sadiki, A., and Janicka, J., (2017), ‘‘Efficient Three Three-Dimensional Domain Combustion Noise Simulation of a Premixed Flame Using Acoustic Perturbation Equations and Incompressible LES,” in “*ASME Turbo Expo 2017: Turbomachinery Technical Conference and Exposition,” American Society of Mechanical Engineers*.

[5] Moratilla-Vega, M., et al., (2018), ‘‘An Efficient LES-Acoustic Coupling Method for Sound Generation and High Order Propagation from Jets,” *10th International Conference on Computational Fluid Dynamic Proceedings.*

[6] Moratilla-Vega, M., Xia, H., and Page, G., (2017), ‘‘A Coupled LES-APE Approach for Jet Noise Prediction,” *46th INTER-NOISE Conference.*