# Comparison among Lattice Boltzmann and finite volume solvers for swirled confined flows

Andrea Aniello, D Schuster, P Werner, J F Boussuge, M Gatti, Clément Mirat, Laurent Selle, Thierry Schuller, Thierry Poinsot, Ulrich Rüde

# Comparison among Lattice Boltzmann and finite volume solvers for swirled confined flows

A. Aniello[a,*], D. Schuster[c], P. Werner[c], J.F. Boussuge[c], M. Gatti[b], C. Mirat[b], L. Selle[a], T. Schuller[a], T. Poinsot[a,c], U. Rüde[c,d]

[a] *Institut de Mécanique des Fluides de Toulouse, IMFT, Université de Toulouse, CNRS, France*
[b] *Laboratoire EM2C, CNRS, CentraleSupélec, Université Paris-Saclay, 3 rue Joliot Curie, Gif-sur-Yvette cedex 91192, France;*
[c] *CERFACS, 42 avenue Gaspard Coriolis, 31057 Toulouse, France*
[d] *Chair for Computer Science 10 - System simulation, Friedrich-Alexander-Universität Erlangen-Nürnberg, Cauerstraße 11, 91058 Erlangen, Germany*

**Abstract**

A finite volume and two Lattice-Boltzmann unsteady, flow solvers using LES (Large Eddy Simulation) were compared in a swirling flow configuration, typical of aeronautical combustion chambers. Numerical results were validated against experimental data collected at EM2C laboratory by comparing pressure losses, mean and RMS velocity profiles on multiple planes and axial velocity spectra. Meshes and the overall numerical setups were individually adjusted for each code to obtain the targeted accuracy before comparing CPU efficiencies.

Results confirm that the three LES codes provide high fidelity results, much better than usual RANS especially in terms of RMS data. The analysis of CPU performances shows that LBM (Lattice-Boltzmann Method) solvers are faster than the finite volume solver, even if CPU efficiencies remains of the same order of magnitude. In addition, strong scaling tests from 36 to 900 cores show that the finite volume solver scales more efficiently than the LBM codes (specially when the number of grid points per core is not sufficient).

*Keywords:* CPU efficiency, Large Eddy Simulation, LBM and FVM comparison, Swirling Flow

## 1. Introduction

The question of the efficiency of CFD solvers is an issue which has virtually disappeared for classical RANS (Reynolds Averaged Navier-Stokes) methods but remains essential when it comes to Large

*Corresponding Author
*Email address:* andrea.aniello@imft.fr (A. Aniello)

Eddy Simulation approaches [1, 2] as pointed out recently by Löhner [3]. Arguing that a certain class of CFD methods is the fastest to solve LES critical problems is a game played by multiple teams worldwide: in most cases, these discussions focus on the theoretical reasons which should make such or such approach faster than others for LES. However, the actual speed of CFD solvers for LES does not depend only on the theoretical efficiency of the method: in most cases, the mesh management, the boundary conditions, the sub models, the parallel implementation of the method also play critical roles so that the determination of an exact CPU efficiency is difficult before running real simulations. Furthermore, when simulations are run, the comparison itself between methods becomes difficult: in many cases, each author runs his own code and does not try to compare fully with competing methods. Workshops are commonly organized to compare methods but their conclusions are rarely clear because the collaboration to ensure a proper comparison remains difficult to set up (see for example [4] for such a workshop on compressible methods for LES of turbomachinery).

This paper follows a different path as it compares three LES solvers (Table 1) which were all run by the same group of people at IMFT and CERFACS: an incompressible finite volume solver: $\text{AVBP}_{\text{pgs}}$ and two Lattice Boltzmann (LB) codes: ProLB and waLBerla. All computations were performed on the same machines, with the same number of processors and a systematic comparison was organized to ensure a fair evaluation of methods. The target configuration was the internal turbulent flow in a swirling burner but all cases correspond to a non reacting situation. Unlike many previous studies, the present one focuses on an internal flow, at moderate Reynolds numbers, as found in combustion chambers and not on external flows as found in aerodynamic and aeracoustic studies. Experiments were performed by the EM2C laboratory and include enough detailed data to evaluate the precision of the solvers in terms of pressure losses and full mean and RMS velocity fields.

| **Code** | **Formulation** | **Method** |
|---|---|---|
| $\text{AVBP}_{\text{pgs}}$ | Incompressible | Finite volume (Galerkin) |
| ProLB | Athermal | Lattice Boltzmann |
| waLBerla | Athermal | Lattice Boltzmann |

Table 1: Presentation of solvers used for LES simulations

Of course, the first difficulty in such an exercise is the definition of the rules of the game. In the present case they can be explained as follows: "for each code, build a numerical setup which provides a minimal accuracy in terms of flow field resolution (pressure loss as well as mean and RMS velocity profiles within experimental accuracy levels) and compare the CPU efficiency". Since multiple solvers are used, with different meshes, different algorithms and different submodels, the notion of minimal accuracy remains arbitrary: here, long discussions between CFD and experimental teams members have lead to a minimum quality which was expected from the match between experimental and LES results in terms of average and RMS velocity fields for multiple locations. This was used for the three solvers to determine the minimum grid size required to reach this sufficient level of agreement. Once all solvers were found to provide comparable agreement with the experimental data (taking into account experimental uncertainties), CPU efficiencies were determined. This procedure still contains a clear level of arbitrariness that authors do not want to deny. The error bars expected on the results are certainly significant and of the order of tens of percents. However, the main issue here is not to determine if a method is 50 percent faster than another one: we are interested in orders of magnitudes as required for example for industry to move from one class of methods to another one.

The paper is organized as follows: the experimental configuration is described first in Section 2. The three solvers and the corresponding numerical setups are described next (Section 3) before discussing results in Section 4 in terms of velocity profiles (mean, RMS and spectra). Finally the computational efficiency of the three codes is discussed in Section 5.

## 2. Experimental setup

The configuration (Fig. 1) was designed to analyze the response of swirled flames to flow rate modulations in [5, 6].

Dry air is injected from two diametrically opposed apertures at the bottom of a plenum. The flow crosses a grid and a honeycomb to break the largest turbulent scales. A convergent section produces a top-hat laminar velocity profile with a boundary layer of about 1 mm thickness that was characterized by a hot wire probe (Dantec Dynamics - Probe 55P16 with a mini-CTA 54T30). The diameter of this section is $D = 22$ mm and the bulk flow velocity is fixed to $u_b = 5.44 \pm 0.05$ m s$^{-1}$ corresponding to a Reynolds number $\mathrm{Re}_D = 7\,620$ at room temperature $T = 300$ K. The pressure drop with respect to ambient conditions is recorded in front of the hot wire probe with a differential

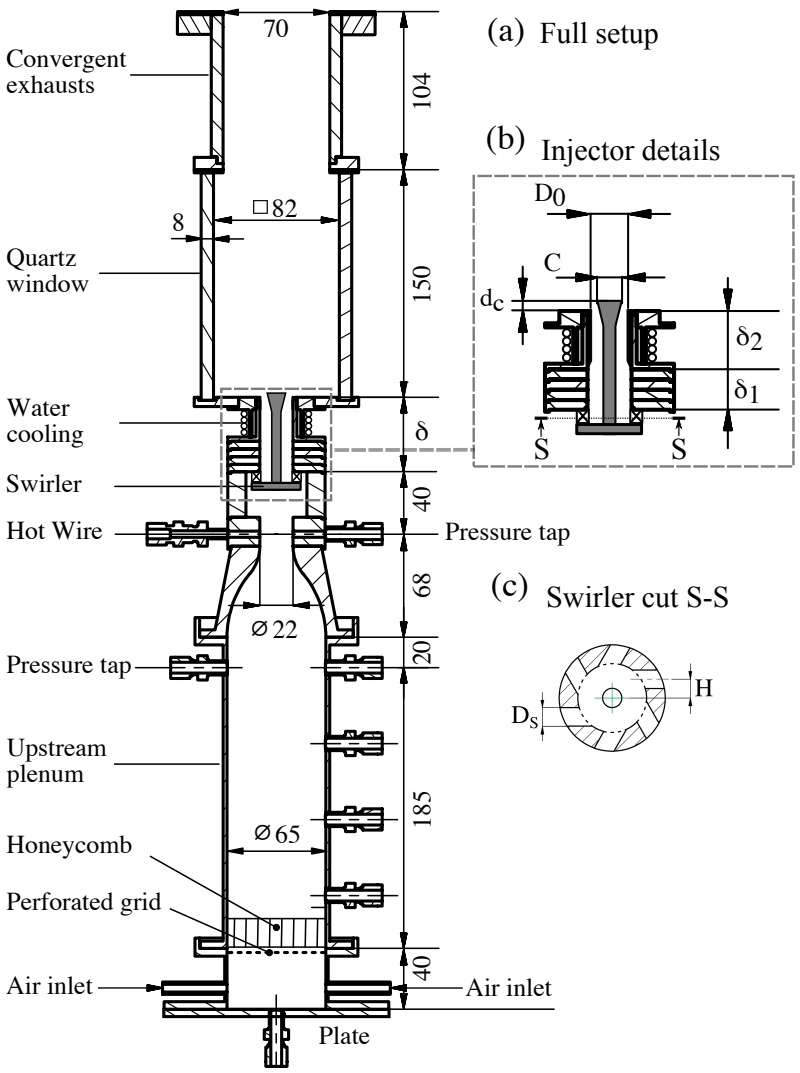


Figure 1: Experimental setup with detailed representation of the injector region and swirler geometry. All dimensions in mm.

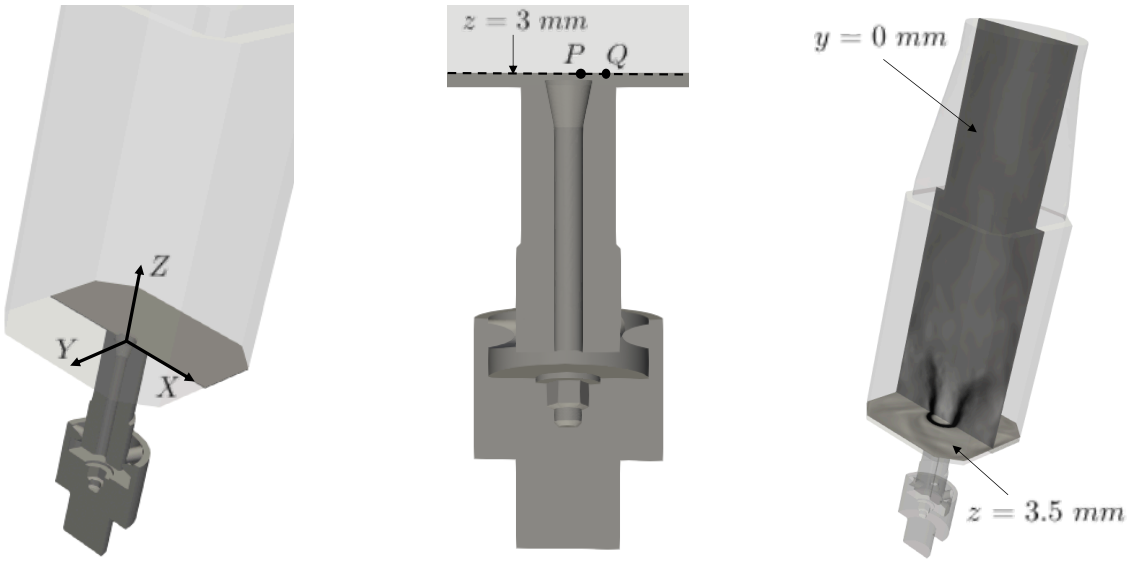


Figure 2: Presentation of cartesian system of reference (left). Identification of points $P$ and $Q$ used for modal analysis (center) and representation of PIV transverse and axial planes of investigation (right)

pressure gauge and indicates $\Delta p = 335 \pm 15$ Pa. The setup was originally designed to analyze effects of geometrical modifications of the injector on flame dynamics and the burner replaceable components are represented in Fig. 1. In the configuration explored, the radial swirling vane consists of $n = 6$ cylindrical tangential inlets of diameter $D_s = 6$ mm with an offset $H = 6$ mm as indicated in Fig. 1c. The flow leaves the swirler into a central injection tube. The diameter of this tube is $D = 22$ mm, over a first section of length $\delta_1 = 16$ mm, followed by a central insert of length $\delta_2 = 34$ mm and diameter $D_0 = 20$ mm. A central rod of diameter $d = 6$ mm ending with a cone of diameter at the top $C = 10$ mm is inserted in the injection tube. The distance between the swirler exit and the combustion chamber back plane is here fixed at $\delta = \delta_1 + \delta_2 = 50$ mm. The central rod protrudes inside the combustion chamber and the distance between the top of the cone and the backplane is 1.5 mm. The combustion chamber, made of 4 transparent quartz windows, has a 82 mm squared cross-section and length 150 mm. It is extended by a nozzle with a square inlet section and a circular outlet section of diameter 70 mm. Transition between this square to circular sections is made over a 104 mm length. This device ensures that there is no reverse flow at the setup outlet. The cartesian system of reference used through the paper is presented in Fig. 2: $z$ corresponds to the symmetry axis of the injector, while the plane defined by axes $x$ and

$y$ corresponds to the backplane of the combustion chamber ($z = 0$ mm).

Particle Image Velocimetry (PIV) is used to measure the cold flow velocity fields. For these measurements, the flow is seeded with small oil droplets of diameter 1-3 $\mu$m. PIV data are gathered on both axial and transverse planes within the combustion chamber. The first is identified by $z$ and $x$ directions ($y = 0$ mm), while the second one is parallel to the burner backplane and located at $z = 3.5$ mm. No PIV data is available at lower axial coordinates. The PIV system consists of $2\times400$ mJ Nd:YAG laser doubled at 532 nm operated at 10 Hz and a $2048\times2048$ pixels CCD camera (Dantec Dynamics, FlowSense EO 4M). Two different optical setups are used with a time interval between the two laser pulses $\Delta t = 10$ $\mu$s and a pixel pitch of 27.88 pixels mm$^{-1}$ for measurements in the axial plane and $\Delta t = 25$ $\mu$s and a pixel pitch of 40.14 pixels mm$^{-1}$ for measurements in a transverse plane. Eight hundred images are taken to assure the convergence of the mean and RMS values of the velocity field, which is computed from the cross-correlation of the PIV images by a three passes window deformation technique (from $64 \times 64$ pixels to $16 \times 16$ pixels interrogation areas), with an uncertainty of 0.1 pixels on the calculated displacement.

The measurements are completed by Laser Doppler Velocimetry (LDV) to capture the time resolved velocity profiles at $z = 3$ mm above the top central insert (see Fig. 2). The Power Spectral Density (PSD) of the axial velocity is calculated to detect the presence of potential hydrodynamic or acoustic modes. To ease optical access the combustion chamber is removed for those latter measurements and LDV data are collected in unconfined configurations. Two laser beams at $\lambda = 514.5$ nm (green) allow to probe the axial velocity. Two other beams at $\lambda = 488$ nm (blue) are used to measure the velocity component along $x$ direction. The data collection rate is always greater than 10000 s$^{-1}$ and for each measurement point at least 250000 particles are considered, in order to obtain fully converged mean and RMS values for all components of velocity. The statistical bias is corrected by the transit time of each particle. The analysis of time traces and PSD of these signals does not reveal any specific coherent structures associated to helical flow instabilities.

## 3. Presentation of solvers

### *3.1. High-order finite volume solver*

AVBP is a multi-species LES explicit solver for Navier-Stokes compressible equations developed at CERFACS (www.cerfacs.fr/avbp7x). This Cell-Vertex (CV) high-order Finite Volume (FV)

code [7, 8] is able to handle structured, unstructured and hybrid grids in both two and three space dimensions. It is a world standard code to compute turbulent reacting flows in combustion chambers [9] or explosions in confined domains [10]. A critical aspect of compressible codes is the treatment of numerical boundary conditions where acoustic reflections must be controlled to avoid spurious phenomena. In order to fulfill those requirements, AVBP exploits Navier-Stokes Characteristic Boundary Conditions (NSCBC) [11, 12, 13, 14].

Like any explicit compressible code, AVBP tends to be less efficient for low Mach number simulations. The problem arises from the large disparity between time scales associated to sound waves propagation and convection: the CFL stability condition imposed by the sound speed is uselessly severe with respect to the limit established by convection alone, resulting in an unnecessarily small time step. In order to overcome this limitation and to be representative of another class of FV solvers, which use incompressible or low-Mach number formulations, a modified version of AVBP, called here $\text{AVBP}_{\text{pgs}}$ is used to remove the acoustic time step limitation: the governing equations solved are manipulated according to the Pressure Gradient Scaling (PGS) technique [15]. PGS rescales the pressure gradient in the momentum equations to reduce the computational sound speed, so that the time step is not limited by the true sound speed which is irrelevant. The PGS methodology is limited to low-speed incompressible flows like the present configuration. Overall, the procedure is equivalent to the $\alpha$-transformation developed in [16], but with the advantage of both easier implementation and wider applicability. Since the Lattice Boltzmann solvers are used here in their athermal weakly compressible form, it seems reasonable to utilise $\text{AVBP}_{\text{pgs}}$ which uses similar assumptions.

### *3.2. Lattice-Boltzmann solvers*

#### *3.2.1. General description of the method*

The LBM considers the dynamic evolution of a mass distribution function of particles $f(t, \mathbf{x}, \xi)$ that collide and propagate at time $t$, position $\mathbf{x}$ on a discrete velocity stencil called Lattice, commonly noted D$d$Q$q$ ($d$ for spatial dimension and $q$ for velocities). In this study, two solvers were compared, ProLB and waLBerla. Both employ a D3Q19 lattice given by:

$$\xi_i = \begin{cases} (0,0,0), & i = 0 \\ (\pm 1,0,0),(0,\pm 1,0),(0,0,\pm 1), & i = 1-6 \\ (\pm 1,\pm 1,0),(\pm 1,0,\pm 1),(0,\pm 1,\pm 1), & i = 7-18. \end{cases} \tag{1}$$

At a mesoscopic scale, each function $f_i = f(t, \mathbf{x}, \xi_\mathbf{i})$ is governed by the Lattice-Boltzmann scheme with the time step $\Delta t$ and space step $\mathbf{\Delta x} = \xi_\mathbf{i} \Delta t$:

$$f_i(\mathbf{x} + \xi_\mathbf{i} \Delta t, t + \Delta t) - f_i(\mathbf{x}, t) = \Delta t \Omega_i(\mathbf{x}, t), \tag{2}$$

where $\Omega_i$ is the collision operator. Modeling the collision component circumvents considering its non-linear behavior and therefore, makes this approach particularly well suited for parallel simulations of athermal weakly-compressible flows. The purpose is to preserve the main characteristics of the collision operator $\Omega_i$ such as the convergence through a local thermodynamic equilibrium during a relaxation time $\tau$ and the conservation of mass and momentum.

While the macroscopic quantities (density $\rho$ and momentum $\rho \mathbf{u}$) are retrieved from the velocity moments of the distribution function given by:

$$\rho = \sum_{i=0}^{q} f_i, \quad \rho \mathbf{u} = \sum_{i=0}^{q} f_i \xi_i, \tag{3}$$

the pressure $p$ is computed from the barotropic equation of state, $p = \rho R T_0$ where $R$ is the gas constant and $T_0$ is a characteristic temperature, instead of the Poisson equation involved in most Navier-Stokes solvers allowing to considerably reduce the computational costs.

Finally, the Lattice-Boltzmann equation is discretized on cartesian cubic grids automatically generated in most solvers which alleviates cumbersome and time-demanding meshing. Both solvers are based on this description but use different numerical implementations.

| Code | ProLB | waLBerla |
|---|---|---|
| Number of nodes [M] | 26.3 | 39.3 |
| Number of fluid nodes [M] | 26.3 | 30.2 |
| Octree level distribution fluid nodes [%] | [67, 25, 5, 3, 0, 0] | [59, 24, 13, 1.9, 1.5] |
| Equivalent Fine Nodes [M] | 21.4 | 29.8 |
| Equivalent Fluid Fine Nodes [M] | 21.4 | 22.6 |

Table 2: Level distribution of fluid nodes in LBM solvers

A key difference between FV and LBM algorithms is that the FV solver uses the same time step in the whole computational domain, whereas in LBM the time step depends on the level of refinement. A cell on the coarser grid level has twice the size but also the time step is two times bigger compared

to the finer level. To take this effect into account we generally use the term Equivalent Fine Nodes (EFN) which counts cells on coarser levels $L$ as a fraction of the finest level (Eq.4). This quantity reflects the workload equivalent of a mesh using only the minimal mesh size $\min(\Delta x)$ and is therefore more suitable for comparison between the codes.

$$\mathrm{EFN} = \sum_{L} N_{\text{nodes on L}} \left(\frac{1}{2}\right)^{L-1} \tag{4}$$

Another consequence is that the main part of the workload is generated by the finest level in the domain. In order to do a fast computation it is essential to minimize the regions with minimal cell size. The differences between the two LBM meshes are summarized in table 2. The ProLB algorithm typically needs fewer EFN to accurately resolve a given problem, whereas the approach of waLBerla generates a non negligible amount of excess cells, that are not part of the fluid domain but have to be computed anyway.

### *3.2.2. ProLB*

ProLB is a commercial suite of tools which emerged from LaBS and CLIMB [17] French research projects carried out by a consortium of industrial companies, universities, research laboratories and institutes. Its inherent massively-parallel solver includes an octree mesher which efficiently handles both complex geometries [18] and multi-resolution refinement layers [19].

For the present study, a modified version of the commercially available ProLB software is used. The numerical resolution of the Lattice-Boltzmann equation is performed through the D3Q19 lattice with a hybrid version [20] of the recursive regularized collision operator [21] (H-RR). It shows superior stability properties than the classical Bhatnager-Gross-Krook (BGK) [22] for high-turbulent flows [23] by filtering out the spurious and non-hydrodynamic modes that could be amplified at grid transitions [24].

To handle mesh refinement, a Direct-Coupling (DC) algorithm is employed [25]. By ensuring mass and momentum conservation at the transition nodes, a singular equilibrium distribution function is computed to recover the missing distributions at both coarse and fine sides. This combination of the H-RR collision model and the DC algorithm offers better accuracy and locality in complex configurations than the classical overlapping method [26].

The boundary nodes need a specific treatment in LBM approaches: since the mesh is completely cartesian, an immersed boundary condition is implemented to handle the solid walls [27] [28]

allowing first the automatic generation of the mesh and second to flag the interface nodes. It is then supplemented by a Grad's moment approximation of the missing populations to recover macroscopic quantities at the interface. This yields a more stable and accurate approach than the well-known interpolated bounce-back [29].

Thanks to the octree multi-resolution mesher, the ultimate grid is built upon a static adaptive refinement strategy [30] where the considered sensor is the dissipation of kinetic energy [31]. From an initial coarse simulation, the time-averaged field of this sensor is computed. Therefore, a smoothed iso-volume based on a lower case-dependent threshold of the sensor yields a finer resolution domain which is directly reintroduced in the octree mesher. Thereby, this process is repeated twice in order to predict pressure losses and optimize the number of fluid nodes by refining only the relevant areas and minimizing computational costs.

#### *3.2.3.* waLBerla

waLBerla [32] is an extreme scale, open-source, C++ multiphysics software framework. It can be used as a tool box for designing various types of applications such as the LBM computation performed here [33].

waLBerla was designed from the ground up for high-performance computing (HPC) on massively parallel clusters [34, 35] and GPU-based systems [36], so that it is used as a reference implementation for LBM performance studies [37].

waLBerla uses automatic code generation [38] to ensure excellent execution performance on a wide range of different architectures. This meta-programming paradigm allows to start the application development from a high-level description of the LBM method. All steps to derive the LBM kernel codes can be performed automatically: the code is not only optimized for specific architectures, but also becomes easier to change to test variants of the LBM methods.

The framework is based on a block-structured domain partitioning in order to achieve extreme scalability and node level performance [39, 40]. The full domain is divided into equally sized cuboids that can only be refined as a whole at desired zones and at a size ratio of 2:1 with direct neighbors. These subdomains are called *blocks* and all hold the same number of grid cells. The computational domain partitioning is partitioned in such blocks that can be assigned to processes. Load balancing is achieved on the level of blocks, not individual cells.

Every process can hold several blocks, but a block can only be assigned to one single process. Data

from blocks is only available to the block that it has been assigned to. This structure allows code parallelization by the Message Passing Interface (MPI) or using hybrid MPI/OpenMPI to guarantee optimal scalability on a wide range of different supercomputer architectures. In complex geometries, the meshing algorithm will loop over all blocks and all cells to determine if they are inside or outside the surface mesh and accordingly set them as fluid or empty cells. Blocks that hold no fluid cells can be discarded, but blocks that hold one or more fluid cells will be kept and stocked with the same set of data. This is necessary because the LBM kernels iterate over over all cells equally, independent if they are fluid or not. Code generation is handled by the *pystencils* package [41, 38]. It uses symbolic manipulation with the *SymPy* algebra system to derive symbolically a stencil formulation from the continuous LBM collision operator. During this procedure several optimization techniques, such as common subexpression elimination and vectorization can be applied to generate highly efficient C/C++ code. Additionally, optimized code for GPUs can be generated. Thus waLBerla with pystencils can achieve performance portabiliy to a wide range of different architectures, including CPUs and GPUs.

### *3.3. Numerical setups*

The three solvers were applied to the same swirler geometry of EM2C but they employ different meshes (Table 3): $\text{AVBP}_{\text{pgs}}$ uses body-fitted unstructured tetrahedral mesh with 18.1 M cells in total, while ProLB relies on cartesian unstructured mesh, offering a local refinement possibility but requiring up to 26.3 M grid elements, which amounts 21.4 M EFN to represent the same geometry. This tendency is exacerbated in waLBerla, where the structured cartesian mesh is only able to refine whole blocks of the mesh, which leads to 39.3 M, or 22.6 M EFN cells overall.

| Code | Topology | Cell type | $\min(\Delta x)$ [$\mu$m] | Total elements [M] |
|---|---|---|---|---|
| $\text{AVBP}_{\text{pgs}}$ | Unstructured | Tetrahedral | 80 | 18.1 |
| ProLB | Unstructured | Cubic | 110 | 26.3 (21.4*) |
| waLBerla | Structured | Cubic | 110 | 39.3 (22.6*) |

Table 3: Overall description of mesh parameters for the three solvers (** marks the number of EFN)*

A detailed representation of each grid, including swirler, injector and near-backplane region, is displayed in Fig. 3. It exhibits an axial cut for each code to show the local mesh structure: the swirler region has the highest resolution to predict the correct velocity field and pressure losses.

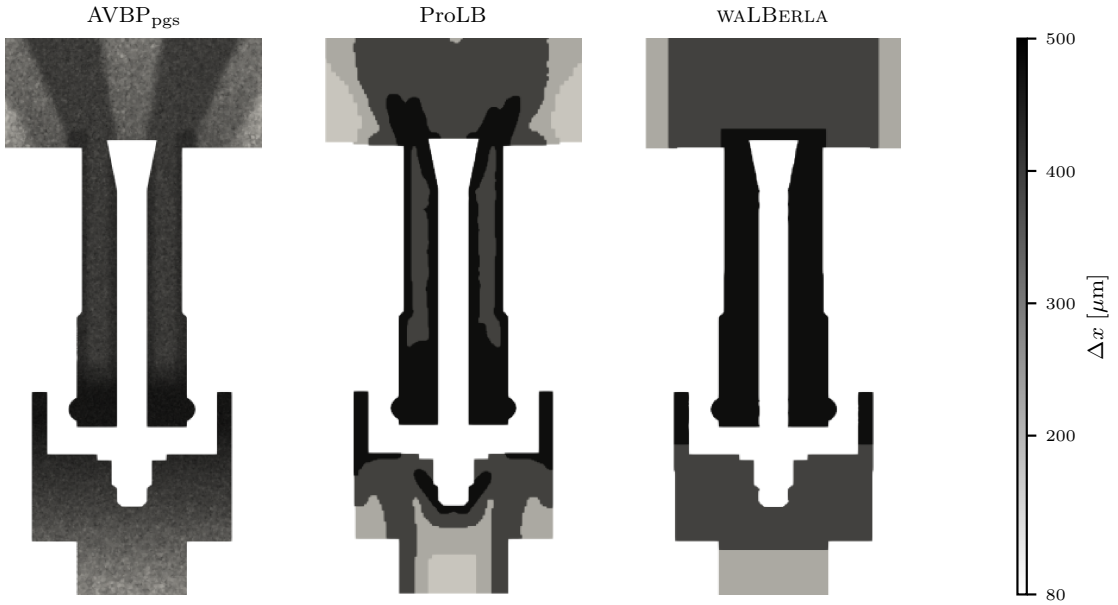


Figure 3: Mesh comparisons via $\Delta x$ contour maps for the three solvers: AVBP$_{pgs}$, ProLB, waLBerla. Differences among structured and unstructured grids, as well as between cubic and tetrahedral elements are highlighted

In this region AVBP$_{pgs}$ adopts a minimum cell parameter ($\Delta x$) of 80 $\mu$m that increases along $z$ up to the burner backplane with an average $\Delta x \approx 180$ $\mu$m. While ProLB mesh is refined through adaptive unstructured blocks of minimal mesh size $\Delta x = 110$ $\mu$m, waLBerla uses a block of constant $\Delta x = 110$ $\mu$m in both swirler and injector. With these meshes, the time steps are adjusted to obtain CFL numbers based on the maximum convective velocity of the order of 0.1 for all codes. AVBP$_{pgs}$ adopts Lax-Wendroff scheme [42], second-order in both space and time. The CFL number (based on the modified sound speed) is set to 0.9. Temperature and volumetric flow rate are fixed at the inlet, while ambient pressure (101325 Pa) with a proper relaxation coefficient is imposed at the outlet. The SIGMA model is used for subgrid Reynolds stresses [43]. Both turbulent Prandtl number Pr and Schmidt number Sc are fixed to 0.6 and only one inert species representative of air is computed. The PGS parameters are set to obtain a maximum computational Mach number of 0.3.

While ProLB employs a H-RR collision operator combined with a DC mesh transition algorithm, waLBerla is using a classical BGK collision model. Both solvers impose a one-seventh power law

velocity profile at the inlet to match the experimental flow rate and a constant pressure of 1 bar at an extended outlet overlaid by a sponge layer to dump non-hydrodynamic reflection waves inside the domain. Walls are treated differently: ProLB uses a Grad's approximation and waLBerla a bounce-back scheme to treat the "missing populations".

Since acoustics do not contribute significantly to the flow behavior, the non-dimensionalized Newtonian sound speed $c_s^*$ is artificially minimized by being cautious that the maximal Mach Number does not exceed the critical value of 0.4 [44]:

$$\mathrm{Ma}_{\max} = \frac{u_{\max}}{c_s^*}\frac{\Delta t}{\Delta x} < 0.4. \tag{5}$$

This requirement is similar to the CFL condition for classical Navier-Stokes numerical schemes [45]. This process intends to increase the time step to its maximal value and therefore allows to lower the computational time while the accuracy is still conserved.

LES in ProLB are performed using a Shear-Improved Smagorinsky turbulence model (LES-SISM) [46] which accounts for the dissipation of the unresolved turbulent scales through an eddy viscosity $\nu_{sgs}$. This subgrid-scale eddy viscosity is introduced by locally shifting the relaxation time $\tau^* = \tau + \tau_{sgs}$ in the collision operator in order to model the additional convection of momentum and energy through the subgrid turbulent eddies. To enclosure this model, $\nu_{sgs}$ is computed via the strain-rate tensor.

waLBerla on the other hand employs a basic version of the Smagorinsky model $\tau_{sgs} = (C_S\Delta)^2|S|$, in which the turbulent viscosity depends on the local strain rate tensor $S$, the Smagorinsky constant $C_S$ and the filter length $\Delta = 1$ in lattice units.

## 4. Comparison with experimental data

This section presents a comparison of experimental and numerical results to validate the entire set of simulations. All three codes are used on sufficiently long times to obtain fully converged averaged solutions which are time-independent in terms of mean and RMS results: AVBP$_{\mathrm{pgs}}$ run for 300 ms of physical time, while ProLB and waLBerla both run for 120 ms. A physical time of 120 ms corresponds to approximately 10 flow-through times ($\tau_t$), where $\tau_t = 10$ ms is obtained by the time taken for a flow element to travel through the chamber in the high velocity zone at 8 m s$^{-1}$.

The first quality indicator for swirler flows is the pressure loss $\Delta p$ through the swirler (Table 4) which controls its performances in a real engine. $\Delta p$ is calculated between the pressure tap location

(see Fig. 2) and the outside ambient pressure.

| | $\Delta p$ (Pa) |
|---|---|
| Experiment | 335±15 |
| $\text{AVBP}_{\text{pgs}}$ | 330 |
| ProLB | 368 |
| waLBerla | 313 |

Table 4: Injector head pressure losses due to swirler

The agreement of $\text{AVBP}_{\text{pgs}}$ with the experiment is slightly better than for LBM codes, as expected for a code which uses body-fitting meshes. ProLB and waLBerla predict a $\Delta p$ of 368 Pa and 313 Pa, respectively above and below the experimental confidence interval. To qualify codes, measuring the pressure loss is not sufficient, however, and the next sections focus on a detailed analysis of the velocity fields.

### *4.1. PIV*

Velocity profiles are compared on two planes: the axial $zx$ plane and the transverse plane $z = 3.5$ mm (see Fig. 2). In the first case data are displayed over a $40{\times}50$ mm$^2$ rectangular window symmetrically located with respect to the $z$ axis. For the second, results are presented over a $40 \times 40$ mm$^2$ squared area centered in the middle of the combustion chamber with sides oriented along $x$ and $y$ directions. Furthermore local one-dimensional, velocity profiles extracted at constant $z$ are also retrieved from both PIV data and simulations.

#### *4.1.1. Axial plane*

Fig. 4 exhibits two rows of images: the top row shows the mean axial velocity component $\bar{u}_z$. The second one shows its RMS noted $u_{z,\text{RMS}}$. From left to right experimental data and numerical results are displayed, as specifically reported on top of each plot.

The experimental mean velocity contour map of Fig. 4 highlights a large Inner Recirculation Zone (IRZ), typical of high swirling flows. This region of negative axial velocity is created by the vortex breakdown and is delimited by iso-velocity lines at $\bar{u}_z = 0$ m s$^{-1}$. Moreover two high velocity branches develop in the wake of the injector annular channel, identified by isolines at 8 m s$^{-1}$.

The comparison with numerical results show that the three solvers properly capture the flow characteristics, for both width and intensity of the IRZ. Minor differences can be highlighted:

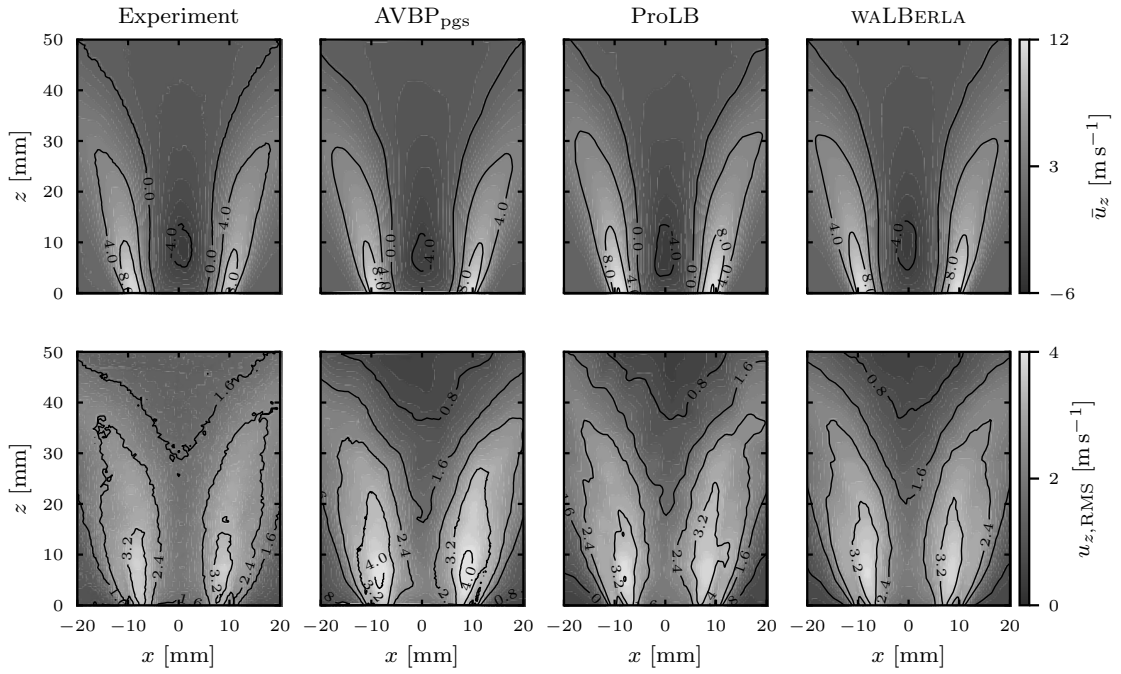


Figure 4: Contour map comparison of experimental and numerical results of the axial ($z$) component of the velocity on the axial plane. First row exhibits the mean velocity field $\bar{u}_z$, while second row displays the RMS distributions $u_{z,\mathrm{RMS}}$

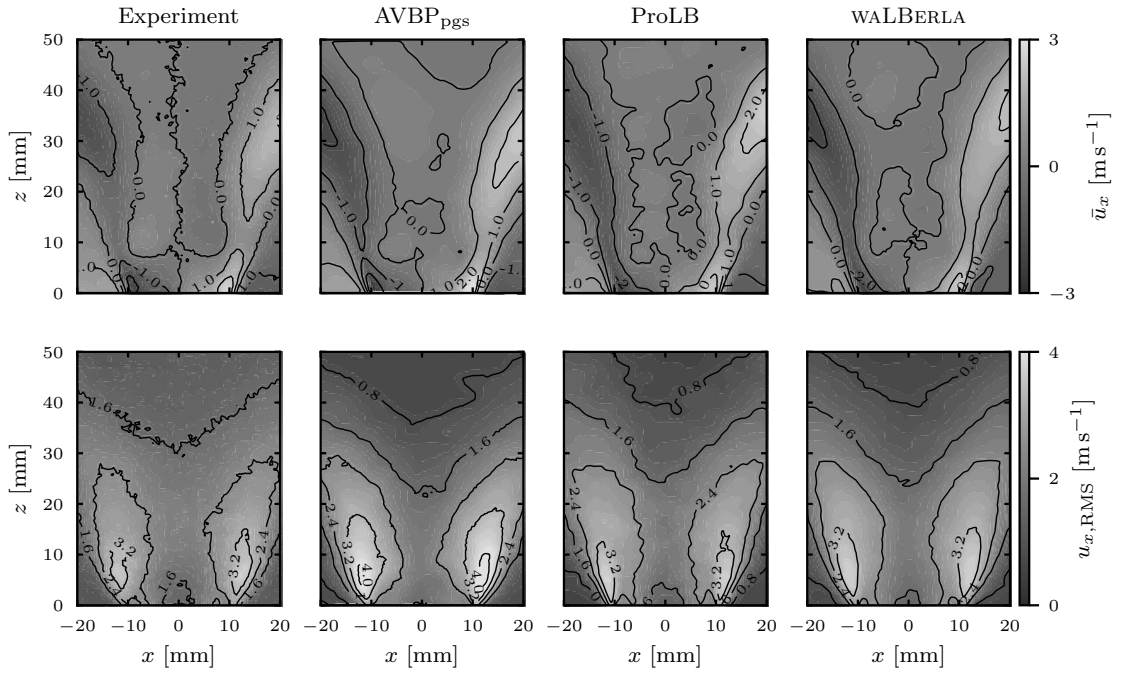


Figure 5: Contour map comparison of experimental and numerical results of the horizontal ($x$) component of the velocity on the axial plane. First row exhibits the mean velocity field $\bar{u}_x$, while second row displays the RMS distributions $u_{r,\mathrm{RMS}}$

the extension of the high mean velocity regions is slightly overestimated by LBM codes, where 8 $\mathrm{m\,s^{-1}}$ and 4 $\mathrm{m\,s^{-1}}$ isolines develop further downstream than in experimental data. In addition, RMS maps obtained with the three solvers highlight less regular contour plots, potentially linked to averaging times which are much smaller in the LES than in the experiment. $\mathrm{AVBP_{pgs}}$ shows slightly higher RMS with respect to other codes and experiment, especially in the high velocity region.

The horizontal component ($x$) results (Fig. 5) confirm the axial component conclusions of Fig. 4: the three solvers are able to match the experimental results and all solvers provide results which are within the experimental range of precision for PIV results (typically 0.3 m/s here). However Figures 4 and 5 are not sufficient to provide an appropriate evaluation of the precision: it is worth considering one-dimensional profiles showing local $\bar{u}_z$ and $\bar{u}_x$ velocity profiles sectioning the axial plane at specific $z$ locations.

Fig. 6 displays mean and RMS profiles of $\bar{u}_z$ and $\bar{u}_x$, where in the first row profiles are extracted at $z = 5$ mm and in the second one at $z = 15$ mm.

First, the $\bar{u}_z$ profile reveals that experimental data are not exactly symmetric with respect to the $z$ axis: the left-hand velocity peak is higher than the other, while simulations do not show the same differences. Numerical predictions show that at both $z$ coordinates, the slope and the minimum of the axial velocity profile, which define the IRZ structure, are correctly represented. On the other hand the maximum axial velocity values at $z = 15$ mm are slightly overestimated by LBM codes by roughly 0.5 $\mathrm{m\,s^{-1}}$, which corroborates the little discrepancy highlighted in Fig. 4. $\bar{u}_x$ profiles are generally hard to match in swirled flows since the mean velocity intensity is comparable to its RMS values. This emphasizes the fidelity of all simulations put in place. In fact the maximum difference between experimental data and simulations is only of the order of few tenths of meters per second. Furthermore, the RMS profiles for both $\bar{u}_z$ and $\bar{u}_x$ at both $z$ coordinates confirm the $\mathrm{AVBP_{pgs}}$ little overestimation at peak velocity locations, which in any case remains largely within the experimental accuracy margin.

Another way to accurately compare the codes is to compute the $L^2$-norm relative errors between the experimental and simulation values present in Fig. 4. These are given in the Tab. 5.

*4.1.2. Transverse plane*

Fig. 7 displays $\bar{u}_x$ and $\bar{u}_y$ mean velocity profiles on the transverse plane $z = 3.5$ mm.

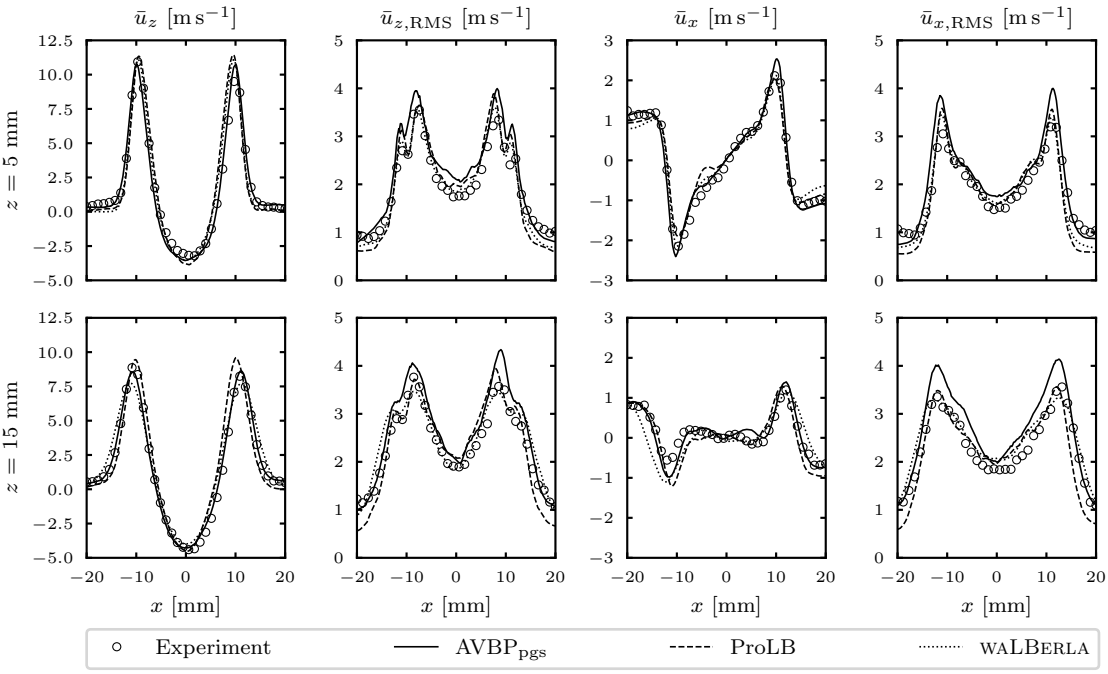


Figure 6: Mean velocity profiles of $\bar{u}_z$ and $\bar{u}_x$ with related RMS at constant axial coordinate on the axial plane. First and second rows correspond to $z = 5$ mm and $z = 15$ mm, respectively

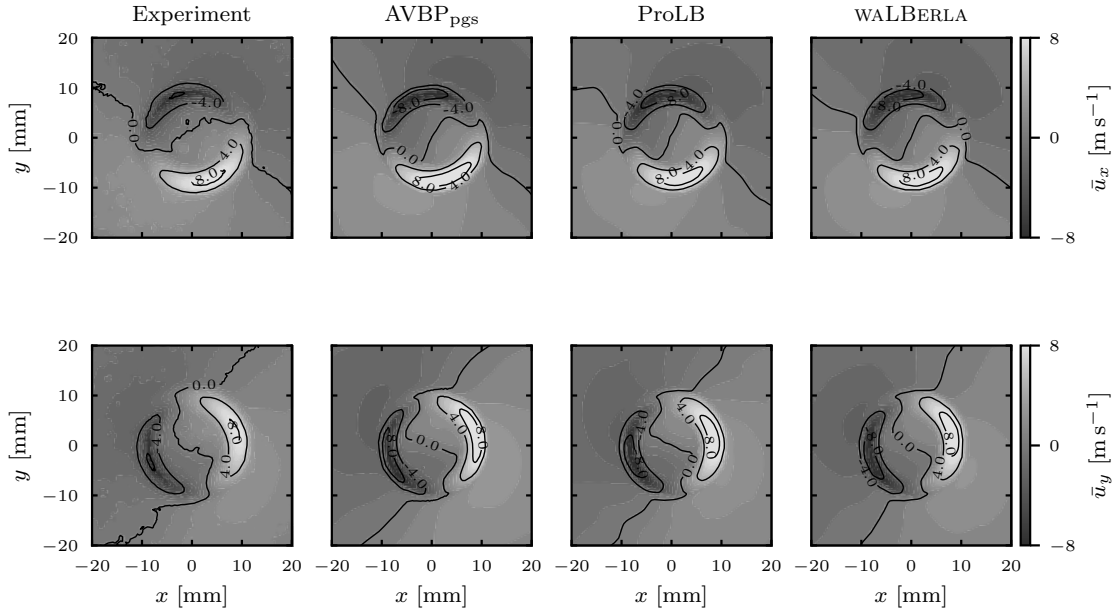


Figure 7: Contour map comparison among experimental and numerical results on transverse plane. The first row exhibits mean velocity component $\bar{u}_x$, while second row displays the mean velocity component $\bar{u}_y$

| Code | $z = 5$mm | | | | $z = 15$mm | | | | $\langle\eta\rangle$ |
|---|---|---|---|---|---|---|---|---|---|
| | $\bar{u}_z$ | $\bar{u}_{z,\mathrm{RMS}}$ | $\bar{u}_x$ | $\bar{u}_{x,\mathrm{RMS}}$ | $\bar{u}_z$ | $\bar{u}_{z,\mathrm{RMS}}$ | $\bar{u}_x$ | $\bar{u}_{x,\mathrm{RMS}}$ | |
| $\mathrm{AVBP}_{\mathrm{pgs}}$ | **0.13** | *0.16* | **0.16** | *0.17* | **0.08** | 0.14 | **0.39** | *0.17* | 0.17 |
| ProLB | *0.26* | 0.15 | *0.21* | 0.16 | *0.26* | *0.14* | 0.6 | 0.15 | 0.24 |
| waLBerla | 0.20 | **0.10** | 0.19 | **0.12** | 0.19 | **0.11** | *0.74* | **0.10** | 0.22 |

Table 5: $L^2$-norm relative error for each solver and each plot of the Fig. 4 respectively. To give an overall comparison, the global error $\langle\eta\rangle$ has been computed as the average of each plot error. Bold and emphasize errors shows the more and less accurate result respectively.

The good agreement between computational and experimental velocity fields confirms the previous findings. Only a slight overestimation of absolute maximum and minimum for both $\bar{u}_x$ and $\bar{u}_y$ can be pointed out looking at iso-velocity lines of $\pm 8\ \mathrm{m\,s^{-1}}$.

Furthermore the two high/low velocity symmetric patches result slightly counterclockwise-rotated due to the square shape of the combustion chamber. Remarkably, the flow field is correctly captured by numerical computations.

### *4.2. Power Spectral Density (PSD) of axial velocity*

In addition to the mean and RMS values, it is also interesting to look at axial velocity spectra and compute Power Spectral Density (PSD) from local time signals: PSD are obtained experimentally from LDV velocity signals acquired along the $x$ axis: from $x = -15$ mm to $x = 15$ mm with 0.5 mm step. It is worth mentioning that in contrast with simulations, measurements have been performed without combustion chamber. However it has been verified that experimental mean and RMS axial velocity are comparable with both PIV data the numerical predictions obtained in confined configuration. This feature is attributed to the weak impact of the confinement on the flow structure close to the injector outlet, at $z = 3$ mm.

Simulations and experimental signals are extracted for axial velocity over a span of 120 ms in order to share the same numerical frequency resolution of 8.3 Hz. Even though the upper-limiting frequency of the spectra is mathematically fixed by the sampling frequency, this limit could be misleading since oil particles used to seed the flow act like a low-pass filter, not responding to high perturbation frequencies. In the present case, the cut-off frequency of the small oil particles is of the order of 4 kHz: above this value, experimental spectra can not be physically considered.

Fig. 8 displays the PSD at two locations (values in $mm$): $P = (3,\ 0,\ 3)$ is located in the wake of the central bluff-body and $Q = (7,\ 0,\ 3)$ lies in the shear layer of the swirling jet (Fig. 2). The gray scale marks the fact that at high frequencies only numerical results can be interpreted.

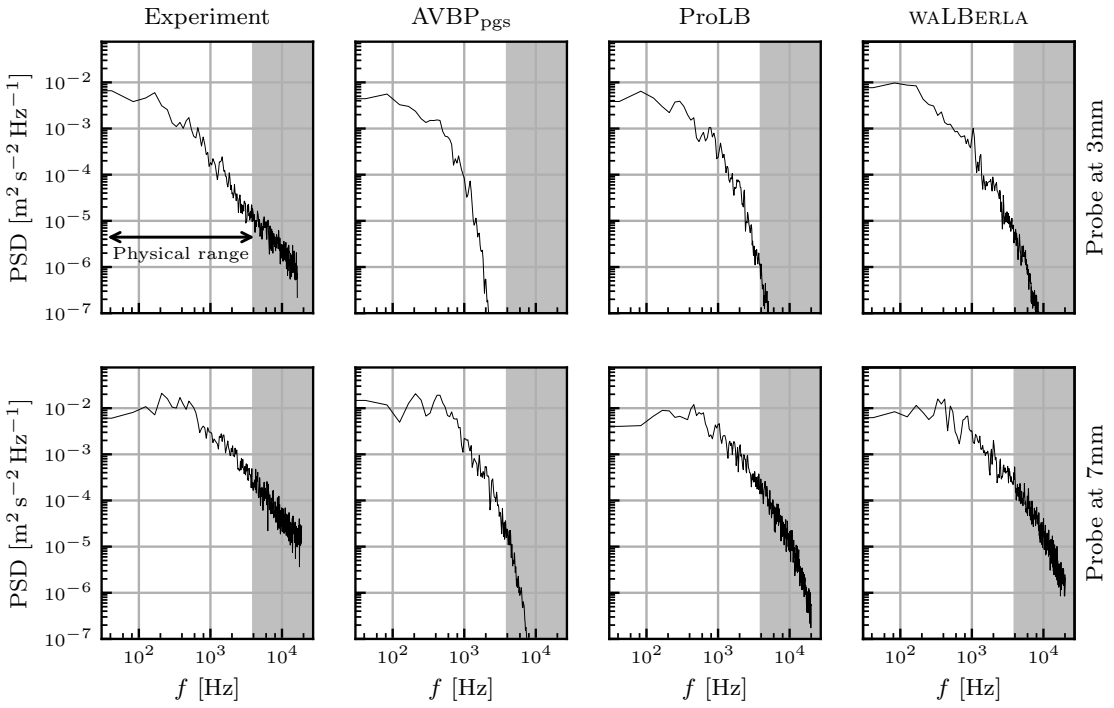


Figure 8: Axial velocity spectra of kinetic energy for the different solvers at $P = (3,\ 0,\ 3)$ (top row) and $Q = (7,\ 0,\ 3)$ (bottom row)

For the present flow rate, experimental PSD results (left column in Fig. 8) do not exhibit peaks associated to coherent structures such as Precessing Vortex Cores (PVC) which are frequently found in swirling flows [47, 48, 49, 50]. Similarly all three simulations do not reveal any peak related to hydrodynamic modes. The PSD decay above 1 kHz shows that calculations exhibit a higher dissipation with respect to experimental results, maybe due to the LES subgrid model used in the three codes. Limited differences are observed among codes at point $P$. For point $Q$, a difference emerges in the high frequency range: the two LBM codes introduce less dissipation than the finite volume solver but it is difficult to say if this is physically right or not.

## 5. Comparison of computational costs

The code performances are given by Table 6. A first parameter, which is independent of mesh size and time step is the reduced computational time, i.e. the time required to perform one cell update. The last line of Table 6 displays a second parameter which is the most important one for the user: the total CPU time required by each code to compute 1 ms of physical time.

| **Code** | **AVBP$_{\mathbf{pgs}}$** | **ProLB** | **waLBerla** |
|---|---|---|---|
| Time step [s] | $4.4 \times 10^{-7}$ | $8.5 \times 10^{-7}$ | $7.1 \times 10^{-7}$ |
| Equivalent Fine Fluid Nodes [M] | 18.1 | 21.4 | 22.6 |
| CPU time per iteration [ms] | 83 | 90 | 24 |
| Reduced computational time [$\mu$s iteration$^{-1}$ cell$^{-1}$ core] | 1.67 | 1.50 | 0.38 |
| Cost 1 ms physical time [CPUh] | 19.1 | 10.6 | 3.4 |

Table 6: Comparison of computational efficiency of the three different solvers on a 360 cores run [51]

For Table 6, all outputs and post-processing routines are disabled: only the fluid solver itself is considered. All codes run on 360 processes on a cluster [51] which uses a Intel Xeon Gold 6140 Skylake chipset.

Table 6 shows that the LBM solvers are faster than the finite volume solver but the speed ratios are not different by orders of magnitude: the fastest code waLBerla goes 5 times faster than the AVBP$_{\mathrm{pgs}}$ solver.

### *5.1. Scaling*

In addition to the computational cost at a fixed number of cores, scalability is an important question in HPC: the strong scaling behaviour of the three codes was tested here from 36 to 900 cores. We measure the parallel efficiency $E$ by relating the the computational time per iteration $T_P$ on a given number of cores $N_P$ to the time per iteration $T_{36}$ on 36 cores, which is equivalent one full node on the utilized architecture.

$$E = \frac{36}{N_P} \frac{T_{36}}{T_P} \tag{6}$$

Fig. 9a shows that AVBP scales almost ideally over the whole range of cores, while the LBM solvers efficiencies drop by 30-40% when increasing the number of cores by a factor of 25. In this

particular configuration, parallel scalability is controlled by the spatial distribution of grid cells to achieve an even workload balance among all processes. In AVBP there are few constraints on the decomposition of the computational domain as long as the surface area between subdomains is kept at a minimum. Moreover, AVBP remains efficient even when only a few thousand mesh nodes are handled by each core.

ProLB preserves performance up to 72 cores but then loses efficiency beyond 144 cores because of an increase in waiting time: the solver has an optimum scalability estimated between $10^5$ and $10^6$ fluid elements per core to efficiently manage industrial configurations that require much larger grids. In other words, while weak scaling works well in ProLB, the present strong scaling exercise is more difficult since the order of magnitude of fluid elements per core is $10^4$ at 900 cores.

In waLBerla whole blocks are assigned to each core. When using an excessive amount of cores, there are not enough blocks per process to find an even workload distribution. On the other hand waLBerla exhibits excellent weak scaling until almost half a million cores [39, 40]. Furthermore in the LBM scheme the different levels of refinement have to be executed sequentially. This is an inherent obstacle to achieving even workload distribution and it limits strong scaling capabilities. The strong scaling limit of the LBM codes may affect the overall conclusion: at 900 cores, ProLB becomes less efficient than AVBP. waLBerla maintains the lowest computation time over the whole range of cores (Fig. 9b).

## 6. Conclusions

One finite volume and two Lattice-Boltzmann solvers suitable for Large Eddy Simulation have been compared in terms of accuracy and CPU efficiency in a swirling flow, a typical aeronautical application.

The fidelity of the three solvers was demonstrated by comparing numerical and experimental PIV data in terms of: injector head pressure losses, mean and RMS velocity profiles and axial velocity spectra. Despite minor differences, the three solvers provide very similar and accurate results: the discrepancies with respect to experimental results are limited to the tenth of $\mathrm{m\,s^{-1}}$ on velocity profiles. For pressure losses, the finite volume solver using body-fitted meshes captures the experimental result very well (330 Pa for $\mathrm{AVBP_{pgs}}$ versus 335 Pa for the experiment) but the two LBM codes results, using structured meshes are also close to measurements (368 Pa for ProLB and 313 Pa for waLBerla). These results confirm that LES formulations provide high accuracy

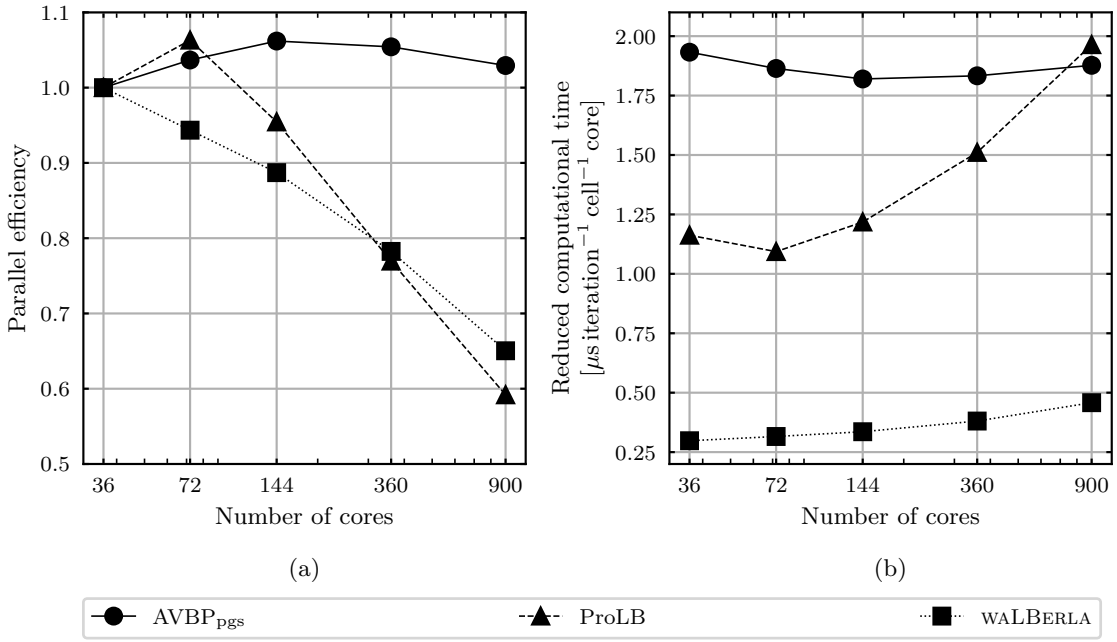


Figure 9: Parallel performance of the three codes up to 900 cores, normalized by the performance at 36 cores

results for swirled flows, much better than usual RANS codes especially in terms of RMS data for all components. Furthermore LDV data at two specific locations are used to create PSD analysis of axial velocity: the three solvers as well as the LDV data do not reveal any hydrodynamic mode. Moreover, AVBP$_{pgs}$ shows higher numerical dissipation in the high frequency range with respect to LBM solvers.

Strong scaling tests from 36 to 900 cores reveal that the finite volume solver maintains its performance, whereas the LBM codes exhibit some loss in efficiency as the workload per core decreases. On 360 cores the CPU times necessary to compute 1 ms of physical time are: 3.4 for waLBerla, 10.6 for ProLB and 19 CPU hours for AVBP$_{pgs}$. The three solvers offer similar orders of magnitude in terms of absolute performance, especially considering the fact that the finite volume solver carried more equations (energy and chemical species) as well as much more complex thermochemical models.

Finally, a point which has been left for further studies is the importance of the mesh quality on the results. The results shown in the paper were obtained with user-optimized meshes which play

a crucial role in the final results, maybe as important as the solvers themselves. AMR (Automatic Mesh Refinement) is clearly a topic to address in future works.

## Acknowledgments

The help and support of Dr Gabriel Staffelbach and Dr Thomas Jaravel on all HPC aspects of AVBP are gratefully acknowledged. We also want to thank CS Group for providing ProLB's sources and particularly Romain Cuidard for grateful discussions on CPU efficiency. The authors acknowledge CINES (Centre Informatique National de l'Enseignement Supérieur) of GENCI (Grand Équipement National de Calcul Intensif) for giving access to HPC resources under the allocations A0052B10157 and A0092B10627. Moreover it has received funding from the European Research Council under the European Union's Horizon 2020 research and innovation program Grant Agreement 832248, SCIROCCO.

## References

[1] P. Sagaut, S. Deck, Large eddy simulation for aerodynamics: status and perspectives, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences 367 (1899) (2009) 2849. doi:https://doi.org/10.1098/rsta.2008.0269.

[2] Z. Wang, K. Fidkowski, R. Abgrall, F. Bassi, D. Caraeni, A. Cary, H. Deconinck, R. Hartmann, K. Hillewaert, H. Huynh, N. Kroll, G. May, P.-O. Persson, B. van Leer, M. Visbal, High-order cfd methods: Current status and perspective 72 (2013) 811–845. doi:https://doi.org/10.1002/fld.3767.

[3] R. Löhner, Towards overcoming the LES crisis, International Journal of Computational Fluid Dynamics 33 (3) (2019) 87–97. doi:https://doi.org/10.1080/10618562.2019.1612052.

[4] Comparison of Various CFD Codes for LES Simulations of Turbomachinery: From Inviscid Vortex Convection to Multi-Stage Compressor, Vol. Volume 2C: Turbomachinery of Turbo Expo: Power for Land, Sea, and Air. doi:10.1115/GT2018-75523.

[5] M. Gatti, R. Gaudron, C. Mirat, T. Schuller, Impact of swirl and bluff-body on the transfer function of premixed flames, Proceedings of the Combustion Institute 37 (2019) 5197–5204. doi:https://doi.org/10.1016/j.proci.2018.06.148.

[6] F. Dupuy, M. Gatti, C. Mirat, L. Gicquel, F. Nicoud, T. Schuller, Combining analytical models and les data to determine the transfer function from swirled premixed flames, Combustion and Flame 217 (2020) 222–236. doi:https://doi.org/10.1016/j.combustflame.2020.03.026.

[7] T. Schonfeld, M. Rudgyard, Steady and unsteady flow simulations using the hybrid flow solver avbp, AIAA journal 37 (11) (1999) 1378–1385. doi:https://doi.org/10.2514/2.636.

[8] O. Colin, M. Rudgyard, Development of high-order taylor-galerkin schemes for unsteady calculations, J. Comp. Physics 162 (2) (2000) 338–371. doi:https://doi.org/10.1006/jcph.2000.6538.

[9] T. Poinsot, Prediction and control of combustion instabilities in real engines (Invited Hottel lecture), Proc. Comb. Inst. (2017) 1–28doi:http://dx.doi.org/10.1016/j.proci.2016.05.007.

[10] O. Vermorel, P. Quillatre, T. Poinsot, LES of explosions in venting chamber: a test case for premixed turbulent combustion models, Comb. Flame 183 (2017) 207–223. doi:http://dx.doi.org/10.1016/j.combustflame.2017.05.014.

[11] T. J. Poinsot, S. Lele, Boundary conditions for direct simulations of compressible viscous flows, Journal of computational physics 101 (1) (1992) 104–129. doi:https://doi.org/10.1016/0021-9991(92)90046-2.

[12] L. Selle, F. Nicoud, T. Poinsot, Actual impedance of nonreflecting boundary conditions: Implications for computation of resonators, AIAA journal 42 (5) (2004) 958–964. doi:https://doi.org/10.2514/1.1883.

[13] V. Granet, O. Vermorel, T. Leonard, L. Gicquel, , T. Poinsot, Comparison of non reflecting outlet boundary conditions for compressible solvers on unstructured grids, J. 48 (10) (2010) 2348–2364. doi:https://doi.org/10.2514/1.J050391.

[14] G. Daviller, G. Oztarlik, T. Poinsot, A generalized non-reflecting inlet boundary condition for steady and forced compressible flows with injection of vortical and acoustic waves, Computers & Fluids 190 (2019) 503–513. doi:https://doi.org/10.1016/j.compfluid.2019.06.027.

[15] J. Ramshaw, P. O'Rourke, L. Stein, Pressure gradient scaling method for fluid flow with nearly uniform pressure, Journal of Computational Physics 58 (3) (1985) 361–376. doi:https://doi.org/10.1016/0021-9991(85)90168-8.

[16] P. O'Rourke, F. Bracco, Two scaling transformations for the numerical computation of multidimensional unsteady laminar flames, Journal of Computational Physics 33 (2) (1979) 185–203. doi:https://doi.org/10.1016/0021-9991(79)90015-9.

[17] http://www.prolb-cfd.com/research-and-development/, cLIMB: ComputationaL methods with Intensive Multiphysics Boltzmann solver BPIFrance Project No. P3543-24000.
URL http://www.prolb-cfd.com/research-and-development/

[18] Y. Hou, D. Angland, A. Sengissen, A. Scotto, Lattice-Boltzmann and Navier-Stokes Simulations of the Partially Dressed, Cavity-Closed Nose Landing Gear Benchmark Case, in: 25th AIAA/CEAS Aeroacoustics Conference, American Institute of Aeronautics and Astronautics, Delft, The Netherlands, 2019. doi:10.2514/6.2019-2555.

[19] H. Touil, D. Ricot, E. Lévêque, Direct and large-eddy simulation of turbulent flows on composite multi-resolution grids by the lattice Boltzmann method, Journal of Computational Physics 256 (2014) 220–233. doi:10.1016/j.jcp.2013.07.037.

[20] J. Jacob, O. Malaspinas, P. Sagaut, A new hybrid recursive regularised Bhatnagar–Gross–Krook collision model for Lattice Boltzmann method-based large eddy simulation, Journal of Turbulence (2018) 1–26Publisher: Taylor & Francis. doi:10.1080/14685248.2018.1540879.

[21] J. Latt, B. Chopard, Lattice Boltzmann method with regularized pre-collision distribution functions, Mathematics and Computers in Simulation 72 (2-6) (2006) 165–168. doi:10.1016/j.matcom.2006.05.017.

[22] P. L. Bhatnagar, E. P. Gross, M. Krook, A Model for Collision Processes in Gases. I. Small Amplitude Processes in Charged and Neutral One-Component Systems, Phys. Rev. 94 (3) (1954) 511–525. doi:10.1103/PhysRev.94.511.

[23] Y. Feng, P. Boivin, J. Jacob, P. Sagaut, Hybrid recursive regularized thermal lattice Boltzmann

model for high subsonic compressible flows, Journal of Computational Physics 394 (2019) 82–99. doi:10.1016/j.jcp.2019.05.031.

[24] T. Astoul, Analysis and reduction of spurious noise generated at grid refinement interfaces with the lattice Boltzmann method, Journal of Computational Physics (2020) 51.

[25] T. Astoul, G. Wissocq, J.-F. Boussuge, A. Sengissen, P. Sagaut, Lattice Boltzmann method for computational aeroacoustics on non-uniform meshes: A direct grid coupling approach, Journal of Computational Physics 447 (2021) 110667. arXiv:2004.14887, doi:10.1016/j.jcp.2021.110667.

[26] D. Lagrava, O. Malaspinas, J. Latt, B. Chopard, Advances in multi-domain lattice Boltzmann grid refinement, Journal of Computational Physics 231 (14) (2012) 4808–4822. doi:10.1016/j.jcp.2012.03.015.

[27] J. Jacob, P. Sagaut, Solid wall and open boundary conditions in hybrid recursive regularized lattice Boltzmann method for compressible flows, Physics of Fluids 31 (12) (2019) 126103. doi:10.1063/1.5129138.

[28] S. Bocquet, D. Ricot, A. Sengissen, C. Vincent-Viry, B. Demory, M. Henner, F. Ailloud, Evaluation of the Lattice Boltzmann Method for Aero-acoustic Simulations of Industrial Air Systems, in: 25th AIAA/CEAS Aeroacoustics Conference, American Institute of Aeronautics and Astronautics, Delft, The Netherlands, 2019. doi:10.2514/6.2019-2415.

[29] B. Dorschner, S. Chikatamarla, F. Bösch, I. Karlin, Grad's approximation for moving and stationary walls in entropic lattice Boltzmann simulations, Journal of Computational Physics 295 (2015) 340–354. doi:10.1016/j.jcp.2015.04.017.

[30] M. Falese, A study of the effects of bifurcations in swirling flows using Large Eddy Simulation and mesh adaptation, Ph.D. thesis, Institut Natioanl Polytechnique de Toulouse (2013).

[31] G. Daviller, M. Brebion, P. Xavier, G. Staffelbach, J.-D. Müller, T. Poinsot, A Mesh Adaptation Strategy to Predict Pressure Losses in LES of Swirled Flows, Flow, Turbulence and Combustion (1) 93–118. doi:10.1007/s10494-017-9808-z.

[32] waLBerla.
URL https://www.walberla.net

[33] M. Bauer, S. Eibl, C. Godenschwager, N. Kohl, M. Kuron, C. Rettinger, F. Schornbaum, C. Schwarzmeier, D. Thönnes, H. Köstler, U. Rüde, WALBERLA: A block-structured high-performance framework for multiphysics simulations, Computers  Mathematics with Applications (2020). doi:10.1016/j.camwa.2020.01.007.

[34] T. Pohl, M. Kowarschik, J. Wilke, K. Iglberger, U. Rüde, Optimization and profiling of the cache performance of parallel lattice boltzmann codes, Parallel Processing Letters 13 (04) (2003) 549–560. doi:https://doi.org/10.1142/S0129626403001501.

[35] C. Feichtinger, S. Donath, H. Köstler, J. Götz, U. Rüde, Walberla: Hpc software design for computational engineering simulations, Journal of Computational Science 2 (2) (2011) 105–112. doi:https://doi.org/10.1016/j.jocs.2011.01.004.

[36] C. Feichtinger, J. Habich, H. Köstler, U. Rüde, T. Aoki, Performance modeling and analysis of heterogeneous lattice boltzmann simulations on cpu–gpu clusters, Parallel Computing 46 (2015) 1–13. doi:https://doi.org/10.1016/j.parco.2014.12.003.

[37] Z. Liu, X. Chu, X. Lv, H. Meng, S. Shi, W. Han, J. Xu, H. Fu, G. Yang, Sunwaylb: Enabling extreme-scale lattice boltzmann method based computing fluid dynamics simulations on sunway taihulight, in: 2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS), IEEE, 2019, pp. 557–566. doi:https://doi.org/10.1109/IPDPS.2019.00065.

[38] M. Bauer, H. Köstler, U. Rüde, lbmpy: Automatic code generation for efficient parallel lattice boltzmann methods (2020). arXiv:2001.11806.

[39] T. Pohl, F. Deserno, N. Thurey, U. Rude, P. Lammers, G. Wellein, T. Zeiser, Performance evaluation of parallel large-scale lattice boltzmann applications on three supercomputing architectures, in: SC'04: Proceedings of the 2004 ACM/IEEE Conference on Supercomputing, IEEE, 2004, pp. 21–21. doi:https://doi.org/10.1109/SC.2004.37.

[40] C. Godenschwager, F. Schornbaum, M. Bauer, H. Köstler, U. Rüde, A framework for hybrid parallel flow simulations with a trillion cells in complex geometries, ACM Press, 2013, pp. 1–12. doi:10.1145/2503210.2503273.

[41] pystencils.
URL https://i10git.cs.fau.de/pycodegen/pystencils

[42] M. Hall, Cell vertex multigrid solution of the euler equations for transonic flow past aerofoils, Rapport technique, Royal Aerospace Establishment (1984) 26.

[43] F. Nicoud, H. B. Toda, O. Cabrit, S. Bose, J. Lee, Using singular values to build a subgrid-scale model for large eddy simulations, Physics of Fluids 23 (8) (2011) 085106. doi:https://doi.org/10.1063/1.3623274.

[44] T. Krüger, H. Kusumaatmaja, A. Kuzmin, O. Shardt, G. Silva, E. M. Viggen, The Lattice Boltzmann Method: Principles and Practice, Graduate Texts in Physics, Springer International Publishing, Cham, 2017. doi:10.1007/978-3-319-44649-3.

[45] R. Courant, K. Friedrichs, H. Lewy, On the Partial Difference Equations of Mathematical Physics, IBM J. Res. & Dev. 11 (2) (1967) 215–234. doi:10.1147/rd.112.0215.

[46] E. Lévêque, F. Toschi, L. Shao, J.-P. Bertoglio, Shear-improved Smagorinsky model for large-eddy simulation of wall-bounded turbulent flows, Journal of Fluid Mechanics 570 (2007) 491–502, publisher: Cambridge University Press (CUP). doi:10.1017/S0022112006003429.

[47] K. Oberleithner, M. Stöhr, S. H. Im, C. M. Arndt, A. M. Steinberg, Formation and flame-induced suppression of the precessing vortex core in a swirl combustor: Experiments and linear stability analysis, Comb. Flame 162 (8) (2015) 3100–3114.

[48] S. Terhaar, K. Oberleithner, C. O. Paschereit, Key parameters governing the precessing vortex core in reacting flows: An experimental and analytical study, Proc. Comb. Inst. 35 (2014).

[49] J. P. Moeck, J.-F. Bourgouin, D. Durox, T. Schuller, S. Candel, Nonlinear interaction between a precessing vortex core and acoustic oscillations in a turbulent swirling flame, Comb. Flame 159 (8) (2012) 2650–2668.

[50] N. Syred, A review of oscillation mechanims and the role of the precessing vortex core in swirl combustion systems, Prog. En. Comb. Sci 32 (2) (2006) 93–161.

[51] KRAKEN.
URL https://cerfacs.fr/en/cerfacs-computer-resources/