Contents lists available at ScienceDirect

Comptes Rendus Mecanique

www.sciencedirect.com

High Performance Computing / Le Calcul Intensif

High performance parallel computing of flows in complex geometries

Calcul parallèle haute performance des écoulements en géométries complexes

Laurent Y.M. Gicquel [a,*], N. Gourdain [a], J.-F. Boussuge [a], H. Deniau [a], G. Staffelbach [a], P. Wolf [a], Thierry Poinsot [b]

[a] *CERFACS, 42, avenue Georges-Coriolis, 31057 Toulouse cedex , France*
[b] *IMFT, 1, allée du Professeur Camille-Soula, 31400 Toulouse, France*

A R T I C L E   I N F O

Article history:
Available online 8 January 2011

Keywords:
Computer science
Parallel computing
Computational Fluid Dynamics

Mots-clés :
Informatique, algorithmique
Calcul parallèle
Dynamique des fluides numérique

A B S T R A C T

Efficient numerical tools taking advantage of the ever increasing power of high-performance computers, become key elements in the fields of energy supply and transportation, not only from a purely scientific point of view, but also at the design stage in industry. Indeed, flow phenomena that occur in or around the industrial applications such as gas turbines or aircraft are still not mastered. In fact, most Computational Fluid Dynamics (CFD) predictions produced today focus on reduced or simplified versions of the real systems and are usually solved with a steady state assumption. This article shows how recent developments of CFD codes and parallel computer architectures can help overcoming this barrier. With this new environment, new scientific and technological challenges can be addressed provided that thousands of computing cores are efficiently used in parallel. Strategies of modern flow solvers are discussed with particular emphases on mesh-partitioning, load balancing and communication. These concepts are used in two CFD codes developed by CERFACS: a multi-block structured code dedicated to aircrafts and turbo-machinery as well as an unstructured code for gas turbine flow predictions. Leading edge computations obtained with these high-end massively parallel CFD codes are illustrated and discussed in the context of aircrafts, turbo-machinery and gas turbine applications. Finally, future developments of CFD and high-end computers are proposed to provide leading edge tools and end applications with strong industrial implications at the design stage of the next generation of aircraft and gas turbines.

© 2010 Académie des sciences. Published by Elsevier Masson SAS. All rights reserved.

R É S U M É

L'utilisation et l'accès à des codes de calcul numérique tirant avantage de la puissance croissante des calculateurs hautes-performances, est devenu un élément clef dans les domaines de la production d'énergie et des transports. Ces outils sont non seulement critiques d'un point de vue scientifique mais aussi pour les concepteurs de l'industrie. En effet, les écoulements autour des produits industriels de l'énergie ou de l'aéronautique sont si compliqués qu'ils nécessitent souvent l'utilisation de modèles simplifiés avec l'hypothèse de stationnarité. Ce document présente comment les dévelopements récents des codes de calcul et des calculateurs massivement parallèles peuvent aider à repousser ces limites. Dans cet environement particulier, de nouveaux challenges technologiques et scientifiques peuvent être abordés en utilisant efficacement des milliers de coeurs de calcul en parallèle. Le parallèlisme de ces solveurs modernes est décrit avec un regard particulier sur le

* Corresponding author.
E-mail addresses: laurent.gicquel@cerfacs.fr (L.Y.M. Gicquel), thierry.poinsot@cerfacs.fr (T. Poinsot).

1631-0721/$ – see front matter © 2010 Académie des sciences. Published by Elsevier Masson SAS. All rights reserved.
doi:10.1016/j.crme.2010.11.006

découpage de maillage, d'équilibrage de charge et de communication. Deux exemples issus des travaux du CERFACS sont utilisés pour illustrer ces concepts : un code de calcul multi-blocs structuré dédié à la simulation des écoulements avions et turbo-machines ainsi qu'un code non structuré ciblant les écoulements dans les turbines à gaz. Pour finir, des axes de recherche et développements de ces codes et calculateurs sont énnoncées afin d'identifier les utilisations futures possibles et en accord avec les besoins industriels apparaissant lors de la définition des futures concepts d'avions et turbines à gaz.

1. Introduction

Computational Fluid Dynamics (CFD) is the simulation, by computational methods, of the governing equations that describe the fluid flow behavior (usually derived from the Navier–Stokes equations). It has grown from a purely scientific discipline to become an essential industrial tool routinely used at various design stages. A wide variety of applications can be found in the literature, ranging from nuclear to aeronautic applications. In fact CFD flow solvers are considered to be standard numerical tools in industry for design and development, of aeronautic and propulsion domains. This approach is attractive since it reproduces physical flow phenomena more rapidly and at a lower cost than experimental methods. However, industrial configurations remain complex and CFD codes usually require high-end computing platforms to obtain flow solutions in a reasonable amount of time. Within the realm of super-computing architectures, High Performance Computing (HPC) has no strict definition but it usually refers to (massively) parallel processing for running quickly application programs. The term applies especially to systems that work above 10^{12} FLoating-point Operation Per Second (FLOPS). Recent HPC computers rely on superscalar cache-based computing cores interconnected to form systems of symmetric multi-processing nodes. This type of hardware has rapidly developed over the past decade due to its cost effectiveness since relying on everyone's personal computer technology. This tendency is best illustrated by the list of Bell Prize winner which, during the last four years, rewarded mostly massively parallel codes (www.top500.org). However, difficulties are still present with cache-based processors especially for most CFD applications [1] since the common computing efficiency of these codes is less than 20% on these machines. One of the reasons is that flow solvers process large amount of data, meaning a lot of communication required between the different cache levels and processors. Indeed such parallel platforms impose to run tasks on an ever larger number of computing cores to be attractive and provide new possibilities for CFD by reducing computational time and giving access to large amounts of memory.

Parallel platforms integrate many processors that are themselves composed of one or more computing cores (a Processing Unit (PU) corresponds to the task performed by one computing core). Programming for parallel computing where simultaneous computing resources are used to solve a problem can become very challenging and performance is related to a great quantity of parameters. With a relatively small number of PU (< 128), calculations efficiency scales with communications as they are the primary parameters that counterbalance the architecture peak power. When thousands of PUs are used, load balancing between cores and communications dominate and both aspects become the limiting factors of computational efficiency. This article focuses on the work done in CERFACS flow solvers to efficiently run on such powerful super-computers. The objectives are to illustrate the techniques and methods implemented in two CFD solvers specifically developed for HPC of industrial flows [2,3]. It aims at providing an overview of the authors' daily experience when dealing with such computing platforms for applications in aeronautical and gas turbine applications. Note that depending on the flow to be simulated, there is no consensus today in the CFD community on the type of mesh or data structure to be used on parallel computers; although the associated algorithm can greatly differ. Two different flow solvers are considered here to illustrate the different tasks that have to be considered to obtain a good efficiency on HPC platforms, with a special interest for (massively) parallel computing: a multi-block structured flow solver dedicated to aerodynamics problems (elsA) and an unstructured flow solver designed for reactive flows (AVBP). High-end physical solutions they can provide are also described and discussed. The intent of this discussion is to provide in the specific context of aeronautical industry, a first glimpse at the potential of current massively parallel CFD codes when used along with high end computing facilities. To conclude, potential directions and future computer code applications are proposed to further improve the impact of numerical tools in the daily tasks of engineers.

2. The flow solvers

Modern flow solvers have to efficiently run on a large range of super-computers, from single core to massively parallel platforms. The most common systems used for HPC applications are the Multiple Instruction Multiple Data (MIMD) based platforms [4]. This category can be further divided in two sub-categories, Fig. 1:

- Single Program, Multiple Data (SPMD). Multiple autonomous PUs simultaneously executing the same program (but at independent points) on different data. The flow solver elsA is an example of an SPMD program;
- Multiple Program, Multiple Data (MPMD). Multiple autonomous PUs simultaneously operating at least two independent programs. Typically such systems pick one PU to be the host (the "master"), which runs one program that farms in

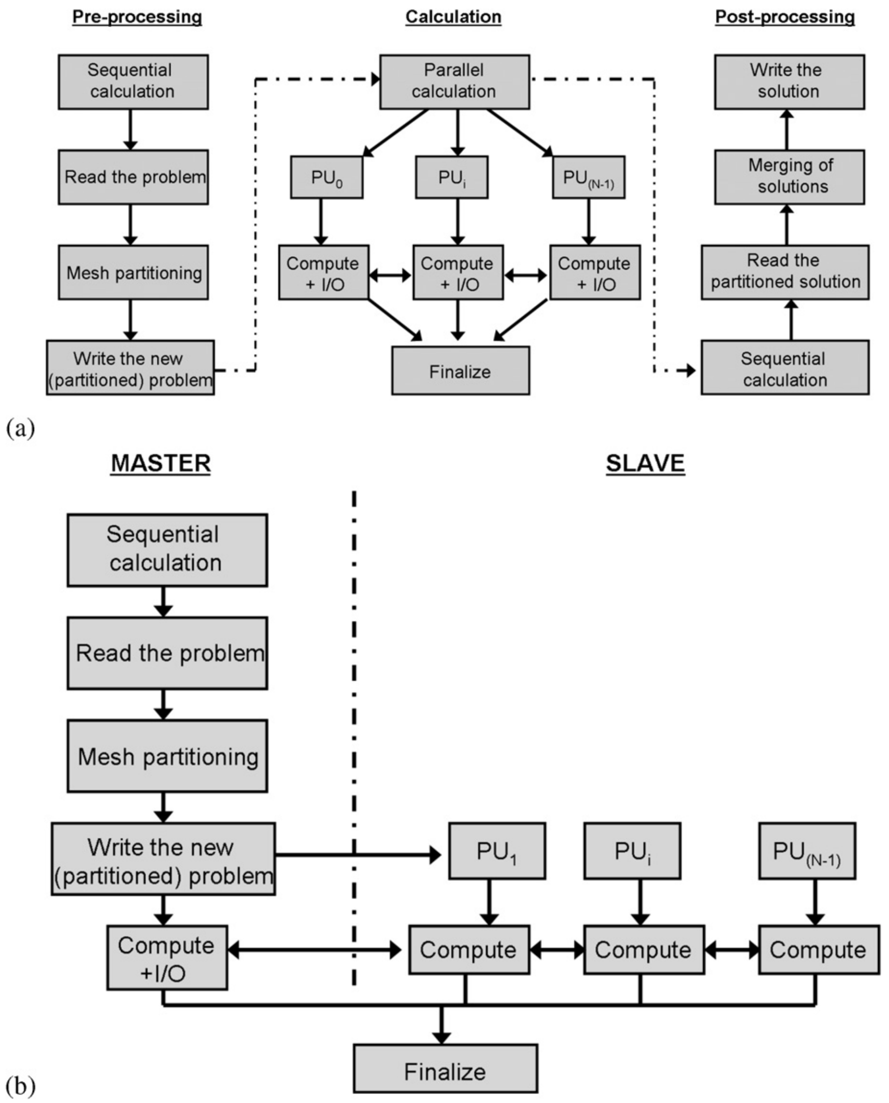


Fig. 1. Modern flow solver models: (a) SPMD and (b) MPMD concepts.

and out data to and from all the other PUs (the “slaves”) which all run a second program. The software AVBP uses this method (as well as SPMD), also referred to as the “master–slave” strategy.

The elsA software is a reliable design tool in the process chains of aeronautical industry and is intensively used by several major companies [5,6]. This multi-application CFD simulation platform deals with internal and external aerodynamics from low subsonic to hypersonic flow regime. The compressible 3D Navier–Stokes equations for arbitrary moving bodies are considered in several formulations according to the use of absolute or relative velocities. A large variety of turbulence models from eddy viscosity to full differential Reynolds stress models is implemented for the Reynolds-Averaged Navier–Stokes (RANS) equations [7], including criteria to capture laminar-turbulent transition phenomena for academic and industrial geometries [8]. In order to deal with flows exhibiting strong unsteadiness or large separated regions, high-fidelity methods such as Detached Eddy Simulation (DES) [9,10] and Large Eddy Simulation (LES) [11] are also available. For computational efficiency reasons, connectivity and data exchange between adjacent blocks are set by means of ghost cells. High-flexibility techniques of multi-block structured meshes have been developed to deal with industrial configurations. These advanced techniques include totally non-coincident interfaces [12] and the Chimera technique for overlapping meshes [13]. The flow equations are solved by a cell centered finite-volume method. Space discretization schemes include upwind schemes (Roe [14]; AUSM [15]) or centered schemes, stabilized by scalar or matrix artificial dissipation. The semi-discrete equations are integrated, either with multistage Runge–Kutta schemes or with a backward Euler integration that use implicit schemes solved by robust Lower–Upper (LU) relaxation methods [16]. For time accurate computations, the implicit Dual Time Stepping (DTS) method is available [17]. Several programming languages are used to develop the elsA software:

C++ is the main language for implementing the object-oriented design (backbone of the code), Fortran for CPU efficiency in calculation loops and Python for the user interface. The code has already been ported on a very large variety of platforms, including vector and massively scalar super-computers. Vector capabilities are still supported, but focus today is clearly on scalar parallel platforms with an increasing number of computing cores. To run in parallel, elsA uses a simple coarse-grained SPMD approach, where each block is allocated to a PU (note that in some circumstances several blocks can be allocated to the same PU). To achieve a good scalability, a load-balancing tool with block-splitting capability is included in the software.

The AVBP project was historically motivated by the idea of building a modern CFD software tool with high flexibility, efficiency and modularity. Recent information about this flow solver can be found in [18]. The solver is routinely used and developed in the frame of cooperative works with a wide range of industrial companies. AVBP is an unstructured flow solver capable of handling hybrid grids of many cell types (tetra/hexahedra, prisms, etc.). The use of these grids is motivated by the efficiency of unstructured grid generation, the accuracy of the computational results (using regular structured elements) and the ease of mesh adaptation for industrial flow applications. AVBP solves the laminar and turbulent compressible Navier–Stokes equations in two or three space dimensions and focuses on unsteady turbulent flows (with and without chemical reactions) for internal flow configurations. The prediction of these unsteady turbulent flows is based on LES which has emerged as a promising technique for problems associated with time dependent phenomena, and coherent eddy structures [19]. An Arrhenius law reduced chemistry model allows investigating combustion for complex configurations. The data structure of AVBP employs a cell-vertex finite-volume approximation. The numerical methods are the Lax–Wendroff scheme [20] or finite-element type low-dissipation Taylor–Galerkin [21,22] discretizations in combination with linear-preserving artificial viscosity models. Moving mesh capabilities following the Arbitrary Lagrangian–Eulerian (ALE) approach [23] have been developed to allow the simulation of moving bodies (piston in a cylinder, etc.). The AVBP library includes integrated parallel domain partitioning and data reordering tools, handles message passing and includes supporting routines for dynamic memory allocation, parallel Input/Output (I/O) and iterative methods.

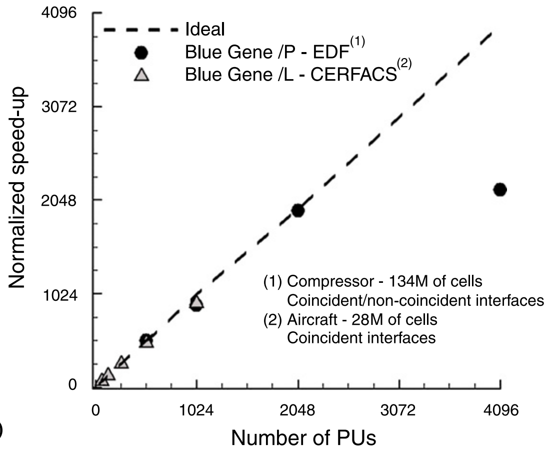
3. Parallel efficiency

In CFD, the most common and efficient parallel computing strategy is to decompose the original problem into a set of tasks. Simultaneous multiple computing resources are then used to solve the problem. A given CFD problem must therefore be divided into N subproblems (where N is typically the number of available PUs), and each part of the problem is solved by one of the PU concurrently. Ideally, the overall time $T_{parallel}$ of the computation will be $T_{sequential}/N$, where $T_{sequential}$ is the calculation time for the problem using only one PU.

3.1. Flow solver scalability

Usually, the parallel efficiency of a flow solver is evaluated with the “strong” speed-up that represents the gain with respect to a sequential calculation. However, this criterion does not give always satisfying indications. First, a good speed-up can be related to poor sequential performance (for example, if a PU computes slowly with respect to its communication network). This point is critical, especially for cache-based processors: super-linear speed-ups are often observed in the literature with these chips, mainly because the large amount of data required by the flow solver leads to a drop of the processor performance for a sequential task. For CFD applications, the convenient solution is to increase the size of the cache memory and to reduce the communication time between the different cache levels. Second, the sequential time is usually not known, mainly due to limited memory resources. Except for vector super-computer, the typical memory size available for one PU ranges from 512 Mb to 32 Gb. As a consequence, and for most cases, industrial configurations cannot be run on scalar platforms with a single computing core and no indication about the real speed-up can be obtained. A common definition of the speed-up is often used by opposition to the “strong” speed-up: the normalized speed-up that is related to the time ratio between a calculation with N PUs and N_{min} PUs, where N_{min} is the smallest number of PUs that can be used for the application. The normalized speed-up is thus useful to give an idea of the flow solver scalability.

A short overview of the normalized speed-up obtained for many applications performed with elsA and AVBP is indicated in Fig. 2. While elsA shows a good scalability until 2048 PUs, results for 4096 PUs are quite disappointing. This is mainly related to the difficulty for balancing a complex geometry configuration (non-coincident interfaces between blocks). Moreover, the load balancing error can be modified during this unsteady flow simulation (due to sliding mesh). AVBP has a long experience in massively parallel calculations, as indicated by the wide range of scalar super-computers used. This flow solver shows an excellent scalability until 6144 PUs since the normalized speed-up is very close to ideal. Results are still good for 16,000 PUs (efficiency is around 92%). This decrease of performance for a very large number of PUs (point around 12,000 PUs) underlines a physical limit often encountered on massively parallel applications. Indeed for very large numbers of PUs the ratio between computation time and communication time is directly proportional to the problem size (number of grid points and unknowns). For this specific illustration, the configuration tested with 12,000 PUs corresponds to a calculation with less than 3000 cells by PU or a too low computational workload for each PU compared to the amount of exchanges needed to proceed with the CFD computation. It shows that a given task is limited in terms of scalability and no increase of performance is expected beyond a given number of PUs. The notion of “strong” speed-up still remains of importance although more difficult to ensure. Indeed, it is always desirable to ensure that a CFD code allows to decrease the overall simulation time when the number of processors is increased for a given computational domain. Ideally, the overall waiting

(a)


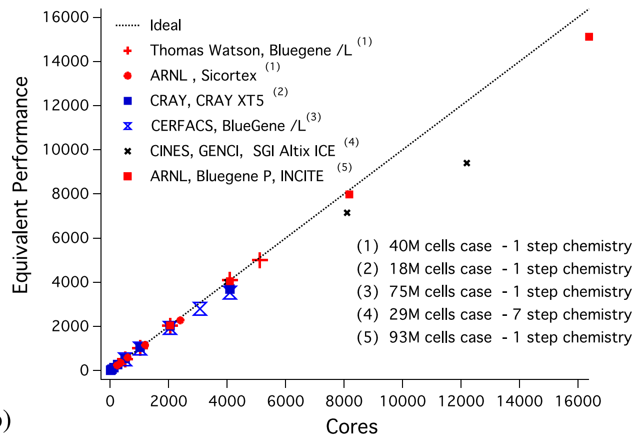
(b)


Fig. 2. Normalized speed-up curves obtained for (a) elsA and (b) AVBP on different HPC super-computing facilities.

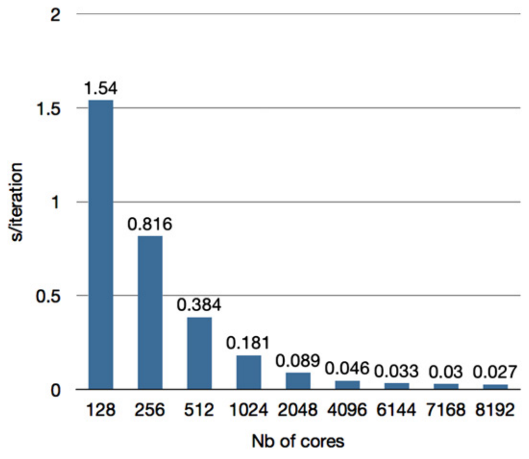


Fig. 3. Strong speed-up curve obtained for AVBP: i.e. fixed size problem and varying number of PUs used for the same computation.

time to produce the results should decrease nearly proportionally with the increase in PU number. Such a behavior ensures increased flexibility at the design stage by giving access to a large amount of design tests on fixed size problems. A typical strong speed-up obtained with AVBP is illustrated in Fig. 3.

3.2. Mesh partitioning

The critical step to maintain good efficiency as the number of computing cores increases relies in the mesh partitioning or the generation of the *N* sub-domains created from the original problem. Intuitively and for CFD codes where a mesh is needed, this step can be linked to the splitting of the grid CFD domain aiming at simulating the flow, Fig. 4. Many claims of “poor parallel scaling” are often ill-found and can vanish if care is brought to ensure good load balancing and minimization of overall communication time. Indeed, the possibility to run a problem on a large number of PUs is related to the capacity of efficiently splitting the original problem into parallel sub-domains each with an associated amount of arithmetic operations. The original mesh is partitioned in several sub-domains that are then (ideally equally) distributed between all PUs. One of the limitations is that the use of parallel partitioning algorithms often needed when using many computing cores remains a very complex optimization problem and such a task is usually performed sequentially by one PU, leading to memory constraints. These constraints are much more important for unstructured meshes than for block-structured meshes, mainly because mesh-partitioning of unstructured grids requires large tables for mesh coordinates and connectivity tables.

3.2.1. General organization

In most flow solvers, mesh partitioning is done in three steps: graph partitioning, nodes (re)ordering and blocks generation. An additional step is the merging of output data generated by each subtask (useful in an industrial context for data visualization and post-processing). As shown in Fig. 1, two strategies can be used: either the mesh-partitioning is done as a pre-processing step and all computing cores execute the same program during the computation (the SPMD approach used

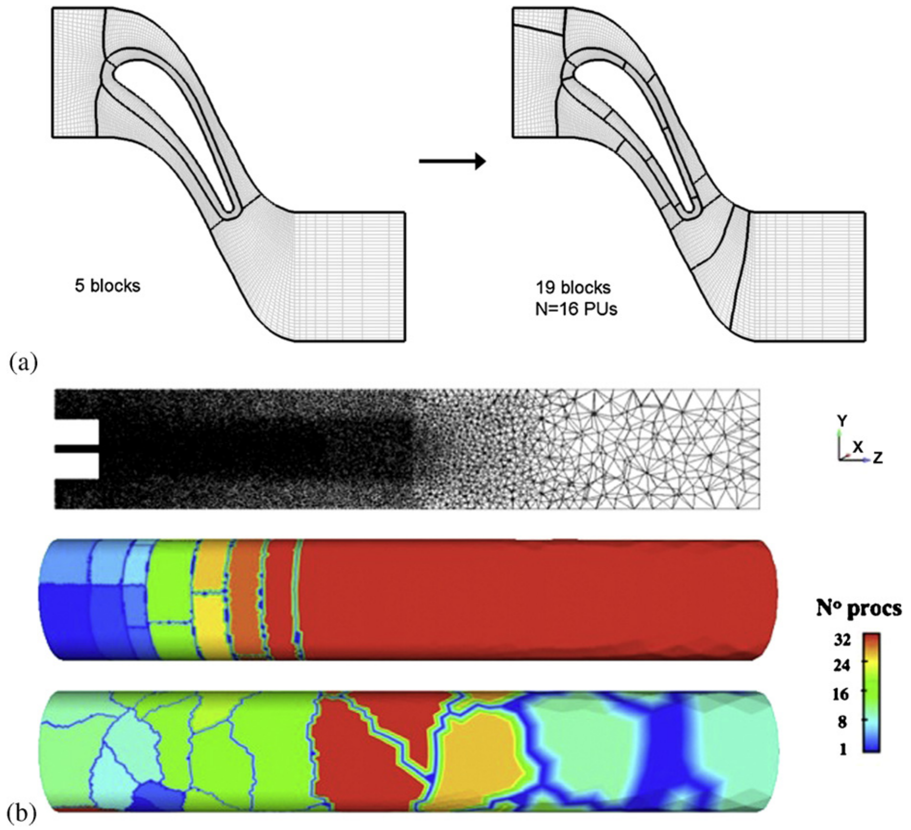


Fig. 4. Illustration of the partitioning process applied to (a) a structured mesh to be used with elsA and (b) an unstructured mesh of AVBP.

by elsA), or a master PU is designed to perform the mesh partitioning task before or during the calculation (the MPMD approach used by AVBP). In the first approach (Fig. 1(a)), a new problem adapted to the desired number of PUs is first defined, and the parallel calculation is performed using all PUs for computation and I/O. However, this partitioning strategy does not allow dynamic partitioning procedures which can be of interest for specific applications and that happens during the calculation when load balancing errors may vary during the calculation. The second approach (Fig. 1(b)) is more user-friendly (the mesh partitioning step is hidden to users) and is adapted to dynamic partitioning and (re)meshing (for moving bodies). During the computation, the master PU can be dedicated only to I/O or can be used for computation and I/O. Note that during the computational steps, the communication strategy defining the exchanges between master and slave PUs can become quite a complex problem (especially if evolving as the computation proceeds) and very penalizing for parallel efficiency.

3.2.2. Partitioning algorithms and parameters

Splitting the original mesh into a new set of sub-domains is a necessary step to obtain correct load balancing. Some specific and very often code dependent or flow physical constraints have to be taken into account at this stage to ensure the proper definition of the optimization problem that is supplied to the splitting algorithm.

For example, multi-grid capacities and connectivity between blocks (coincident or non-coincident interfaces) are of clear importance. A major difficulty with multi-block meshes is often related to the use of ghost cells that are needed for boundary condition treatments and information exchanges between blocks. Calculations that use thousands of PUs impose to split the original configuration in order to obtain a very high number of blocks, resulting into a number of ghost cells that can be largely increased, Fig. 5. Ghost cells affect directly the overall memory requirements and the computational time (the same quantity of information is stored in "real" and ghost cells) as well as the numerical scheme accuracy. The second example stems from multi-phase flow problems where droplets evolve freely in a stream of air that is affected by the surrounding walls of a combustor. This problem, typical of gas turbine combustors can be addressed using a Lagrangian solver that tracks the evolution and position of individual droplets within a grid of the combustor and onto which a gaseous flow field is obtained by the conventional Euler models. Of course droplet trajectories are in this case not known *a priori* and can cover all PU sub-domains or jump from one PU to its neighbors resulting into load balancing errors: i.e. one PU is in charge of a large number of particles (a lot of work) when others have no particle (no work). As illustrated by Fig. 6, depending on the sub-domain definition clear differences can appear in the work load distribution among PUs.

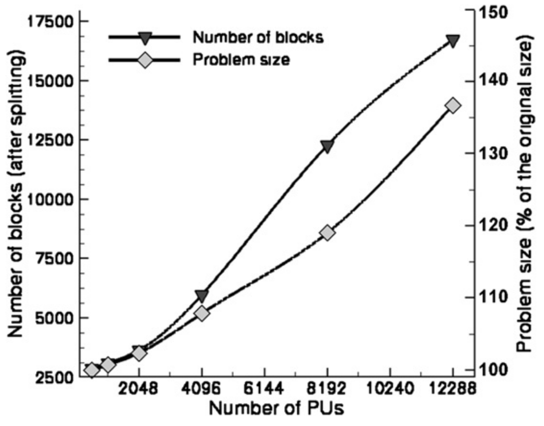


Fig. 5. Effect of the ghost cell number on the overall size of a CFD problem as a function of the number of sub-domains (elsA application).

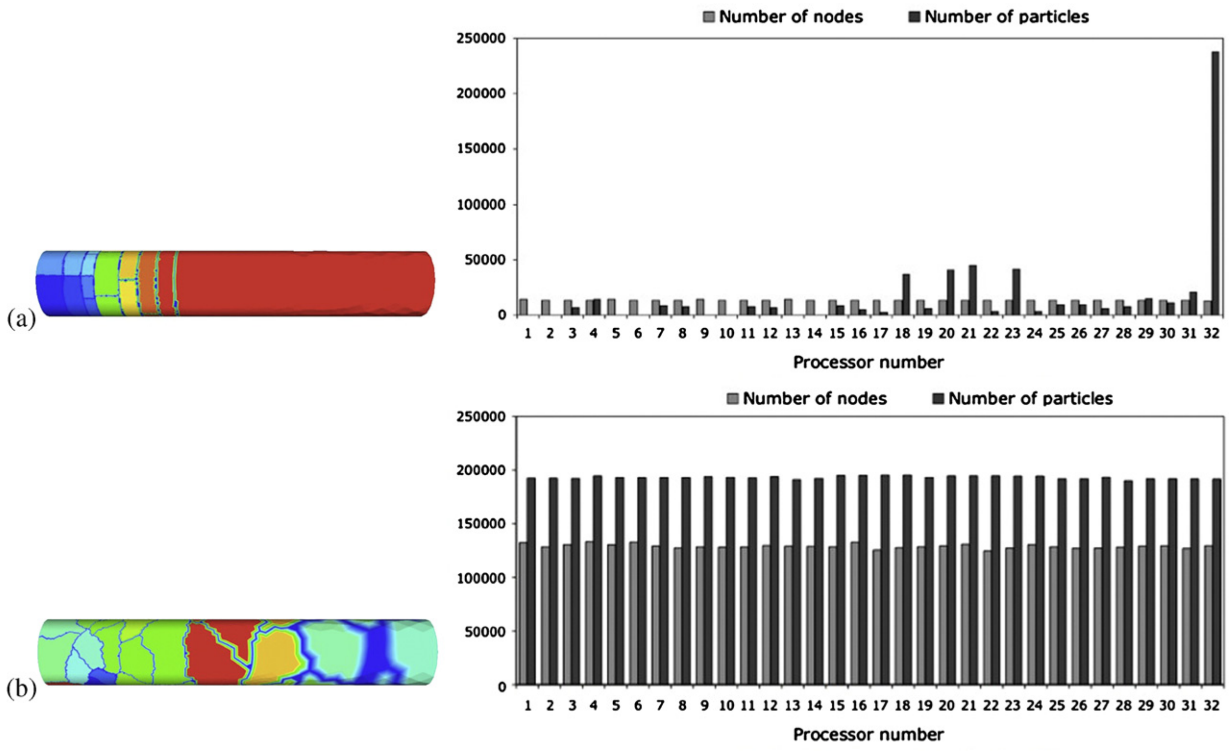


Fig. 6. Effect of the splitting strategy on the work load balance in the context of two-phase Lagrangian–Eulerian CFD flow solver (AVBP application).

Both examples point to the relative differences in work load and the difficulty in quantifying this load or more importantly its localization in the grid. Hence, relative errors appear between the allocated number of operations used to produce the splitting and the ideal number of operations for each PU which would result into an optimal work load of the parallel code. The primary objective of mesh partitioning is to minimize such load balancing errors. Many mesh partitioning algorithms are available in elsA and AVBP. All can be recast into the symbolic illustration of Fig. 7. The example of an unstructured grid and its associated dual graph is shown in Fig. 7. Each vertex represents a finite element of the mesh and each edge represents a connection between the faces of two elements (and therefore, the communication between adjacent mesh elements). The number of edges of the dual graph that are being cut by partitioning is called an edge-cut. This subdivision results in an increase of the total number of grid points due to a duplication of nodes at the interface between two sub-domains as pointed out earlier. The communication cost of a sub-domain is a function of its edge-cut as well as the number of neighboring sub-domains that share edges with it. In practice, the edge-cut of a partitioning is used as an important indicator of partitioning quality for cell-vertex codes. It also relates to a sparse matrix bandwidth that needs to be re-ordered to optimize the code memory use and performance.

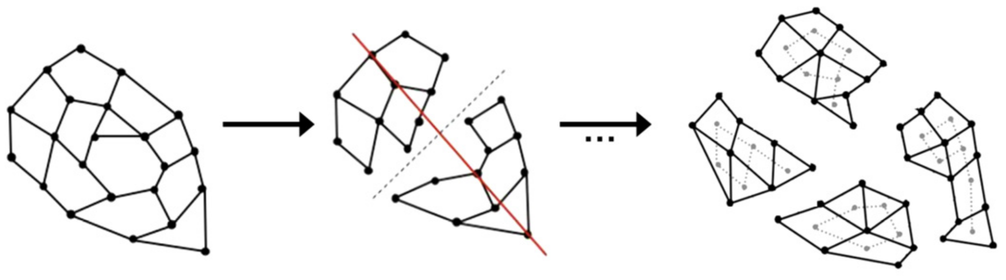


Fig. 7. Representation of a grid partitioning problem.

Many partitioning algorithms exist. They are geometric based algorithms, considering only coordinates and distances between nodes (Euclidean distance). Or graph theory based algorithms that use the graph connectivity information. All these algorithms give satisfactory results for most applications but show limitations in terms of computational time and mesh partitioning quality for large size problems (such as the one usually considered with HPC). New algorithms have thus been implemented (METIS) in AVBP for example [24], which are especially well adapted for Lagrangian and massively parallel calculations. A short description of these methods available in elsA and AVBP is provided below.

- The Recursive Edge or Coordinate Bisection (REB or RCB) algorithm [25] works only on the largest domain (it splits the largest block along its longest edge until the number of blocks equals the desired number of PUs indicated by the user). This very simple algorithm is well adapted to simple geometries such as cubes but it usually gives poor quality results in complex configurations;
- The so-called greedy algorithm is a more powerful tool to split blocks and is based only on geometric criteria [26]. This approach loops over blocks looking for the largest one (after splitting when necessary) and allocating it to the PU with the smaller number of cells until all blocks are allocated.
- Recursive Inertial Bisection (RIB) is similar to RCB [27] but it provides a solution that does not depend on the initial mesh orientation. The weakness of this algorithm is identical to the RCB method.
- Recursive Graph Bisection (RGB) considers the graph distance between two nodes [28]. In AVBP, the determination of pseudo-peripheral nodes, which is critical for the algorithm, is obtained with the Lewis implementation of the Gibbs–Poole–Stockmeyer algorithm [29]. The interest of this method is that it considers the graph distance.
- The METIS algorithms are based on multilevel graph partitioning: multilevel recursive bisectioning [24] and multilevel k-way partitioning [30]. As the partitioning algorithms operate with a reduced-size graph, they are extremely fast compared to traditional partitioning algorithms. Testing has also shown that the partitions provided by METIS are usually better than those produced by other multilevel algorithms [31].

3.2.3. Communications and scheduling

A final and sometimes hardware related parameter is to be taken into account. This is the ability to transfer/exchange/retrieve data between PUs. Indeed, for most CFD applications, processes are not fully independent and data exchange is required at some points (fluxes at block interfaces, residuals, etc.). The use of an efficient strategy for message passing is thus necessary for parallel computing, especially with a large number of computing cores and the potential differences in interconnecting networks, speeds and numbers available on the various HPC architectures. Specific instructions are given through standard protocols to communicate data between computing cores, such as Message Passing Interface (MPI) and OpenMP with specific command lines to do the same operations in a more or less efficient way depending on the hardware present.

3.3. Impact on numerical solutions

Mesh-partitioning, node/block and communication ordering can affect the numerical solution. Message scheduling are responsible for a nondeterministic behavior and mesh-partitioning modifies the size of the initial problem. The effects of these features on rounding errors and implicit schemes need to be monitored to ensure that solutions are reliable. Typical behaviors and tests are presented below.

Implicit stage done on a block basis (such as in elsA) is a difficulty for mesh-partitioning. Block splitting performed to achieve good load balancing may reduce the numerical efficiency of the implicit algorithm. Simulations of the flow in a high-pressure turbine have been performed (without block splitting) with different numbers of PUs, using RANS then LES. For both elsA tests, no difference is observed on the instantaneous solutions and convergence is strictly identical. A second set of simulation is performed by considering the same test case but with different mesh partitioning keeping an identical number of PUs. The unsteady RANS flow simulation is performed with elsA and results of tests are shown in Fig. 8. No significant difference on the convergence level between both configurations is observed until a normalized time of $t^+ = 10$. After $t^+ = 10$, although convergence levels are of the same order, differences are observed on the instantaneous residuals. In practice, no significant degradation on convergence level has ever been reported.

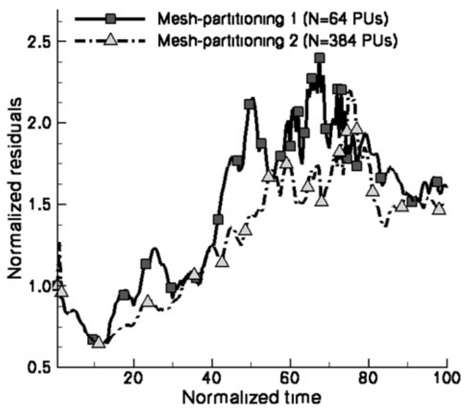


Fig. 8. Illustration of the partitioning on the temporal evolution of an elsA simulation.

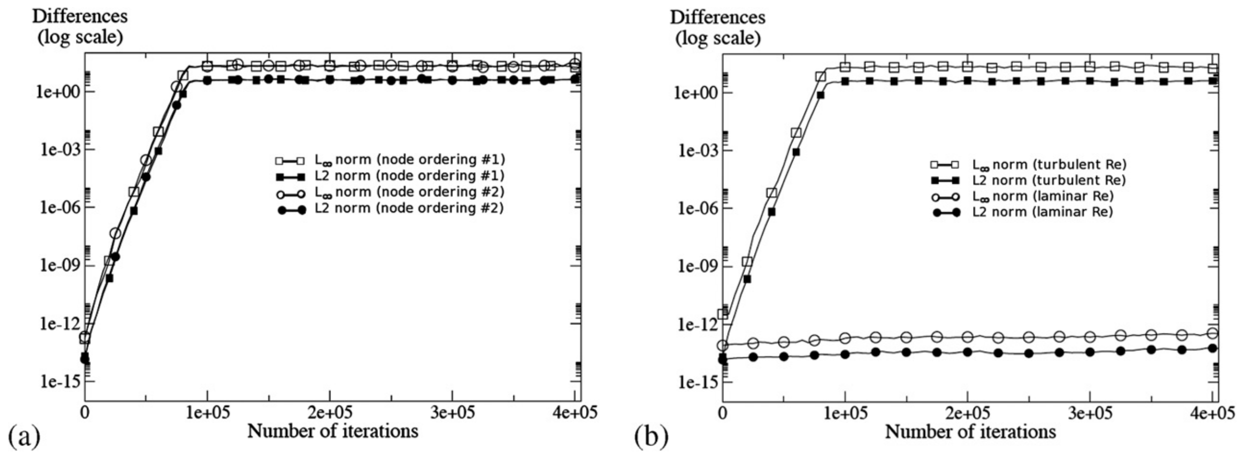


Fig. 9. Illustration of the partitioning on the temporal evolution of an AVBP simulation (L_∞- and L_2-norms of the difference between the multi-processor prediction and the single sector prediction): (a) effect of node ordering and (b) effect of the flow Reynolds number.

The effect of rounding errors is evaluated with AVBP for LES applications [32]. Rounding errors are not induced only by parallel computing and computer precision is also a major parameter. However, non-blocking MPI communications (such as used by AVBP) induces a difficulty: the arbitrary message arrival of variables to be updated at partition interfaces and the subsequent differences in the addition of the contributions of cell residuals at these boundary nodes is responsible for an irreproducibility effect [33]. A solution to force a deterministic behavior is to focus on the reception of the overall contributions of the interface nodes (by keeping the arbitrary message arrival) and its posterior addition always in the same order. This variant of AVBP is used for error detection and debugging since it would be time and memory consuming in massively parallel machines. This technique is used to quantify the differences between solutions produced by runs with different node ordering for a turbulent channel flow test case, computed with LES. Mean and maximum norms are computed using the difference between instantaneous solutions obtained with two different node ordering and results are shown in Fig. 9(a). The difference between instantaneous solutions increases rapidly and reaches its maximum value after 100,000 iterations. The same test is performed for a laminar Poiseuille pipe flow, in order to show the role of turbulence, and results are compared with a fully developed turbulent channel flow in Fig. 9(b). While instantaneous solutions for the turbulent channel diverge rapidly (even if statistical solutions remain identical), the errors for the laminar case grow only very slowly and do not exceed the value of 10–12 $\mathrm{m\,s^{-1}}$ for the axial velocity component. This behavior is explained by the fact that laminar flows do not induce exponential divergence of flow solution trajectories in time. In contrast, the turbulent flow acts as an amplifier for rounding errors. The role of the architecture precision and the number of computing cores play an identical role in such problems. The use of high machine precision such as double and quadruple precision shows that the slope of the perturbation growth is simply delayed. The conclusion is that instantaneous flow fields produced by LES are partially controlled by rounding errors and depends on multiple parameters such as node reordering, machine precision and initial conditions. These results confirm that LES reflects the true nature of turbulence insofar as it may exponentially amplify very small perturbations. It also points out that debugging LES codes on parallel machines is a complex task since instantaneous results cannot be used to compare solutions.

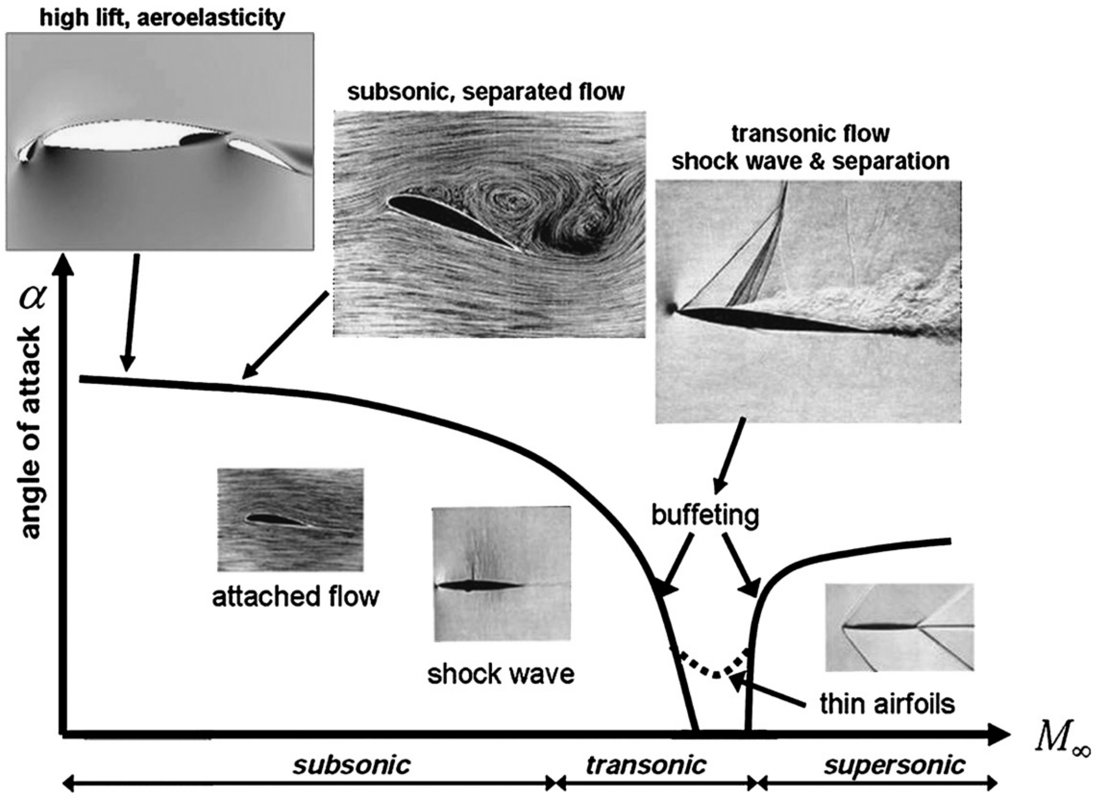


Fig. 10. Overview of the flow phenomena, depending on the flight conditions [34,3].

4. Recent applications with industrial implications

The following section illustrates the use of elsA and AVBP CFD solvers on industrial flows. Emphasis is here focused on the type of problems encountered by the aeronautical and power generation industries as well as on the type of solutions CFD can offer if properly used on HPC facilities. Note that although code architecture is of great importance, as underlined in the previous section, the fundamental strategy devised to model turbulent flows is also to be considered for an efficient use of CFD by industry. One major contribution of HPC in the industrial context and which is partly highlighted below, is that HPC and massively parallel codes can not only offer an efficient way to fully capitalize on the classical turbulent closure methods as proposed with RANS, but they also offer access to more advanced modeling strategies that are mandatory to extend our understanding of these complex flows. Three specific areas of interest are addressed: flows around aircraft, with examples of aeroelasticity, buffeting and aeroacoustic simulations; flows in rotating machineries and the advent of unsteady CFD models; and to finish, a gas turbine application and the contribution of LES for the prediction of thermo-acoustic instabilities.

4.1. Aircraft design

Designers face complex challenges for the next generation of civil aircraft that will operate under strict environmental rules in an ever increasing market. Performance estimation plays an extensive role at all stages of a commercial aircraft development program and design methodologies, together with computational methods, largely support the creation of these aircraft. Innovative design requires interdisciplinary, multi-physics and interactive design methods that need more and more computational power. In this context, a rapid and detailed design of a greener and quieter aircraft could benefit from HPC platforms. First, HPC can be useful at the design stage by reducing the time required to obtain the solution (at constant cost). Second, based on the engineering experience, the number of numerical simulations necessary to evaluate the complete flight domain (cruise conditions, high lift, take-off, etc.) could be as high as a few thousands. HPC is a very interesting tool to better manage the scheduling of this task. Another direct application of HPC is to achieve "high-fidelity" simulations in a reasonable amount of time, taking into account unsteady flows (URANS/LES) and technological effects. An overview of the flow phenomena encountered during flight conditions is proposed in Fig. 10. The region above the dark line represents the forbidden flight domain, due to undesirable phenomena. As shown, the design of new aircrafts has to tackle with complex physics such as shock/boundary layer interaction (buffet phenomenon), massively separated flows, aeroelastic instabilities or aeroacoustic prediction.

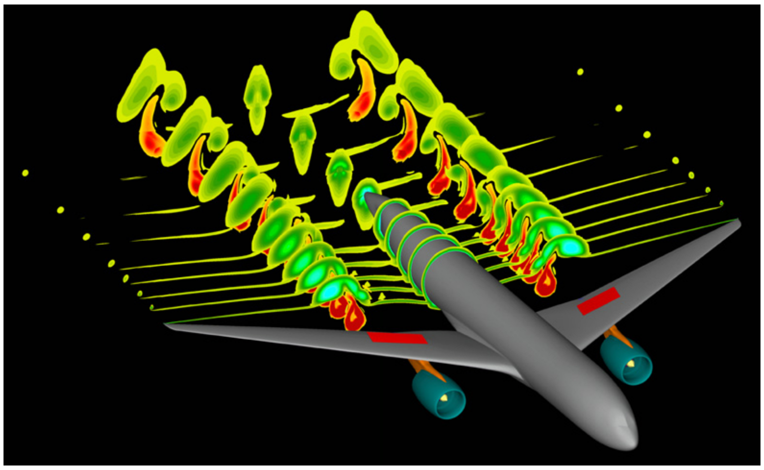


Fig. 11. Simulation of aeroelastic effects in a whole generic long range aircraft performed with elsA – instantaneous solution of total pressure.

Table 1
Comparisons of TSM and URANS costs for aeroelastic simulations (elsA).

Computing method	Computational time	Memory consumption
Standard aeroelastic solver	1	1
TSM-aeroelasticity, 1 harmonic	0.09	3
TSM-aeroelasticity, 2 harmonic	0.16	5
TSM-aeroelasticity, 3 harmonic	0.24	7

4.1.1. Aeroelasticity simulations

Today, the flow simulation around a whole generic aircraft (including propulsion system) is routinely done (see Fig. 11) using RANS approach. Nevertheless, if the aeroelastic efforts are required on the flight envelope, the computational resource necessary increases drastically and a direct fluid/structure coupling is still beyond industrial resources. A common practice is thus to perform unsteady forced motion simulations to obtain the unsteady loads on the wings of a complete aircraft configuration [35]. For such systems, the first issue is the size of the problem (a typical grid ranges from 30 to 100 million cells); and a second point of concern is the unsteady nature of the studied phenomena that requires a long turnaround time. The classical unsteady approach to treat such configurations is the URANS method. Although it can potentially capture most of physics nonlinearities, this method suffers from a high computational cost: the simulation must cover several periods of the unsteady phenomena, leading to a long transient phase (from 2 to more than 20 periods). For time-periodic flows (which is the case in forced motion simulations), it is possible to use an alternative method called TSM for Time Spectral Method [36], also referred as the harmonic balance technique [37]. Based on a Fourier decomposition of the flow variables, it transforms an unsteady problem into several coupled steady calculations. The coupling is done by a source term that can be viewed as a spectral derivative operator, expressed in the time domain. The TSM strategy has been successfully applied to the CFD and the structural dynamics solver. Table 1 presents a comparison of the normalized computational time and memory usage obtained with standard (URANS) and TSM aeroelasticity solvers: the use of TSM significantly reduces the total computational time. For an industrial aircraft configuration, the CPU time savings range from a factor 10 for a single harmonic representation to a factor 4 when three harmonics are considered. However, the counterpart for this CPU reduction is an increased memory requirement: ranging from a factor 3 for a single harmonic to 7 for 3 harmonics. This approach requiring a large amount of memory (depending on number of harmonics), is ideally suited to parallel computing. In this context, HPC is an interesting solution to reduce the cost of aeroelastic simulations.

4.1.2. Application to buffeting

Aircraft buffeting is a vibratory phenomenon that may appear during maneuvers or cruising conditions. Buffeting is eventually driven by aerodynamic mechanisms, involving separations of the wing boundary layer, but the performance limitation depends on the structure dynamic response. Buffeting can reduce the life duration of components due to high vibration levels. As a consequence, manufacturers have to design aircraft with a large lift margin that tends to limit minimum flying speed and maximum cruising altitude of the aircraft. The understanding of this phenomenon is thus crucial for designers. However, the simulation of buffet with CFD tools is challenging, due to massive flow separations and unsteady behavior. Considering realistic aircraft shapes, it is now well established that buffeting exhibits a broad band frequency distribution which presents high instability levels. Buffeting on 3D configurations can be considered as an interaction between the separation unsteadiness and the shock, so that (U)RANS methods are not adapted to this kind of simulation since the basis of these approaches is a full modeling of the turbulent spectrum (only decoupled phenomena are correctly simulated such

Table 2
Typical computing effort related to RANS and ZDES simulations of transonic buffet on a civil aircraft configuration – NEC-SX8+ computations with elsA.

	RANS (steady)	ZDES (unsteady)
Number of points	10,000,000	10,000,000
Total CPU (hours) effort for converged state (RANS) or 30 ms of simulation (ZDES)	10	2000

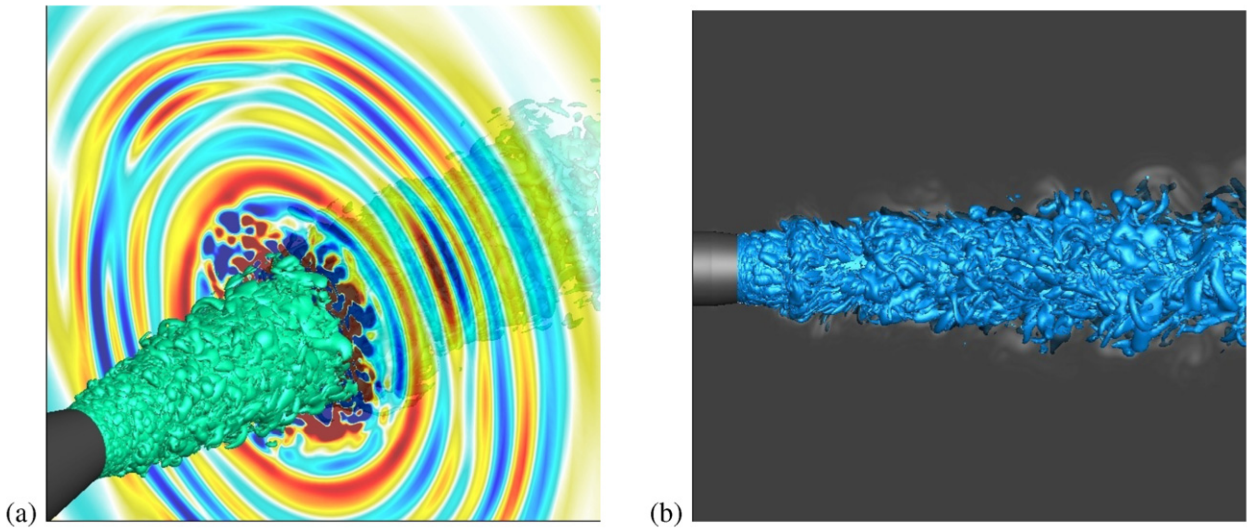


Fig. 12. Iso-surface of vorticity and dilatation field: (a) single jet and (b) coaxial jet.

as Von Karman streets). A promising way is thus to perform a simulation that solves (at least a part of) turbulent eddies. However and despite the fact that LES is a clear solution for such problems, it is still very costly in terms of computational resources for aircraft configurations at realistic Reynolds numbers. It is also noted that RANS methods are accurate enough in boundary layers for a low CPU cost. Therefore, RANS/LES hybrid methods, such as the DES approach [10] that proposes to use RANS for attached boundary layers and LES for separated regions, are very promising. The Zonal-DES (ZDES) method suggested by Deck [9] is an adaptation of the original DES approach to overcome some specific difficulties related to DES (the so-called "grey region"). Such costly unsteady flow simulations take advantage of parallel computing platforms to reduce the computational time (and cost) and obtain a correct description of complex flow phenomena. The feasibility of a 3D buffet simulation with the ZDES method implemented in elsA was studied by Brunet and Deck [38] on a realistic aircraft configuration ($\mathrm{Re} = 2.8 \times 10^6$, $M_\infty = 0.82$). The computing efforts are compared in Table 2 for RANS (using the Spalart–Allmaras turbulence model [39]) and ZDES methods. The cost ratio between the two approaches is around 200. While a purely RANS simulation is not able to reproduce the unsteady behavior of buffet, ZDES correctly predicts most flow features related to buffet such as unsteady separated flow region that triggers the shock movement.

4.1.3. Aeroacoustic simulation

During the last years, the need of high fidelity simulations on complex geometries for aeroacoustic predictions has grown. In order to characterize the acoustic field, we need to capture pretty small variation of the pressure field (of the order of few Pascal). Classical solvers used in industry are based on Finite Volume approach with second order discretization in space and time. This numerical modeling is inadequate for acoustic prediction. In order to overcome this issue, high order schemes (6th order in space) have been implemented in elsA [40] and tested on jet noise configuration (Fig. 12) with the LES formulation. Such unsteady simulations on dense mesh (several tens of million points) request several weeks of CPU time on several hundreds of processors. This kind of application is simply unthinkable without the use of HPC platforms.

4.2. Rotating machines

Rotating machines, such as encountered in gas or wind turbines, are involved in most processes for the conversion of energy. Increasing their efficiency is thus necessary to reduce pollutant emissions related to energy supply and transportation systems. However, the flow features in these machines are still not well understood mainly because the flow is highly turbulent, 3D and the fixed and moving parts strongly interacts together (the so-called "rotor–stator interactions"). The consequence of this lack of knowledge is a difficulty to design highly efficient turbo-machines. While experimental campaigns are the historic way to provide reliable data for studying these flows, this approach leads to complex measurement tech-

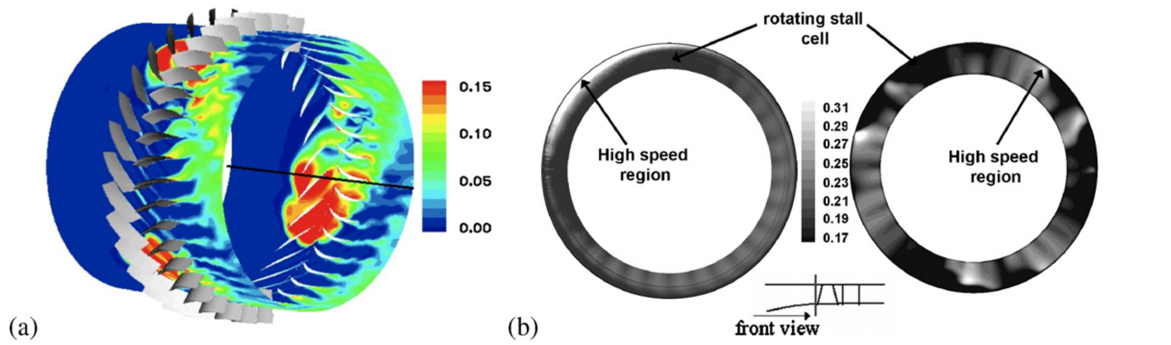


Fig. 13. (a) Simulation of rotating stall in a subsonic compressor [42] (unsteady RANS, 30 M points grid) and (b) comparison with experimental data.

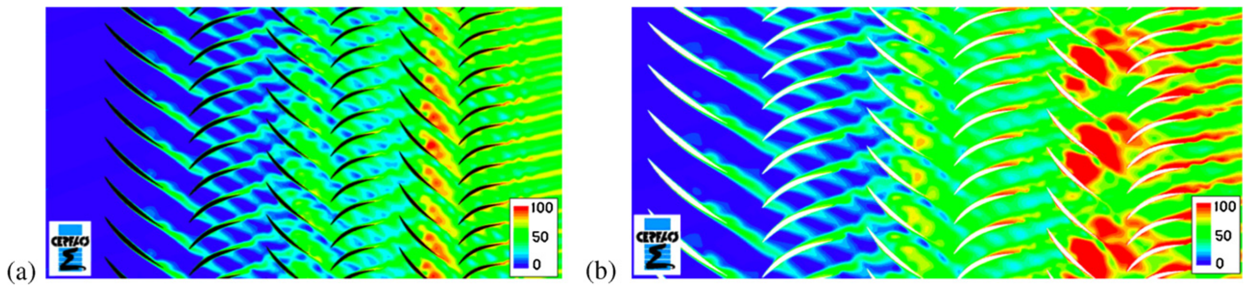


Fig. 14. Unsteady flow simulation in a multistage compressor at (a) design operating conditions and (b) off-design conditions [45]. Instantaneous flow field colored with entropy.

niques and high costs and delays. Again, CFD offers a very attractive method to reduce cost and design time and improve the knowledge. However, flow solvers require a very large computational power to tackle such complex flows, especially in industrial configurations. With classical computing methods (single processor), the flow simulation is limited to a small part of the system (such as an isolated blade) that is solved with a steady-state assumption. The use of a massively parallel approach opens new perspectives for flow solvers and the simulation of unsteady flows in large systems (including complex technological effects such as cooling devices or tip gaps) is now feasible.

4.2.1. Unsteady flow simulation in a whole turbomachine

Some authors have recently shown that a correct physical description of unsteady flows in a whole turbomachine can be obtained by using high-end computing platforms [41–43]. These works show that the flow can be simulated in the totality of the machine (including all blades and vanes and tip leakage flows) by means of an unsteady RANS method combined with a large computing power. Such achievements provide invaluable information to investigate the flow at design and off-design conditions in these systems.

Results presented hereafter are obtained with the flow solver elsA. Fig. 13 presents the unsteady flow during the development of rotating stall in a whole subsonic compressor stage [42]. This simulation has been performed in 2005 with the first generation of parallel computer (<8 computing cores). The solution was obtained after more than 40 days with a 30 M points grid.

In 2008, the use of the “second generation” of parallel computers led to a simulation of the unsteady flow in a 3-stage axial compressor [44,45] (representative of a high pressure compressor of modern turbojet engines). With a standard density, a 600 M points grid is necessary to represent the unsteady flow in the whole compressor. To give an order of magnitude, a simulation of the steady flow (RANS) requires only a 6.7 M points grid. Considering a quite coarse grid (134 M cells), the flow is simulated in less than one month at design operating conditions with 512 computing cores and at near stall conditions with 4096 computing cores (Fig. 14). Simulations are performed thanks to IBM Blue Gene/P (EDF) and IBM Blue Gene/L (CERFACS) platforms. The computational costs related to unsteady RANS and (steady) RANS approaches are summarized in Table 3. The cost ratio between the steady flow and the unsteady flow approaches is about 5000. It corresponds roughly to the power ratio between a micro-processor designed in 2010 and a processor design in 1992! Nevertheless the results provided by these unsteady flow simulations are very interesting. First it gives an overview of all the flow phenomena that develop in the machine (rotor–stator interactions, tip leakage flows, etc.). Then, combined with experimental investigations [44], it represents a very powerful tool to optimize and design innovative solutions to improve the compressor performance.

Table 3
Computing effort related to RANS and unsteady RANS approaches in a multistage compressor (IBM Blue Gene/P platform).

	RANS (coarse grid)	RANS (standard grid)	URANS (coarse grid)	URANS (standard grid)
Grid cells (millions)	1.5	6.7	134	600
Total CPU effort (hours)	55	220	280,000	1,250,000

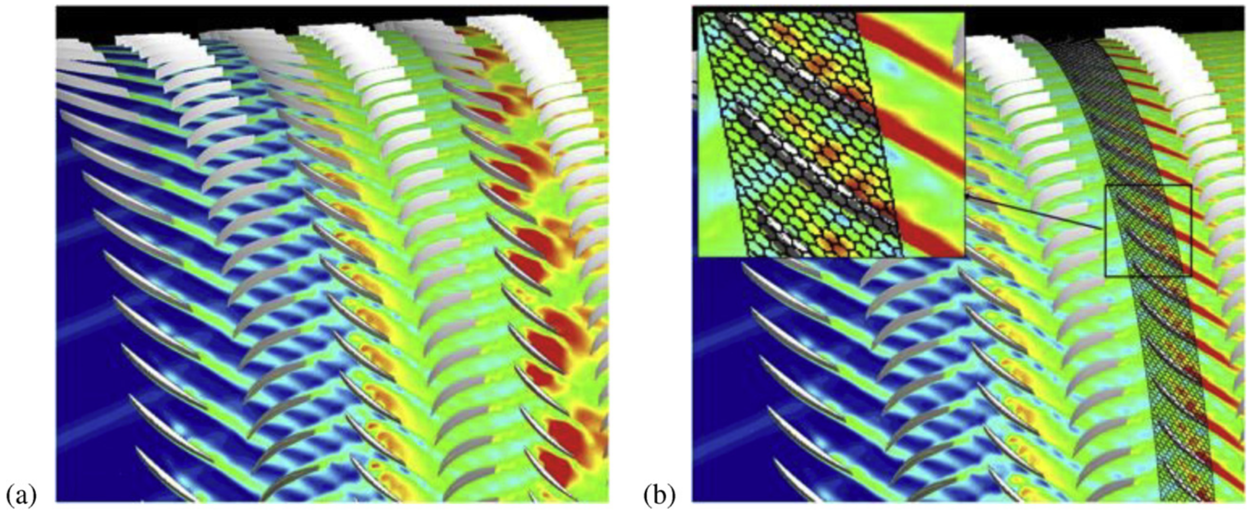


Fig. 15. Instantaneous flow field colored with entropy. (a) Reference configuration and (b) the same compressor with a casing treatment.

4.2.2. Solutions to improve the compressor performance

For example, Fig. 15 shows a comparison of the flow field between the reference configuration (i.e. the compressor shown in Fig. 14) and a configuration equipped with a passive control device designed at CERFACS. In this case, the casing treatment increases the stable operating range of the compressor by +40% without any efficiency loss. A HPC method was necessary to obtain the numerical data for the reference configuration (without control) and help in the design of the control device. It was also necessary to simulate the flow in the compressor with the control device. Indeed, HPC is not only useful to provide valuable data, but it is also an excellent mean to test advanced designs that are out-of-range with steady flow simulations.

4.2.3. High-fidelity unsteady flow simulations

As shown in the previous paragraph, flow simulations based on unsteady RANS methods give interesting and valuable data to estimate the mean performance (pressure ratio, efficiency, etc.) of a machine. However, flow solvers are still far to predict accurately local flows (secondary flows, boundary layer separations, laminar to turbulent transition etc.), especially at high Reynolds numbers such as encountered in most industrial configurations. For example, an estimation of the wall heat transfer (which is a design parameter for the high pressure turbine) remains out of range with RANS methods, mainly because turbulence plays a major role. In this context, unsteady flow simulations that solve a part of the turbulent spectrum (instead of modeling all the turbulent scales as it is done with RANS) emerge as a promising way for wall bounded flows at high Reynolds numbers. Both flow solvers (elsA and AVBP) are used to compute this specific flow configurations by means of LES.

Figs. 16 and 17 present the results of an LES for two typical turbo-machinery target applications. The configuration 1 is an inlet guide vane of a high pressure turbine, experimentally studied at the Von Karman Institute [46,47]. The Reynolds number based on chord is $\mathrm{Re} = 2.8 \times 10^6$ and the outlet Mach number is 0.79. The computational costs related to RANS, unsteady RANS and LES are compared in Table 4 (configuration 1). The cost of LES with elsA is moderate (about 100 times more costly when compared to RANS) because the implicit scheme (based on a Dual Time Stepping method) used for the time integration allows using large time steps. A comparison of instantaneous flow field of the density gradient is displayed in Fig. 16. As expected, the steady RANS approach cannot reproduce all the flow phenomena and only unsteady flow methods (URANS and LES) are able to compute large coherent structures that are experimentally observed in the turbine vane wakes (the so-called von Karman vortices). Moreover, the RANS approach predicts the development of a non-physical shock-wave on the suction side of the blade, indicating a difficulty to correctly estimate the boundary layer thickness. In this case, only a LES correctly estimates the vortex shedding frequency [48].

The configuration 2 is a highly loaded turbine guide vane, also studied at the Von Karman Institute [49]. This guide vane operates at high Mach and Reynolds numbers, close to operating conditions of a typical industrial configuration ($\mathrm{Re} = 1.1 \times 10^6$ and $M_{out} = 0.92$). At the investigated operating point, observations show a laminar to turbulent transition on

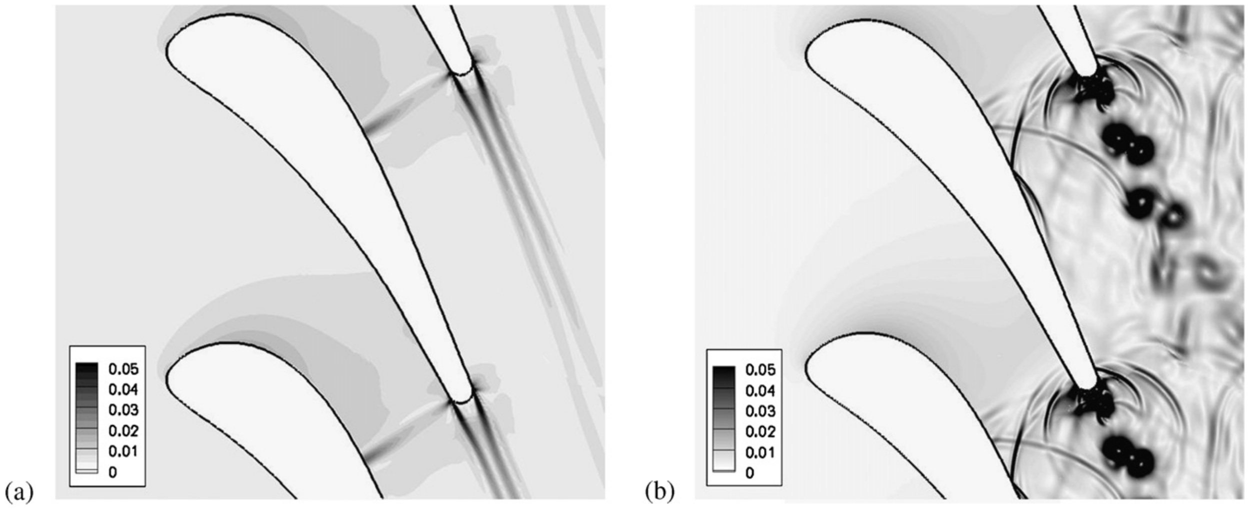


Fig. 16. Comparisons of the (a) RANS and (b) LES approaches. Instantaneous flow field of (grad ρ)/ρ in a turbine guide vane at high Reynolds number [48].

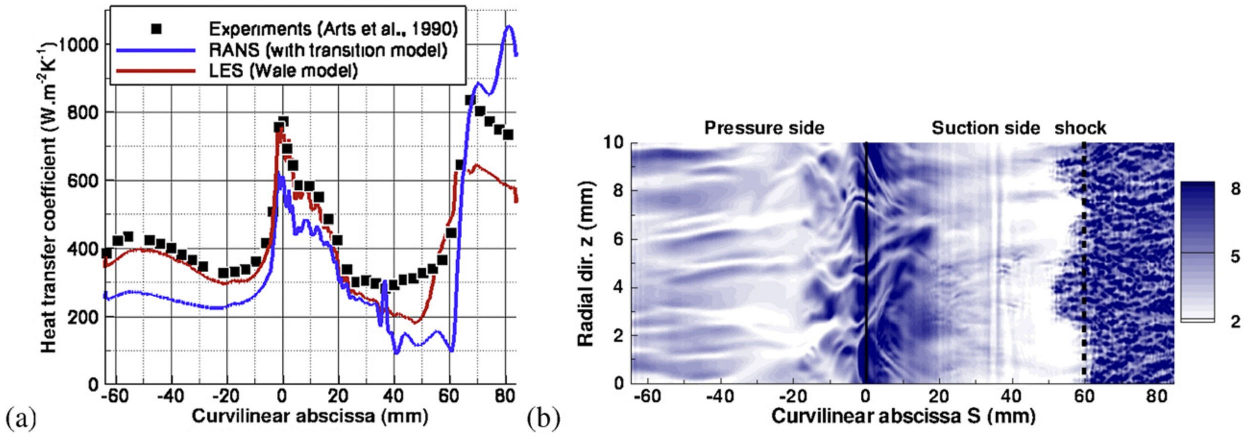


Fig. 17. (a) Comparisons of the RANS and LES approaches for the prediction of wall heat fluxes. (b) The instantaneous flow field colored with heat flux Q (W/m^2) points out the laminar to turbulent transition of the suction side boundary layer.

the blade suction side. The position of this phenomenon depends on the inlet turbulence intensity that is a very difficult parameter to set with flow solvers. As previously, the computational costs related to RANS and LES are compared in Table 4 (configuration 2). The main difference with configuration 1 is the size of the mesh grid (and thus the time step) required to compute highly challenging flows such as boundary layer transition. The consequence is that the LES is roughly 2,000 times more costly than RANS (elsA) and even 7,000 times more costly with AVBP. Fortunately, all LES are run with 512 PUs and the solution is obtained in less than one week. The predictions of RANS and LES for the wall heat transfer are presented in Fig. 17 and compared to experimental data. On the one hand, RANS is able to detect transition but it fails to estimate the correct level of heat transfer both on pressure and suction sides (error is close to 100% at some locations). On the other hand, LES successfully predicts the impact of inlet turbulent features and the wall heat transfer is estimated with a good accuracy ($< 10\%$) further emphasizing the potential of this method for such flows.

4.3. *Gas turbines*

Gas turbine combustion chambers are very complex devices involving highly turbulent flows, mixing of fuel with air and products of combustion issued by very thine reaction zones. It is a center piece of the gas turbine engines and allows the conversion of the fuel exothermic power which will then be used to produce mechanical power that is used to propel aircrafts or produce electricity. The combustion chamber is also the locus of creation of most of the combustion products including the pollutants. With the evolution of the recent regulations on emissions, new designs of combustion chambers are being developed and engineers face new challenges and require in-depth understanding of the flow and chemical processes that take place in the combustor. With the advent of new technological solutions to lower emissions and reduce fuel consumption, new problems arise including thermo-acoustic instabilities. In this type of operating condition, the com-

Table 4
Computing effort related to RANS, unsteady RANS and LES approaches in two inlet guide vanes [48] (SGI Altix platform).

	Configuration 1			
	RANS	URANS	LES (elsA)	LES (AVBP)
Grid cells (millions)	0.8	0.8	6.4	1.9
Total CPU effort (hours)	20	230	2000	1900
	Configuration 2			
	RANS		LES (elsA)	LES (AVBP)
Grid cells (millions)	0.7		29.7	29.3
Total CPU effort (hours)	20		35,000	132,000

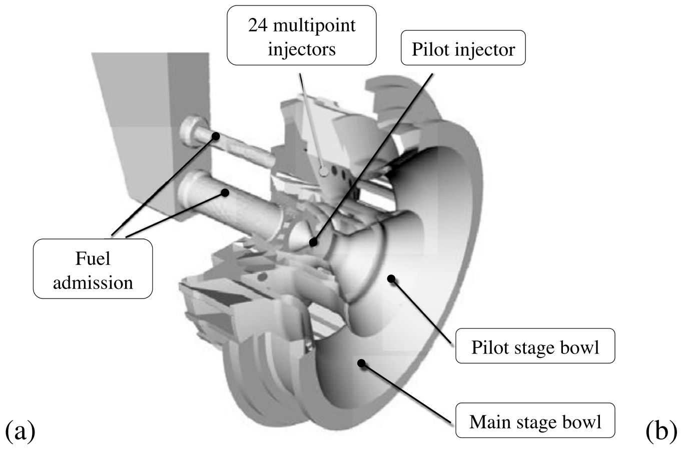


Fig. 18. Views of (a) a typical fuel injection system and (b) the combustion chamber of an aeronautical gas turbine engine.

bustion chamber enters a fully unsteady oscillating regime issued by the coupling between the system acoustic modes and combustion. In most circumstances, such operating conditions seriously reduce the life expectancy of the engine, increase pollutant emissions and in extreme cases break the engine. The main issue is that, as of today, such observations cannot be anticipated prior to the final testing of the entire engine.

Gas turbine combustion chambers are usually composed of multiple fuel injection systems whose geometry is highly complex (Fig. 18) and almost impossible to grid with structured meshes. Fully unstructured solvers are mandatory in this context, although they usually result in a serious overhead in the overall computational time when compared to structured codes (cf. Table 4 for typical comparisons). The primary objective of these devices is of course to inject the fuel in the co-flowing air coming from the compressor thereby allowing mixing prior to its combustion in the combustor. A secondary objective is to use the injection system to induce a rotation to the flow in the combustion chamber to enhance mixing and help positioning the flame at the given location. In this case, the injection system is also referred to as the swirler. One key point in the design of the next generation of gas turbines, is to understand the fluid dynamics of these devices and assess their effect on the thermo-acoustic behavior of the burner. Because of the strong links between mixing and the reaction which happens at very small scales (below the Kolmogorov turbulent scale) LES seems to be the only modeling approach that provides decent predictive capabilities in real applications [50]. It however remains very computer intensive and requires heavy developments and validation steps to help understanding thermo-acoustic instabilities. Such investigations are usually performed by means of experimental, theoretical and numerical developments among which LES of turbulent reacting flows has proven great potential by contributing to the fundamental understanding of turbulent reacting flows. It can also provide a mean of evaluating new designs before investing in costly manufacturing and experimental processes by simply offering a fully unsteady predictions of the real burner.

In the following, the use of AVBP on a HPC facility allows to produce a full picture of an azimuthal thermo-acoustic instability in a real helicopter combustion chamber. As detailed below the conjunction of HPC and advanced CFD codes provides data for detailed analyses of the instability as well as the assessment of easy-to-use and cheaper theoretical models to be used at the primary stages of an industrial burner. In this demonstration, each burner contains two co-annular counter-rotating swirlers. The fuel injectors are placed on the axis of the swirlers. To avoid uncertainties on boundary conditions (especially on inlet and outlet impedances) the chamber's casing is also computed. The computational domain starts after the inlet diffuser and ends at the throat of the high pressure stator where the choked flow is explicitly computed by the solver, avoiding uncertainties on acoustic impedances. The air and fuel inlets use non-reflective boundary conditions [51]. The cold air flowing in the casing feeds the combustion chamber through the swirlers, films and dilution holes. The reacting flow in the combustor of Fig. 19 is investigated using the AVBP compressible LES solver [52–59]: simulations are performed

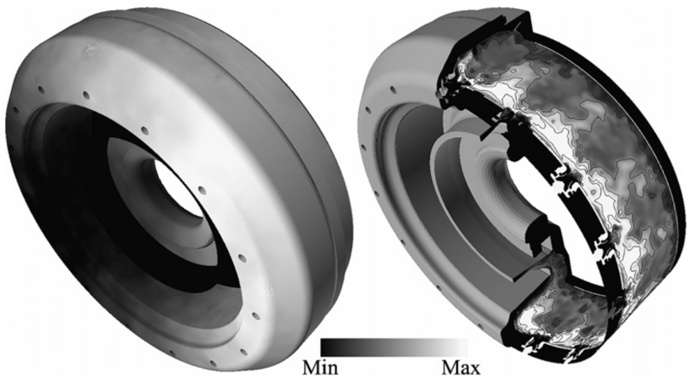


Fig. 19. Flow visualization. Left: pressure field on the combustor skin. Right: temperature field with temperature iso-contours on a cylindrical plane passing through the mid-radial plane of the combustor.

by computing first a single sector, duplicating the result N times around the turbine axis and letting the computation choose its most amplified oscillation mode. No azimuthal mode is added: the LES captures (or not) the oscillation modes of the combustor without any forcing [60]. In most cases, after a transient period of growth, the acoustic field exhibits azimuthal modes, leading flames to oscillate in this field, azimuthally but also longitudinally, causing periodic flashbacks inside the injectors, as illustrated by Fig. 19.

Even though the configuration is axisymmetric, the swirl imposed in each burner makes one rotation direction preferential, leading to the existence of a mean swirling velocity in the combustor. For the two rotating modes, this induces an important difference between the co-rotating mode (the azimuthal mode turning in the direction of the swirl induced by the injectors) called here the "+" mode and the counter-rotating mode called the "−" mode. To first order, the + mode turns at a velocity $c + U$ where U is the mean swirl velocity and c the mean sound speed in the chamber while the "−" mode turns at $c - U$. The mean swirling velocity U is small compared to the sound speed c: average typical swirl velocities of 10 m/s are observed in the LES. This allows to separate small and long time effects: the main high frequency azimuthal mode is observed at a high frequency of the order of $c/2\pi R$ and it is modulated by a lower frequency (of the order of $U/2\pi R$). To illustrate this point and facilitate the interpretation of LES results, a simple model is described here to express the pressure oscillations p' resulting from the combination of the + and − modes in an annulus of radius R where the period and the pulsation of the azimuthal mode without swirling flow are $T_{azi} = 2\pi R/c = 2\pi/\omega$ and $\omega = c/R$ respectively:

$$p' = \hat{p}e^{(-i\omega t)} = \left[A_+ e^{i(\theta - Ut/R)} + A_- e^{i(-\theta + Ut/R)}\right]e^{-i\omega t} \tag{1}$$

where θ is the angle measuring a point position along the azimuthal direction. The Ut/R terms are induced by the mean swirl convection at speed U. They change very slowly compared to the ωt term so that a structure can be defined for p' by observing it over a few periods of the short (acoustic) time: this structure then changes over long (convective) times. Typically, gas turbine experts observe standing wave modes (oscillating at hundreds of Hertz) where the pressure nodes are turning very slowly (one full rotation in a few minutes to a few hours). The period required for a complete rotation of the structure is simply $2\pi R/U$ or T_{azi}/M_a where $M_a = U/c$ is the Mach number of the swirling flow component. This observation makes the analysis of azimuthal modes more complicated: a standing mode (observed over a few periods) can exhibit a structure which rotates slowly (with the swirl velocity). Such a mode is not a "turning" mode where the pressure field rotates with the sound speed. When using LES, sampling over very long times is difficult so that observing a full rotation of such a structure is costly. In the present LES, the oscillations were computed for 130 ms corresponding to 100 cycles of the azimuthal mode but only slightly more than one rotation of the rotating structure. However, as soon as the rotation effect due to the mean swirl component has been identified, the structure can be studied over a few periods of the azimuthal mode, knowing that it will rotate with the mean swirl velocity U if one observes it for a long time. This rotation will not change the fundamental mode structure observed at short times. As an example, Fig. 20 shows the mode structure (modulus and phase of $\hat{p}_i$[1]) computed at short times using Eq. (1) at two instants t_i $(i = 1, 2)$

$$\hat{p}_i = A_+ e^{i(\theta - Ut_i/R)} + A_- e^{i(-\theta + Ut_i/R)} \tag{2}$$

where $t_1 = 0.143$ s and $t_2 = 0.153$ s all other quantities are retrieved from the fully unsteady LES predictions. Hence LES predictions and the simplest model agree within the mode keeping the same structure, very similar to a standing wave mode, but it has turned slowly between the two instants, shifting the pressure nodes and antinodes by approximately $\pi/2$.

For this previous diagnostics, the LES results on the 40 M cells grid are analyzed over a large number of cycles (50) to investigate the mode structure. Fig. 20 displays the mode structures obtained at two instants of the simulation $t_1 = 0.143$ s

[1] The exact expression for the phase ϕ_1 plotted in Fig. 20 is the difference between the argument of p' (which depends on t in Eq. (1)) and its value at a fixed θ_0 which is fixed here to $\theta_0 = 0$. The plotted phase is $\phi_1 = \arg(p') - \arg(p'(\theta = 0) = \arg(p') + \omega t = \arg(\hat{p})$.

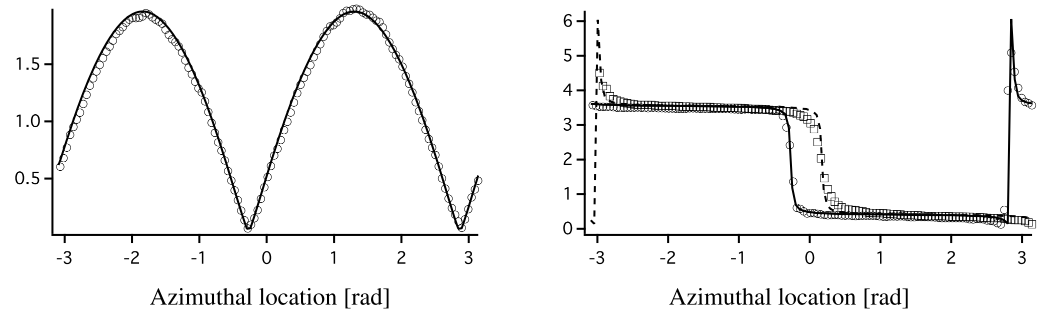


Fig. 20. LES results: pressure perturbations modulus (left) and phase (right) at two times $t_1 = 0.143$ s (circles) and $t_2 = 0.153$ s (squares). A fit using Eq. (1) is also added with $A_-/A_+ = 0.96$ (solid and dashed lines).

Table 5
Characteristics of unstructured meshes used for LES and corresponding CPU time on a BlueGene machine using 16,384 processors.

Grid	Number of points	Number of cells	Time step (s)	Elapsed time for one azimuthal cycle (hours)
Coarse	6,916,125	37,696,365	6×10^{-8}	2 h 10 min
Medium	16,466,145	93,147,720	3×10^{-8}	11 h 15 min
Fine	54,954,975	336,078,255	3×10^{-8}	33 h 30 min

and $t_2 = 0.153$ s. To construct these structures, seven cycles at 750 Hz are sufficient. Fig. 20 demonstrates that standing modes are observed and that these standing modes are rotating slowly. The rotation velocity is 44 rad/s corresponding to approximately 7.8 m/s in the combustion chamber. This value matches the levels of mean swirl velocities measured in the LES. The curves of Fig. 20 can be used to deduce A_+ and A_- (as defined in Eq. (1)). In the present case, the best fit (also shown in Fig. 20) corresponds to $A_-/A_+ = 0.96$. The conclusion is that LES in this case exhibits a standing wave mode. This mode is slowly turning because of the mean swirl produced in the chamber. No purely turning mode (for which either A_+ or A_- must be zero) is observed. Prior work on the issue [61] suggests that these standing modes are less likely to be found when the limit cycle is reached than rotating modes. The outcome of the present study is not consistent with recent developments who predicts standing waves in asymmetrical configurations but rotating mode in symmetrical ones. Further studies will require LES of non-axisymmetric cases to see whether the standing mode is replaced by a purely turning mode or not.

One key parameter that is known to be of great importance in assessing LES predictive capabilities [56] is the grid resolution and from a scientific point of view such effect needs to be assessed to ensure grid independence of the predictions for proper modeling assessment. Although such investigations are mandatory and are accessible for single burner LES's, producing such diagnostics is much more computer intensive for full burner computations. Thanks to the AVBP massively parallel capability and dedicated access to HPC facilities such sensitivity issues have been produced for the full burner. Table 5 presents the three meshes used to verify grid refinement effect. The objective of this test was to compare the mean flow obtained on the three grids but also the self-amplified modes. Note that CPU time depends not only on the number of grid points but also on the time step, as the LES code used throughout this study is explicit.

Runs were first started on the coarse mesh, the simulation was then continued using the medium and fine grids. The mean flow showed very limited changes. In particular, combustion behaves similarly on the three grids, not only in time, Fig. 21(a), but also in a mean statistical sense. Fig. 21(b) presents the probability density function (PDF) of the resolved equivalence ratio in reacting zones. PDFs obtained on all three meshes are almost identical. Most of the combustion takes place in lean premixed zones, with a strong peak at an equivalence ratio of $\phi = 0.35$. Another peak is located at $\phi = 1$ and stems from the presence of diffusion flamelets. A weaker peak appears around $\phi = 1.5$ and corresponds to rich premixed flames created close to the fuel injection. More interestingly for our study, the unsteady activity remained the same on the refined grid: the instability cycle continued at the same frequency and with the same amplitude showing that the results depend only weakly on the mesh.

5. Conclusions

This article has been devoted to programming aspects and the impact of HPC on CFD codes. The development and strategy necessary to perform numerical simulations on the powerful parallel computers have been presented. Two typical CFD codes have been investigated: a structured multi-block flow solver (elsA) and an unstructured flow solver (AVBP). Both codes are affected by the same requirements to run on HPC platforms. Due to the MPI-based strategy, a necessary step is to split the original configuration to define a new problem more adapted to parallel calculations (the work must be shared between all PUs). Mesh partitioning tools integrated in both flow solvers have been presented. Advantages and drawbacks

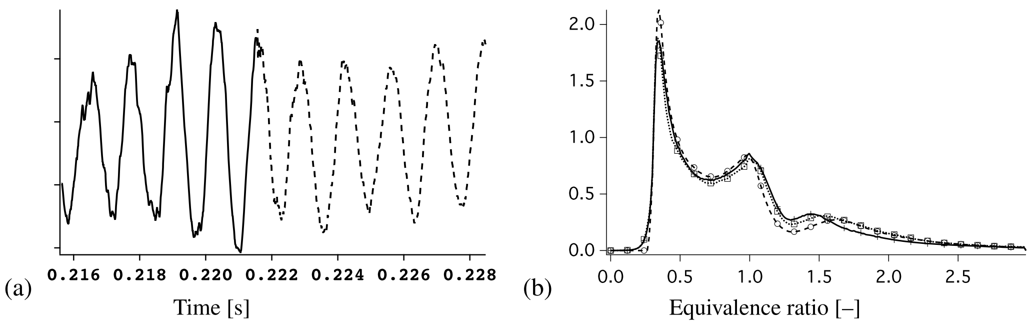


Fig. 21. Mesh independence tests: (a) pressure perturbations for two consecutive computations on two different grids and (b) probability density function of equivalence ratio for the three considered meshes: coarse (solid lines with crosses), medium (dotted lines with squares) and fine (dashed lines with circles).

of partitioning algorithms such as simple geometric methods (REB, RIB, etc.) or multilevel graph bisection methods (METIS) have been briefly discussed. However, both flow solvers do not face the same difficulties.

The mesh partitioning stage with unstructured grids (AVBP) is very flexible and usually leads to correct load balancing among all PUs. The computational cost of this partitioning stage can become very important. For example, a 44 M cell grid requires 25 min and 650 Mb of memory to be partitioned with the most efficient algorithm (k-way, METIS) on 4096 PUs. Other difficulties can appear when dealing with two-phase flow simulations (with particles seeding) for example, reducing the algorithm efficiency. For most applications, the method that provides the best parallel efficiency is the use of the RCM algorithm combined with a multilevel graph-partitioning algorithm (such as proposed in the METIS software). Compared to unstructured flow solvers, it is very difficult to achieve a good load balancing with structured multi-block meshes (elsA) but partitioning algorithms require very low computing resources in terms of memory and computational time. A 104 M cells grid is partitioned on 2048 PUs in less than 3 min and requires only 100 Mb of memory with the greedy algorithm.

Another important aspect is related to the time spent for communication between PUs. Both flow solvers use the MPI library. The working scheme of AVBP is based on a master-slave model that requires intensive communications between slaves and master processes, especially for I/O. To be more efficient, MPI communications are managed in AVBP through non-blocking calls. However, benchmarks showed that the implementation of MPI communications (collective or point-to-point calls) is critical for scalability on some computing platforms. A last difficulty is that non-blocking calls add "random" rounding errors to other sources such as machine precision. The consequence is that simulations considering the LES method exhibit a nondeterministic behavior. Any change in the code affecting the propagation of rounding errors will thus have a similar effect, implying that the validation of a LES code after modifications may only be based on statistical fields (comparing instantaneous solutions is thus not a proper validation method). A solution to minimize these rounding errors (at least for parallel computations) is to use blocking calls with a scheduler based on the graph theory (such as implemented in the flow solver elsA). This method gives good results in terms of communication time, even if some difficulties still exist for the treatment of global communications (non-coincident interfaces) that arise after the point-to-point communications (coincident interfaces). If all non-coincident interfaces are not correctly shared between PUs, this strategy leads to load balancing errors. To conclude, work is still necessary to improve the code performance on current (and future) computing platforms. First, a parallel implementation of I/O is necessary to avoid performance collapse on sensitive systems, especially for unsteady flow simulations. Second, implementation with MPI is complex and at very low level. A solution that uses both OpenMP (for communication inside nodes) and MPI (communication between nodes) is probably a way to explore for improving code scalability with multi-core nodes. Finally, improvement of mesh partitioning and load balancing tools will be necessary (dynamic/automatic load balancing). For AVBP, works focus on the improvement of the communication methods between master and slave computing cores. For elsA, current works focus on the communication strategies (suppression of collective calls for non-coincident interfaces) and on the mesh partitioning algorithms (integration of a communication graph). Work will also be necessary to implement massively parallel capacities for specific functionalities such as the Chimera method. The impact of ghost cells on HPC capabilities should be better considered for the development of future flow solvers. Finally, flow solvers face new difficulties such as the problem of memory bandwidth, which is expected to be crucial with the advent of the last super-computer generation that uses more and more cores by computing node. According to recent publications, adding more cores per chip can slow data-intensive applications, even when computing in a sequential mode. Future high-performance computers have been tested at Sandia National Laboratories with computing nodes containing 8–64 cores that are expected to be the next industrial standard. Results were disappointing with conventional architecture since no performance improvement was observed beyond 8 cores. This fact is related to the so-called memory wall that represents the growing disparity between the CPU and data transfer speeds (memory access is too slow). Because of limited memory bandwidth and memory-management schemes (not always adapted to super-computers), performance tends to decline with nodes integrating more cores, especially for data-intensive programs such as CFD flow solvers. The key for solving this problem is probably to obtain better memory integration to improve memory bandwidth. Other architectures such as graphic processors (GPU) are also a potential solution to increase the computational power available for

CFD flow solvers. However, such architectures would impose major modifications in the code, more adapted programming language and optimized compilers.

To finish and despite the numerous challenges that are still present for HPC and CFD use on the next generation of super-computers, a series of industrial problems have been presented and for which use of the state-of-the-art CFD tools definitely opens new perspectives to the aeronautical industry.

Acknowledgements

We thank Onera, Insitut francais du pétrole (IFP) and development partners for their sustained effort to make elsA and AVBP successful projects. Authors are also grateful to teams of CERFACS involved in the management and maintenance of computing platforms (in particular the CSG group). Furthermore, we acknowledge all people that contributed to this paper by means of discussions, support, direct help or corrections. Special thanks to industrial and research partners for supporting code developments and permission for publishing results. In particular, we thank Airbus, Snecma and Safran for collaborative work around elsA and AVBP projects. Finally, we acknowledge EDF, Météo-France, GENCI-CINES and computing centres around the world for providing computational resources.

Some of the numerical results presented in this document were obtained by use of resources of the Argonne Leadership Computing Facility at Argonne National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under contract DE-AC02-06CH11357. GENCI (Grand Equipement National de Calcul Intensif) and CINES (Centre Informatique National de l'Enseignement Superieur) also provided part of the computing power necessary for simulations under grant FAC-2554. Finally, gas turbine results are partly supported by the ANR under the project "SIMTUR" ANR-07-CIS7-008.

References

[1] D.E. Keyes, D.K. Kaushik, B.F. Smith, Prospects for CFD on petaflops systems, Technical Report TRT-97-73, Institute for Computer Applications in Science and, Engineering, 1997.
[2] N. Gourdain, L.Y.M. Gicquel, M. Montagnac, O. Vermorel, M. Gazaix, G. Staffelbach, M. Garcia, J.-F. Boussuge, T. Poinsot, High performance parallel computing of flows in complex geometries – Part 1: Methods, Comput. Sci. Discov. 2 (November 2009) 015003 (26 pp.).
[3] N. Gourdain, L.Y.M. Gicquel, G. Staffelbach, O. Vermorel, F. Duchaine, J.-F. Boussuge, T. Poinsot, High performance parallel computing of flows in complex geometries – Part 2: Applications, Comput. Sci. Discov. 2 (November 2009) 015004 (28 pp.).
[4] M.J. Flynn, Some computer organizations and their effectiveness, IEEE Trans. Comput. C-21 (9) (1972) 948–960.
[5] L. Cambier, M. Gazaix, elsA: an efficient object-oriented solution to CFD complexity, in: 40th AIAA Aerospace Science Meeting and Exhibit, 2002.
[6] L. Cambier, J.P. Veuillot, Status of the elsA CFD software for flow simulation and multidisciplinary applications, in: 46th AIAA Aerospace Science Meeting and Exhibit, AIAA paper 664, 2008.
[7] L. Davidson, S.H. Peng, Hybrid LES-RANS modeling: a one-equation SGS model combined with a k–ω model for predicting recirculating flows, Internat. J. Numer. Methods Fluids 43 (9) (2003) 1003–1018.
[8] J. Clique, R. Houdeville, D. Arnal, Application of laminar-turbulent transition criteria in Navier–Stokes computations, in: 46th AIAA Aerospace Science Meeting and Exhibit, Reno, USA, 2007.
[9] S. Deck, Numerical simulation of transonic buffet over a supercritical airfoil, AIAA J. 43 (2005) 1556–1566.
[10] P.R. Spalart, W.-H. Jou, M. Stretlets, S.R. Allmaras, Comments on the feasibility of LES for wings and on the hybrid RANS/LES approach, advances in DNS/LES, in: Proceedings of the First AFSOR International Conference on DNS/LES, 1997.
[11] J. Smagorinsky, General circulation experiments with the primitive equations: 1. The basic experiment, Mon. Weather Rev. 91 (1963) 99–164.
[12] G. Fillola, M.-C. Le Pape, M. Montagnac, Numerical simulations around wing control surfaces, in: 24th ICAS Meeting, Yokohama, Japan, 2004.
[13] R. Meakin, The chimera method of simulation for unsteady three-dimensional viscous flows, Comput. Fluid Dynam. Rev. (1995) 70–86.
[14] P.L. Roe, Approximate Riemann solvers, parameter vectors and difference schemes, J. Comput. Phys. 43 (1981) 357–372.
[15] M.S. Liou, A sequel to AUSM: AUSM+, J. Comput. Phys. 129 (1996) 364–382.
[16] S. Yoon, A. Jameson, An LU-SSOR scheme for the Euler and Navier–Stokes equations, in: AIAA 25th Aerospace Sciences Meeting,AIAA paper 0600, 1987.
[17] A. Jameson, Time dependent calculations using multigrid, with applications to unsteady flows past airfoils and wings, in: AIAA Computational Fluid Dynamics Conference, 1991.
[18] M. García, Interpolates parallele de solutions, Projet ANR CAMPAS, Livrable L3 CR/CFD/07/149, CERFACS, Toulouse, France, 2007.
[19] P. Moin, S.V. Apte, Large-eddy simulation of realistic gas turbine combustors, Am. Inst. Aeronaut. Astronaut. J. 44 (4) (2006) 698–708.
[20] P.D. Lax, B. Wendroff, Systems of conservation laws, Comm. Pure Appl. Math. 13 (1960) 217–237.
[21] J. Donea, Taylor–Galerkin method for convective transport problems, Internat. J. Numer. Methods Fluids 20 (1) (1984) 101–119.
[22] O. Colin, M. Rudgyard, Development of high-order Taylor–Galerkin schemes for unsteady calculations, J. Comput. Phys. 162 (2) (2000) 338–371.
[23] C.W. Hirt, A.A. Amsden, J.L. Cook, An arbitrary Lagrangian–Eulerian computing method for all flow speeds, J. Comput. Phys. 131 (4) (1974) 371–385.
[24] G. Karypis, V. Kumar, A fast and high quality multilevel scheme for partitioning irregular graphs, SIAM J. Sci. Comput. 20 (1) (1998) 359–392.
[25] M.J. Berger, S.H. Bokhari, A partitioning strategy for nonuniform problems on multiprocessors, IEEE Trans. Comput. C-36 (5) (1987) 570–580.
[26] A. Ytterström, A tool partitioning structured multiblock meshes for parallel computational mechanics, Int. J. High Perform. Comput. Appl. 11 (1997) 336–343.
[27] R.D. William, Performance of dynamic load balancing algorithms for unstructured mesh calculations, Concurrency, Pract. Exp. 3 (1991) 451–481.
[28] H.D. Simon, Partitioning of unstructured problems for parallel processing, Comput. Systems Engrg. 2 (3) (1991) 135–148.
[29] J.G. Lewis, The Gibbs–Poole–Stockmeyer and Gibbs–King algorithms for reordering sparse matrices, ACM Trans. Math. Software 8 (2) (1982) 190–194.
[30] G. Karypis, V. Kumar, Multilevel algorithms for multi-constraint graph partitioning, Technical Report 98-019, University of Minnesota, Department of Computer Science/Army HPC Research Center, 1998.
[31] B. Hendrickson, R. Leland, A multilevel algorithm for partitioning graphs, Technical Report SAND93-1301, Sandia National Laboratories, Albuquerque, NM, 1993.
[32] J.-M. Senoner, M. García, S. Mendez, G. Staffelbach, O. Vermorel, T. Poinsot, Growth of rounding errors and repetitivity of Large–Eddy simulations, AIAA J. 46 (7) (2008) 1773–1781.
[33] M. García, Analysis of precision differences observed for the AVBP code, Technical Report TR/CFD/03/84, CERFACS, Toulouse, France, 2003.

[34] C. Rossow, N. Kroll, High performance computing serves aerospace engineering: Opportunities for next generation product development, in: 46th AIAA Aerospace Science Meeting and Exhibit, Reno, USA, 2008.
[35] J. Delbove, Unsteady simulations for flutter prediction, in: Proceedings of the Third International Conference on CFD – ICCFD3, 2006, pp. 205–210.
[36] F. Sicot, G. Puigt, M. Montagnac, Block–Jacobi implicit algorithms for the time spectral method, AIAA J. 46 (2008) 3080–3089.
[37] A. Gopinath, E. van der Weide, J.J. Alonso, A. Jameson, K. Ekici, K.C. Hall, Three-dimensional unsteady multi-stage turbomachinery simulations using the harmonic balance technique, in: 45th AIAA Aerospace Science Meeting and Exhibit, Reno, USA, 2007.
[38] V. Brunei, S. Deck, Zonal-detached eddy simulation of transonic buffet on a civil aircraft type configuration, in: 46th AIAA Aerospace Science Meeting and Exhibit, Reno, USA, 2008.
[39] P.R. Spalart, S.R. Allmaras, A one equation turbulence model for aerodynamic flows, in: 30th AIAA Aerospace Science Meeting and Exhibit, AIAA paper 92-0439, Reno, USA, 1992.
[40] A.F. Pouangué, H. Deniau, F. Sicot, P. Sagaut, Curvilinear finite-volume schemes using high-order compact interpolation, J. Comput. Phys. 229 (2010) 5090–5122.
[41] M.D. Hathaway, G. Herrick, J. Chen, R. Webster, Time accurate unsteady simulation of the stall inception process in the compression system of a US army helicopter gas turbine engine, in: DoD Users Group Conference, 2004, pp. 182–193.
[42] N. Gourdain, S. Burguburu, F. Leboeuf, G.-J. Michon, Simulation of rotating stall in a whole stage of an axial compressor, J. Comput. Fluids 39 (9) (2010) 1644–1655.
[43] E. van der Weide, G. Kalitzin, J. Schluter, J.J. Alonso, Unsteady turbomachinery computations using massively parallel platforms, in: 44th AIAA Aerospace Sciences Meeting and Exhibit, 2006.
[44] N. Gourdain, X. Ottavy, A. Vouillarmet, Experimental and numerical investigation of unsteady flows in a high speed three stages compressor, in: 8th European Turbomachinery Conference, 2008.
[45] N. Gourdain, M. Montagnac, F. Wlassow, M. Gazaix, High performance computing to simulate large scale industrial flows in multistage compressors, Int. J. High Perform. Comput. Appl. 24 (4) (2010) 429–443.
[46] C. Sieverding, H. Richard, J.-M. Desse, Turbine blade trailing edge flow characteristics at high subsonic outlet Mach number, Trans. ASME 125 (2003) 298–309.
[47] C. Sieverding, D. Ottolia, C. Bagnera, A. Comadoro, J.-F. de Brouckaert, J.-M. Desse, Unsteady turbine blade wake characteristics, J. Turbomach. 126 (2004) 551–559.
[48] T. Leonard, F. Duchaine, N. Gourdain, L.Y.M. Gicquel, Steady, unsteady Reynolds averaged Navier–Stokes and large eddy simulations of a turbine blade at high subsonic outlet Mach number, in: ASME Turbo Expo, 2010.
[49] T. Arts, M. Lambert de Rouvroit, A.W. Rutherford, Aero-thermal investigation of a highly loaded transonic linear turbine guide vane cascade, Technical Note 174, Von Karman Institute, 1990.
[50] T. Poinsot, D. Veynante, Theoretical and Numerical Combustion, 2nd edition, R.T. Edwards, 2005.
[51] T. Poinsot, S. Lele, Boundary conditions for direct simulations of compressible viscous flows, J. Comput. Phys. 101 (1) (1992) 104–129.
[52] A. Roux, L.Y.M. Gicquel, S. Reichstadt, N. Bertier, G. Staffelbach, F. Vuillot, T. Poinsot, Analysis of unsteady reacting flows and impact of chemistry description in large eddy simulations of side-dump ramjet combustors, Combust. & Flame 157 (2010) 176–191.
[53] A. Roux, S. Reichstadt, N. Bertier, L.Y.M. Gicquel, F. Vuillot, T. Poinsot, Comparison of numerical methods and combustion models for LES of a ramjet, C. R. Mecanique 337 (6–7) (2009) 352–361.
[54] G. Boudier, N. Lamarque, G. Staffelbach, L.Y.M. Gicquel, T. Poinsot, Thermo-acoustic stability of a helicopter gas turbine combustor using large-eddy simulations, Int. J. Aeroacoust. 8 (1) (2009) 69–94.
[55] P. Wolf, G. Staffelbach, A. Roux, L. Gicquel, T. Poinsot, V. Moureau, Massively parallel LES of azimuthal thermo-acoustic instabilities in annular gas turbines, C. R. Mecanique 337 (6–7) (2009) 385–394.
[56] G. Boudier, L.Y.M. Gicquel, T. Poinsot, D. Bissières, C. Bérat, Effect of mesh resolution on large eddy simulation of reacting flows in complex geometry combustors, Combust. Flame 155 (1–2) (2008) 196–214.
[57] A. Roux, L.Y.M. Gicquel, Y. Sommerer, T.J. Poinsot, Large eddy simulation of mean and oscillating flow in a side-dump ramjet combustor, Combust. Flame 152 (1–2) (2007) 154–176.
[58] C. Prière, L.Y.M. Gicquel, P. Gajan, A. Strzelecki, T. Poinsot, C. Bérat, Experimental and numerical studies of dilution systems for low emission combustors, Am. Inst. Aeronaut. Astronaut. J. 43 (8) (2005) 1753–1766.
[59] C. Prière, L.Y.M. Gicquel, A. Kaufmann, W. Krebs, T. Poinsot, LES of mixing enhancement: LES predictions of mixing enhancement for jets in cross-flows, J. Turb. 5 (2004) 1–30.
[60] G. Staffelbach, L.Y.M. Gicquel, G. Boudier, T. Poinsot, Large eddy simulation of self-excited azimuthal modes in annular combustors, Proc. Combust. Inst. 32 (2009) 2909–2916.
[61] B. Schuermans, C. Paschereit, P. Monkiewitz, Non-linear combustion instabilities in annular gas-turbine combustors, AIAA paper 2006-0549, 2006.