

A Method To Accelerate LES Explicit Solvers Using Local Time-Stepping

Olivier Esnault, Matthieu Boileau, Ronan Vicquelin, Benoit Fiorina, Olivier Gicquel

▶ **To cite this version:**

Olivier Esnault, Matthieu Boileau, Ronan Vicquelin, Benoit Fiorina, Olivier Gicquel. A Method To Accelerate LES Explicit Solvers Using Local Time-Stepping. 48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, Jan 2010, Orlando, United States. pp.AIAA-2010-123, 10.2514/6.2010-123 . hal-00472692

HAL Id: hal-00472692
https://hal.science/hal-00472692

Submitted on 13 Apr 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A Method To Accelerate LES Explicit Solvers Using Local Time-Stepping

Olivier Esnault[*], Matthieu Boileau[†], Ronan Vicquelin[‡], Benoît Fiorina[§] and Olivier Gicquel[¶]

Laboratoire EM2C, CNRS, Ecole Centrale Paris, Grande voie des vignes, 92295 Châtenay-Malabry, France.

In practical flow configurations, a large disparities of geometrical length scales are often encountered. Inside a combustor, for example, the ratio between the diameter of the injection holes and the size of the entire combustion chamber may present several orders of magnitude. When considering an explicit solver for fully compressible Navier-Stokes equations, the global time step is constrained through a CFL-like condition by the size of the smallest cells in the overall computational domain. Local refinement of the injector leads to an inhomogeneous mesh and the former restriction drastically alters the overall solver efficiency. A new local time-stepping (LTS) method is proposed to address this issue. The domain is divided into subgrids composed of cells that have similar sizes. Flow equations are simultaneously advanced on each subgrid which have a local time step adapted to satisfy the local CFL condition. The accuracy of the method has been verified on a simple convection case using a test code. The method has also been implemented in a large eddy simulation (LES) explicit solver and successfully tested for an acoustic wave propagation. It has been finally used in the two-dimensional large eddy simulation of a turbulent jet.

I. Introduction

Thanks to the progress of turbulent combustion modeling and to the increase in computing power, today numerical simulations are now part of the design process of aeronautical turbine combustors. Main challenges when designing a gas turbine combustor are to control flame ignition and extinction as well as combustion instabilities that may cause the destruction of the engine. Predicting these phenomena requires to take into account flow unsteady effects and their interaction with the reaction zone. The Large Eddy Simulation (LES) technique, combined with significant improvements in the field of combustion modeling, have provided researchers with the tools needed to study a wide variety of turbulent reacting flows. Although LES has the capability to accurately predict the key phenomena, some studies remain beyond the capacities of existing CFD tools. For example, simulations of a turbine combustor are usually limited to a single sector. However, to accurately predict ignition, reignition, quenching and instabilities taking place in a combustor, computations of the full chamber are required. Recently, LES of reacting flows in full combustion chambers of gas turbines have been successfully performed with the AVBP code on several massivelly parallel machines.[1–5] However, such calculations are still too expensive to be used in industry.

So, although the increasing efficiency of parallel computing, computational time remains an important issue. The CPU time directly relies on restrictions imposed by the numerical schemes stability. When considering an explicit solver for fully compressible Navier-Stokes equations, the time step is constrained through an accoustic CFL-like condition by the smallest cells in the overall computational domain. For combustion chamber simulation, local refinement of the fuel injector leads to an inhomogeneous mesh and the former restriction affects the overall solver efficiency. One way to address this issue is to use local time-

[*]Postdoctoral researcher, CNRS.
[†]Research engineer, CNRS. Corresponding author: matthieu.boileau@em2c.ecp.fr
[‡]PhD student, Pôle CHENE, GDF-Suez, Direction de la Recherche, 93211 La Plaine Saint-Denis, France.
[§]Assistant professor, Ecole Centrale Paris.
[¶]Professor, Ecole Centrale Paris.

stepping (LTS) methods. They allow to define local time steps whose values depend on the corresponding local mesh properties. The aforementioned severe CFL restrictions may then be overcome.

Existing LTS methods mainly group in two classes: hierarchical and non-hierarchical methods. The category of hierarchical methods counts AMR-like strategies and adaptative multiscale schemes. AMR methods were introduced in the 80s by Berger and co-workers[6,7] and are widely used and suitable for parallel solvers using structured meshes. They are based on a hierarchical tree of overlapping grids. Adaptative multiscale schemes[8,9] are multiresolution techniques in the line of Harten's work.[10] On the contrary, other LTS methods are rather concerned by interfacing neighboring grids that use different time steps. In this category, one can find some domain decomposition methods for treating evolution problems that base the coupling between subdomains on Schwarz waveform relaxation methods.[11,12] However, these mathematical studies are not ready for application to non-academic multidimensional problems. Still in this second category, one can find methods[13] in the line of a pioneering work by Osher and Sanders.[14] Their work on one-dimensional scalar conservation laws has been extended in[15] that show the possibility for their LTS method to reach a second-order in time. Other LTS methods have also been designed for moving mesh methods.[16] Most of papers use for tests some academic fluid flow problems, but one also find several publications from the electromagnetic community.[17]

The objective of this work is to develop a new LTS method – called DECCOUP – adapted to explicit and parallel solvers. The principle is to balance the computing load by splitting the computational domain into subgrids which size depends on the local time step. To the authors' knowledge, DECCOUP is the first LTS method suited for massively parallel simulations except AMR-like methods.[6,7] The DECCOUP technique is developed in the scope of the ANR-CIS SIMTUR project. Final objectives are to implement it and to use it in the AVBP code.[18] AVBP is an unstructured code that solves the fully compressible LES equations for reacting flows.[19] Using a cell-vertex formulation, it can work with both structured and unstructured grids which makes it easily applicable to complex geometries.[20] Centered spatial schemes and explicit time-advancement are used to control numerical dissipation.[21] Several numerical schemes are available ; among them are a second order finite-volume Lax-Wendroff scheme and a third order finite-element TTGC scheme.

Making an estimate based on the case illustrated on Fig. 1, one can broadly evaluate the speedup resulting from DECCOUP. Consider a mesh involving:

- n_{small} cells from blocks advanced with δt_{small},
- $n_{large} = R_n \cdot n_{small}$ larger cells from blocks advanced with $\delta t_{large} = R_{\delta t} \cdot \delta t_{small}$,
- $n_{buffer} = \lambda R_n \cdot n_{small}$ cells from buffer blocks.

The role of these buffer blocks will be described later in the document. They are intermediate blocks whose cells are calculated twice: first taken as small cells, then taken as large ones.

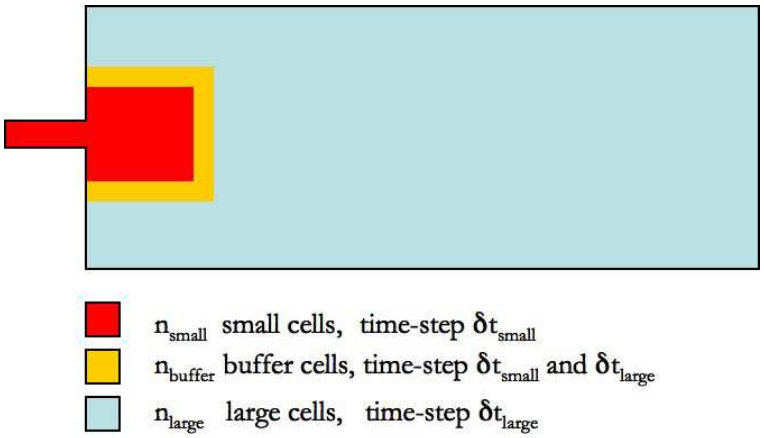


Figure 1. Cell sizes distribution in a combustion chamber mesh.

Without any LTS method, the overall load to be handled to reach the physical time t_{sim} in the simulation writes:

$$Load = (n_{small} + n_{buffer} + n_{large}) \cdot \frac{t_{sim}}{\delta t_{small}} \tag{1}$$

while using DECCOUP it reduces to the following:

$$Load_{LTS} = (n_{small} + n_{buffer}) \cdot \frac{t_{sim}}{\delta t_{small}} + (n_{buffer} + n_{large}) \cdot \frac{t_{sim}}{\delta t_{large}} \tag{2}$$

Thus, assuming that computational time is proportional to the load, the following speedup is expected:

$$Speedup = \frac{Load}{Load_{LTS}} = \frac{1 + (1+\lambda)R_n}{(1+\lambda R_n) + (1+\lambda)\frac{R_n}{R_{\delta t}}} \tag{3}$$

Assuming $\lambda \ll 1$, the former expression underlines that values of *Speedup* ranging from 1 (in the case corresponding to $R_n = R_{\delta t} = 1$) to $\delta t_{large}/\delta t_{small}$ (in the ideal case obtained when $R_n \to 0$) can be expected. Intermediate cases are illustrated on the graph from Fig. 2. In practice, this speedup can be obtained on parallel computations for an optimal load balancing. Indeed, Eq. 3 is satisfied if the load is balanced on processors such as:

$$R_{\delta t}\frac{n_{small}}{N_{P,small}} = \frac{n_{large}}{N_{P,large}} = (1 + R_{\delta t})\frac{n_{buffer}}{N_{P,buffer}}, \tag{4}$$

where $N_{P,small}$, $N_{P,buffer}$, $N_{P,large}$ are the number of processors handling small, buffer and large cells, respectively.

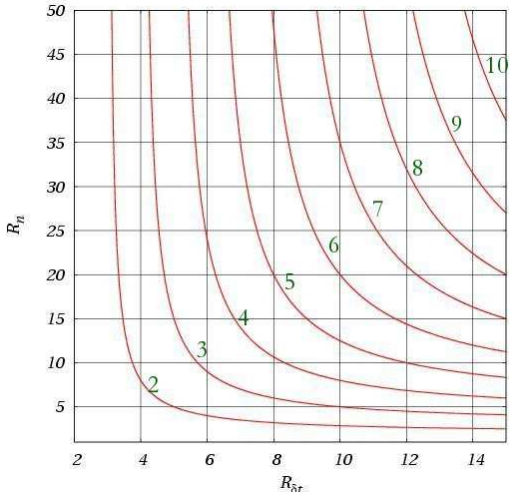


Figure 2. ***Speedup*** **isolines in the** $R_{\delta t} - R_n$ **plane.**

The document is organized as follow. The strategy of DECCOUP is presented in section II. A test configuration and corresponding results are described in section III. Results from the implementation of DECCOUP in the LES code AVBP are presented in section IV. Finally, section V concludes on the present work.

II. The DECCOUP method

In most Navier-Stokes explicit and parallel solvers, a domain decomposition method is used to split the overall computational domain into several blocks (noted B_j). Each block is then affected to one processor. Generally, the initial solver runs over all blocks using a unique global time step, determined through a reduction operation (each processor evaluates the time step value allowed by the CFL criterion over its cells and a reduction operation keeps the most restrictive value). DECCOUP is designed to allow each block B_j to keep its proper minimum time step δt_j. The method is described using 1D examples for clarity but the generalisation to multidimensional problems is straightforward.

Consider a given spatial numerical scheme in an explicit context. To be applied at bordering cells of a block, the corresponding stencil generally requires data from neighbouring blocks at the same time t_n. These data can be made available by using ghost cells surrounding each block and are updated from communications between processors at the begining of each new iteration. When using a unique time step for each block, no approximation is needed to fill in ghost cells since all blocks are simultaneously advanced in time. On the contrary, when using LTS, two neighbouring blocks may not use the same time steps to advance the solution. Consequently, the data required to fill in ghost cells may not be available at each discrete time values. Ghost cells values are then approximated between two communications steps (see Fig. 3).

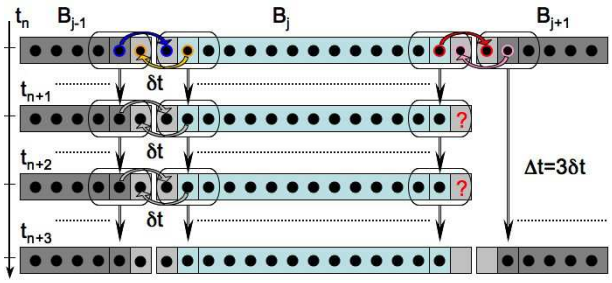


Figure 3. Approximations are required when using LTS : blocks B_{j-1} and B_j use the same time step δt while B_{j+1} uses a larger time step $\Delta t = 3\delta t$. Values to fill in the ghost cells in B_j at its interface with B_{j+1} are missing so they must be approximated.

To illustrate DECCOUP, let's consider a set of N_B blocks $(B_j)_{j=1,N_B}$ resulting from domain decomposition. Each block is handled by a processor P_j that uses a time step δt_j. Local time steps have first to respect the local stability criteria. Secondly, they are choosen such as a global time step $\Delta t = m_j \cdot \delta t_j, \forall j \in [1, N_B]$, where m_j are integers, exists. Times corresponding to the global time step are noted $(t_n)_{n=1,N_{\Delta t}}$, where $N_{\Delta t}$ is the total number of iterations using Δt. By construction, t_n is common to all block solutions. Intermediate times between t_n and t_{n+1} using the local time step δt_j are expressed by $t = t_n + \nu \cdot \delta t_j$, $\nu = 1, m_j$. The computed B_j block solution vector at time t is noted $w(B_j, t)$. Ghost cells solution values belonging to B_j and shared by B_j and B_k are noted $w(B_j \cap B_k, t)$.

At a given interface between two neighbouring one-dimensional blocks, two cases can be distinguished: either blocks using the same time step value so no LTS handling is required or blocks using different time steps so the DECCOUP strategy applies. In that case, DECCOUP requires a buffer block to be inserted at the LTS interface during a DECCOUP-adapted domain decomposition phase. This preprocessing step is detailed further. To illustrate the method, a sequence of 3 blocks B_{small}, B_{buffer} and B_{large} is considered. Their corresponding parameters are all noted with the appropriate subscript (see Fig. 4). In the present case, $\delta t_{small} < \delta t_{large}$ and $m_{small} \cdot \delta t_{small} = m_{large} \cdot \delta t_{large}$.

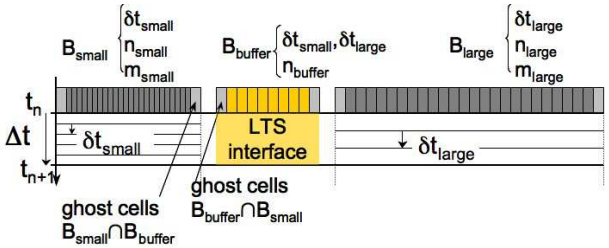


Figure 4. Within DECCOUP, a buffer block is used at LTS interfaces between B_{small} and B_{large}.

The buffer block B_{buffer} is dedicated to advance the solution twice from t_n to t_{n+1}. Grid spacing on this block is designed in such a way that stability criterion is respected for both time step values δt_{small} and δt_{large}. The solution is computed firstly through m_{small} subcycle iterations with $\delta t_{small} = \Delta t / m_{small}$, that lead to a first solution $w(B_{buffer}, t_n + m_{small} \cdot \delta t_{small})$, and secondly using $\delta t_{large} = \Delta t / m_{large}$, leading to a second solution $w(B_{buffer}, t_n + m_{large} \cdot \delta t_{large})$. If δt_{small} is used to advance the buffer block, interior block data are available to compute $w(B_{small}, t_{n+1})$ but ghost cells data $w(B_{large} \cap B_{buffer}, t_n + \nu \cdot \delta t_{large}), \nu = 1, m_{large}$

are missing. They are roughly approximated by:

$$w(B_{large} \cap B_{buffer}, t_n + \nu \cdot \delta t_{large})) = w(B_{large} \cap B_{buffer}, t_n), \forall \nu = 1, m_{large}\,. \tag{5}$$

Here DECCOUP takes advantage of the explicit nature of the solver. Indeed, given an explicit solver that use a stencil requiring s_{left} (resp. s_{right}) cells on the left (resp. right) of the current point, any perturbation introduced at time t_n cannot propagate through the grid of a distance exceeding – per time step δt_j – s_{left} cells to the right and s_{right} cells to the left[a]. Consequently, errors introduced by the previous approximation (Eq. 5) propagate over a known number of cells from the $B_{large} \cap B_{buffer}$ ghost cells. This number of cells is equal to $s_{left} \cdot m_j$ at time t_{n+1}. The corresponding solution $w(B_{buffer}, t_n + m_{small} \cdot \delta t_{small})$ is then polluted by errors near the interface with B_{large} but remains unaffected in the vicinity of the interface with B_{small} (see Fig. 5).

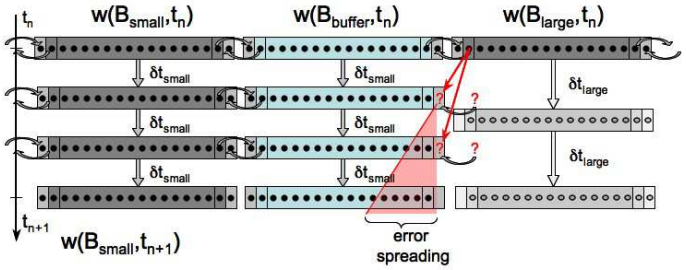


Figure 5. When using δt_{small} to advance B_{buffer} in time, enough data are available to obtain $w(B_{small}, t_{n+1})$ and a partially valid solution on B_{buffer} at the same time.

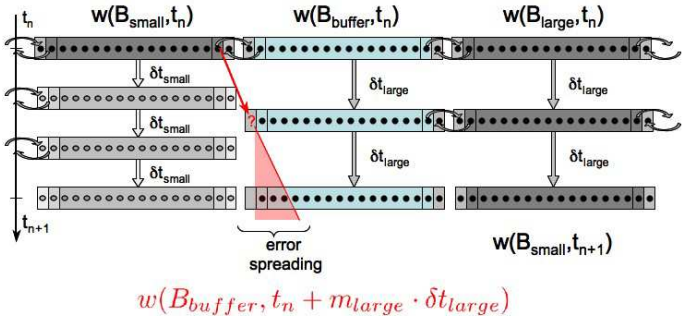


Figure 6. When using δt_{large} to advance B_{buffer} in time, enough data are available to obtain $w(B_{large}, t_{n+1})$ and a partially valid solution on B_{buffer} at the same time.

When δt_{large} is used to advance B_{buffer}, the same difficulty occurs at the interface $B_{buffer} \cap B_{small}$ (see Fig. 6): $w(B_{large}, t_{n+1})$ is obtained, as well as the polluted solution $w(B_{buffer}, t_n + m_{large} \cdot \delta t_{large})$. Finally, the way of interfacing grids using different time steps whithin DECCOUP consists in using an intermediate grid. Advancing the solution with δt_{small} allows to locally preserve a continuity in the accuracy of the solution through the interface between B_{small} and B_{buffer} but introduces errors at the other interface, whereas the use of the biggest time step locally preserve a continuity in the accuracy of the solution through the interface between B_{small} and B_{buffer}. On each cell of B_{buffer}, two estimations of the solution $w(B_{buffer}, t_{n+1})$ are available. Both of them are polluted by errors but, if B_{buffer} is long enough, cells affected by errors in the first computation do not overlap with the polluted ones during the second computation. It is then possible to built $w(B_{buffer}, t_{n+1})$ as a smooth transition between the two polluted solutions that keeps only the unaffected part of them and filters all wrong values. This is done using the following equation:

$$w(x_i, t_{n+1}) = (1 - p(x_i)) \cdot w(x_i, t_n + m_{small} \cdot \delta t_{small}) + p(x_i) \cdot w(x_i, t_n + m_{large} \cdot \delta t_{large})\,, \tag{6}$$

where $x_i \in B_{buffer}$ and the function $p(x)$ can be an hyperbolic tangent located at the center x_0 of B_{buffer} $\left(p(x) = \frac{1}{2}\left(1 + \tanh\left(A(x - x_0)\right)\right)\right)$, so as to keep the not polluted part of each solution.

[a]This is true if the spatial scheme is explicit and cannot be applied if implicit schemes such as compact schemes[22] are used.

III. Evaluation of the method in a test code

In order to evaluate the accuracy of the DECCOUP method, calculations are performed using a test code which solves the one-dimensional convection equation: $\frac{\partial \rho}{\partial t} + U \frac{\partial \rho}{\partial x} = 0$ with $U = cst > 0$. Periodic boundary conditions are used and the temporal integration is explicit. The initial solution is a Gaussian wave in a domain of length L_B. This preliminary study focuses only on the accuracy of the underlying schemes and not on the speedup of the method. The overall domain of calculation is splitted into 6 blocks, involving two successive LTS interfaces with the following alternance: B_{small}, B_{buffer}, B_{large}, B_{large}, B_{buffer}, B_{small}. Thus, the wave crosses LTS-interfaces twice a period, once from small to large time steps , and a second time from large to small time steps. Here, a value of $m_{large} = 1$ is considered, leading to $\Delta t = \delta t_{large}$, and $m_{small} = 8$. The number of iterations is set so as to end the simulation at $t_{sim} = k \cdot (L_B/U)$, i.e. until the wave has traveled k times through the overall domain.

The code developed for testing DECCOUP is used in 8 configurations, depending on the choices for the scheme (HOSRK4/WENO5RK3), for the use of LTS (with/without) and for the nature of the mesh (uniform/non-uniform). Main input parameters are:

- a level L_i to define the size of mesh cells,
- the number of subcycles m_{small},
- the wave speed U,
- a CFL number
- the number of flow-through times k.

A set of meshes L_0 to L_4 is used to compare the solution for various levels of refinement (the width of the Gaussian wave being fixed once for all). These meshes are built on a root one (L_0), called level 0, and they are defined recursively such that L_{n+1} is obtained by splitting every cells of L_n in two identical cells (see Fig. 7). Level 0 characteristics, namely the cell size δx_{large} on B_{large} (and B_{buffer}) as well as cells numbers n_{small}, n_{large} and n_{buffer}, are hardcoded. The cell size δx_{small} on B_{small} is equal to δx_{large} for the uniform mesh and equal to $\delta x_{small} = \delta x_{large}/m_{small}$ in the non-uniform case. For level L_i, δx values are divided by 2^i while cells numbers are multiplicated by the same factor, insuring that block lengths are the same for all levels to allow comparisons. In the non-uniform case, B_{buffer} and B_{large} remains uniform but B_{small} is meshed with cell sizes ranging from $\delta x_{large}/m_{small}$ to δx_{large} through an arithmetic progression. In terms of spatial resolution, L_0 is such that the solution goes from almost 0 to 1 over 5 cells (and over 10, 20, 40 and 80 cells for L_1, L_2, L_3 and L_4, respectively). Concerning time steps, δt_{large} is set using the CFL and wave speed U ($\delta t_{large} = CFL \cdot \delta x_{large}/U$). When LTS is used $\delta t_{small} = \delta t_{large}/m_{small}$. If not, δt_{small} is equal to δt_{large}.

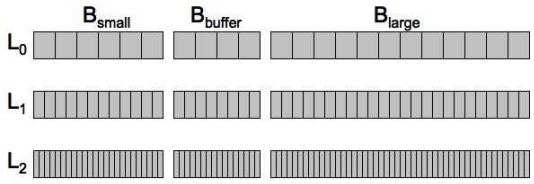


Figure 7. First refinement levels of a schematic LTS interface, in the uniform case, and the Gaussian wave profile with a fixed-width. Here $m_{small} = 3$.

Two schemes are retained: a high order finite differences scheme[23] having a third order in space even on non-uniform grids coupled to a RK4 time integration method (HOSRK4), as well as a fifth order finite volumes scheme of the WENO type[24] coupled to a TVD-RK3 time integration method (WENO5RK3). This WENO scheme is also able to handle non-uniform meshes. The run parameters are: $U = 1$, $CFL = 0.5$, $\delta t_{large}/\delta t_{small} = 8$, where CFL is based on δt_{small} for the uniform mesh cases. A solution with DECCOUP for the convection equation is shown on Fig. 8. After the wave has travelled 10 times through the periodic domain, the DECCOUP solution perfectly matches the theoretical one. For refinement levels L_0 to L_4, the accuracy of the solution is compared when DECCOUP is used and when it is not. The error is defined as the

difference – based on a L_∞-norm – between the computational solution and the exact one (i.e. the Gaussian wave translated by $U \cdot (t - t_0)$), following the equation:

$$\text{error}_{L\infty}(t_n) \quad = \quad \frac{\max_i |\rho_i^n - \rho(x_i, t_n)|}{\max_i |\rho(x_i, t_n)|} \tag{7}$$

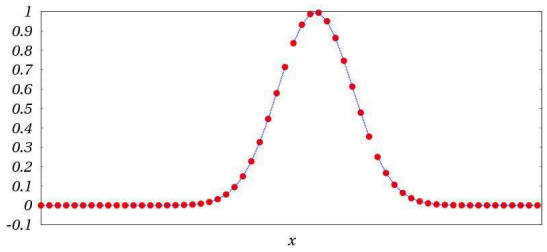


Figure 8. Comparison between the solution obtained by DECCOUP with the WENO scheme (•) and the initial solution ($\cdots$) after exactly 10 periods. The mesh is uniform (level L_1), $U = 1$, $CFL = 0.5$ (based on δt_{small}), $\delta t_{large}/\delta t_{small} = 8$.

Three main results appear on Fig. 9 and 10: i/ DECCOUP do not necessary introduces a loss of accuracy; ii/ the impact of the underlying scheme on performances of DECCOUP is significant; and iii/ the method does not seem to be sensitive to the non-uniformity of the mesh. For HOSRK4 scheme, the accuracy of the solution appears to be either better when DECCOUP is used, in the case of a uniform mesh or unchanged, in the non-uniform case. The performance of DECCOUP also appears to be dependent of the scheme, since results with the WENO scheme are not as accurate as those corresponding to the HOSRK4 scheme.

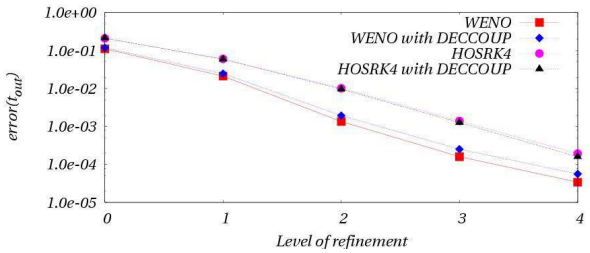


Figure 9. Error as a function of the refinement level L_i for uniform meshes.

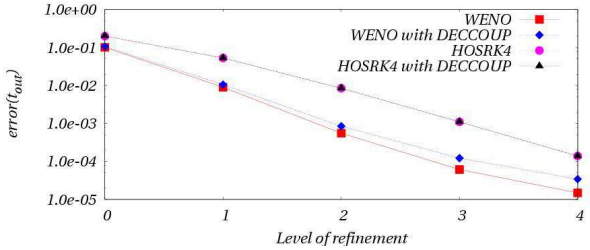


Figure 10. Error as a function of the refinement level L_i for non-uniform meshes.

On Fig. 11, scheme orders with and without DECCOUP have been computed for both schemes. The HOSRK4 scheme appears to be very efficient for all considered levels of refinement, showing a better scheme order with DECCOUP than without. As noticed previously, the method is less accurate when used with the WENO scheme. In that case, DECCOUP introduces a slight loss in the scheme order.

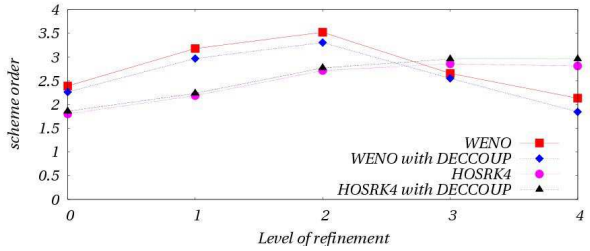


Figure 11. Order of the method as a function of the refinement level L_i for HOSRK4 and WENO schemes.

IV. Implementation in a LES solver

DECCOUP has been implemented in the AVBP solver. Results are presented here concerning an acoustic wave propagating in a 1-D periodic domain (see Fig. 12). The mesh used is non-uniform and contains 220 small cells, 4400 large cells and 400 buffer cells. The following parameters are fixed (see section I): $R_{\delta t} = 10$, $R_n = 20$ and $\lambda = 1/11$.

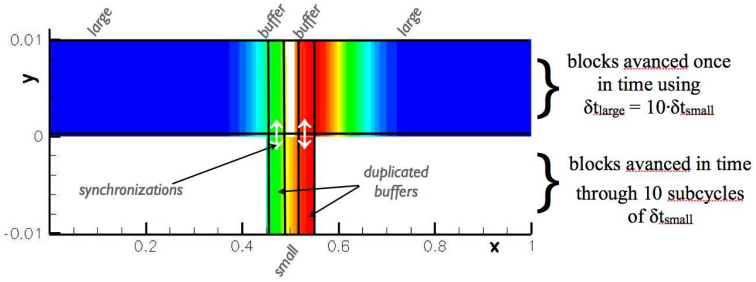


Figure 12. Decomposition of the AVBP computational domain for the acoustic wave test case.

Figure 13 shows that a very good agreement is obtained between the AVBP solution calculated with DECCOUP and the one calculated without. Another result of interest is the speedup obtained with DECCOUP. When no LTS method is used, the CPU time required to reach a given physical time t_{sim} on 5 processors is 4.95 times longer than when LTS is used. Actually, the LTS method requires 2 additional processors to handle the duplicated buffer blocks. The CPU cost due to these additional processors is included in the calculation of the speedup using a correction factor of 5/7. The corrected speedup obtained with DECCOUP is thus equal to 3.53 while the theoretical value given by Eq. 3 (using the same correction) is 3.25.

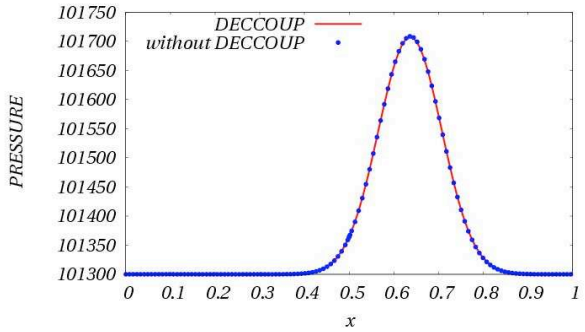


Figure 13. Comparison between the AVBP solution (pressure field in Pa) obtained with (—) and without (•) DECCOUP after 20 000 time iterations.

DECCOUP has also been tested in the 2D large eddy simulation of a plane air jet. This jet is characterized by an inlet Reynolds number of $Re_0 = 30\,000$, an inlet Mach number of $M_0 = 0.1$ and a surrounding co-flow

at Mach number $0.057M_0$. The subgrid stress tensor is modelled by the filtered Smagorinsky model.[25] A third order finite-element TTGC scheme[21] using a two-step time integration is applied to an unstructured triangular grid (see Fig. 14a). A passive scalar is injected at the inlet to visualize the turbulent jet mixing. The aim of this test case is to see how DECCOUP may affect of the flow solution in a turbulent configuration. A simple decomposition in three transverse blocks is performed as shown by Fig. 14a. According to the cell size ratio between large and small cells blocks, a ratio of $\delta t_{large}/\delta t_{small} = 10$ is used. The CFL number based on the smallest cell is equal to 0.7. At some time t_0 of the jet simulation, the DECCOUP method is started and performed during $t_{sim} = 49$ flow times (Fig. 14b) corresponding to 30 000 time iterations of δt_{small}. The scalar field concentration at $t_0 + t_{sim}$ is compared to the solution obtained at the same physical time without using DECCOUP (Fig. 14c). Very small differences in the scalar transport can be seen between the basic and the DECCOUP solutions. These differences may be due to the growth of rounding errors in parallel large eddy simulations, an effect that has been identified by Senoner *et al.*[26] Figure 14b shows that the solution is continuous and regular inside, outside and at the boundaries of the buffer block where the exchange of the flow solution takes place between the small and large time steps regions.

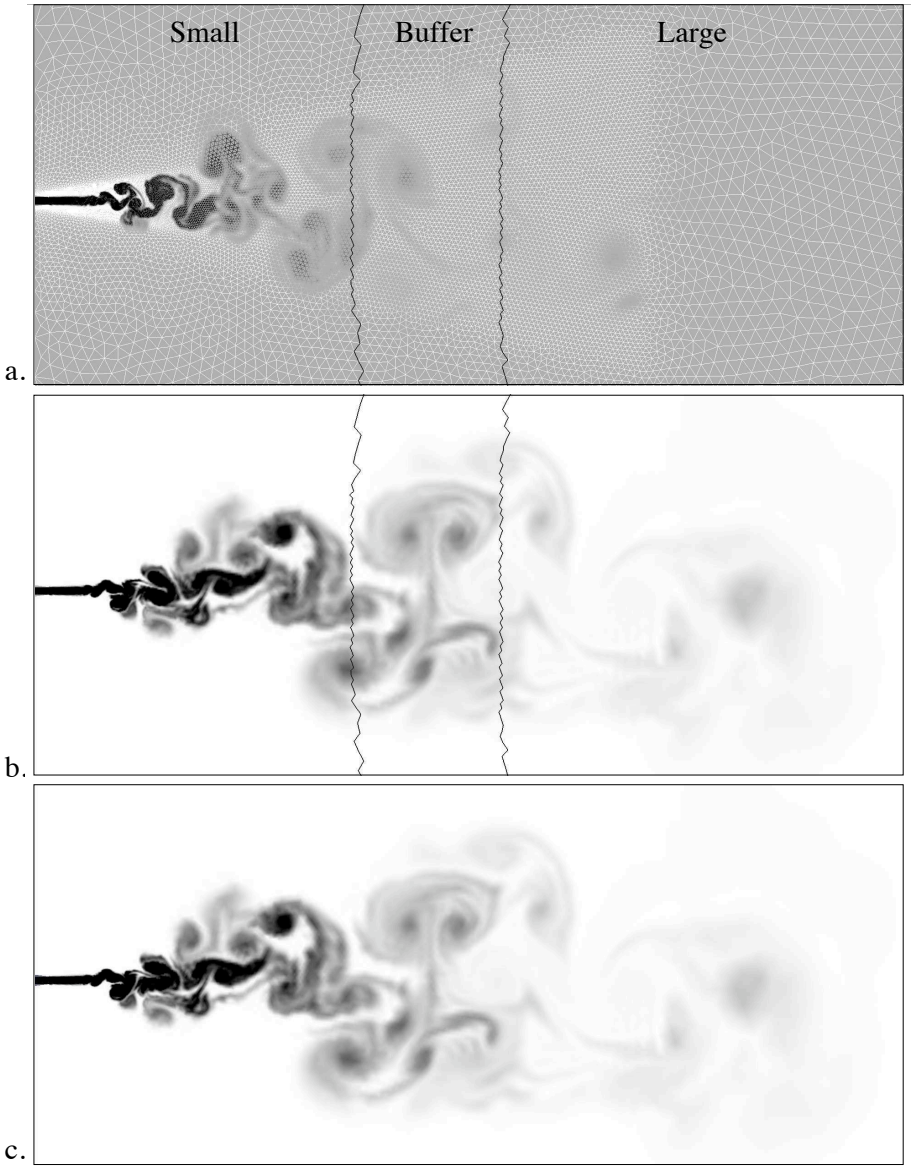


Figure 14. Snapshots of passive scalar fields at initial time t_0 (Fig. a.) and current time $t_0 + t_sim$ where $t_{sim} = 49$ flow times (corresponding to 30 000 time iterations). The flow at $t_0 + t_sim$ have been calculated with DECCOUP (Fig. b.) and without DECCOUP (Fig. c.). Figure a. shows the mesh grid (zoom) and a three-block decomposition: a small-step block, a buffer block and a large-step block. The gray scale has been saturated at 42% of the inlet value.

V. Conclusion

An acceleration method for LES explicit solvers based on local time-stepping has been presented. This method called DECCOUP uses a buffer block to handle interfaces between each LTS zone. DECCOUP has first been evaluated using a test code in a one-dimensional configuration. Results prove that the method has a minor incidence on the accuracy of the solution, whatever the nature of the underlying scheme. DECCOUP has also been implemented in the LES code AVBP. First academic tests shows that DECCOUP keeps the accuracy of the original numerical method while providing a speedup which is in agreement with the theoretical value. A 2D large eddy simulation test case has shown that DECCOUP is also accurate in turbulent configurations.

In practical applications, DECCOUP will be particularly efficient when only a few small cells penalize the overall calculation. Moreover, two requirements have to be fullfilled by the domain decomposition for DECCOUP: i/ the size of buffer blocks has to be large enough to allow to built a valid solution from non-polluted parts of intermediate solutions, and ii/ a relation between the small-size, large-size and buffer cells have to be verified to ensure an efficient load balancing. This load balancing can be achieved using a multi criteria mesh decomposition algorithm.

VI. Acknowledgments

This work was founded by the Agence Nationale de la Recherche through the scientific program ANR-CIS SIMTUR. Calculations were performed using HPC resources from GENCI-IDRIS (Grant 2009 - i2009080164). The authors gratefully acknowledge the assistance of G. Staffelbach from CERFACS.

References

[1]Allsopp, N., Tabary, A., Follows, J., Vezolle, P., Hennecke, M., Reddy, H., Ishibashi, F., Sosa, C., Paolini, M., Prakash, S., Quintero, D., and Lascu, O., *Unfolding the IBM eServer Blue Gene Solution*, IBM Redbooks, First ed., 2005, pp. 277–285.

[2]Boileau, M., Mossa, J.-B., Cuenot, B., Poinsot, T., Bissières, D., and Bérat, C., "Toward LES of an ignition sequence in a full helicopter combustor," *1st Workshop INCA*, SNECMA, Villaroche, France, 2005, pp. 27–34.

[3]Boileau, M., Staffelbach, G., Cuenot, B., Poinsot, T., and Bérat, C., "LES of an ignition sequence in a gas turbine engine," *Combustion and Flame*, Vol. 154, No. 1–2, 2008, pp. 2–22.

[4]Wolf, P., Staffelbach, G., Gicquel, L., Poinsot, T., Moureau, V., and Bérat, C., "Massively parallel LES of azimuthal thermo-acoustic instabilities in annular gas turbines," *2ème colloque INCA*, CORIA, Rouen, France, 2008.

[5]Staffelbach, G., Gicquel, L., Boudier, G., and Poinsot, T., "Large Eddy Simulation of self excited azimuthal modes in annular combustors," *Proceedings of the Combustion Institute*, Vol. 32, No. 2, 2009, pp. 2909–2916.

[6]Berger, M. and Oliger, J., "Adaptive mesh refinement for hyperbolic partial differential equations," *Journal of computational Physics*, Vol. 53, No. 3, 1984, pp. 484–512.

[7]Berger, M. and Colella, P., "Local adaptive mesh refinement for shock hydrodynamics," *Journal of computational Physics*, Vol. 82, No. 1, 1989, pp. 64–84.

[8]Coquel, F., Nguyen, Q., Postel, M., and Tran, Q., "Local time stepping for implicit-explicit methods on time varying grids," *Numerical Mathematics and Advanced Applications: Proceedings of Enumath 2007, the 7th European Conference on Numerical Mathematics and Advanced Applications, Graz, Austria, September 2007*, Springer, 2008, p. 257.

[9]Müller, S. and Stiriba, Y., "Fully adaptive multiscale schemes for conservation laws employing locally varying time stepping," *Journal of Scientific Computing*, Vol. 30, No. 3, 2007, pp. 493–531.

[10]Harten, A., "Multiresolution algorithms for the numerical solution of hyperbolic conservation laws," *Communications on Pure and Applied Mathematics*, Vol. 48, No. 12, 1995, pp. 1305–1342.

[11]Gander, M., Halpern, L., and Nataf, F., "Optimal Schwarz waveform relaxation for the one dimensional wave equation," *SIAM Journal on Numerical Analysis*, Vol. 41, No. 5, 2004, pp. 1643–1681.

[12]Martin, V., "Méthode de décomposition de domaine et de couplage pour des problèmes d'évolution." *Annales mathématiques Blaise Pascal*, Vol. 9, Université Blaise Pascal, Département de mathématiques, 2002, pp. 299–312.

[13]Crossley, A. and Wright, N., "Time accurate local time stepping for the unsteady shallow water equations," *International Journal for Numerical Methods in Fluids*, Vol. 48, No. 7, 2005.

[14]Osher, S. and Sanders, R., "Numerical approximations to nonlinear conservation laws with locally varying time and space grids," *Mathematics of computation*, 1983, pp. 321–336.

[15]Dawson, C. and Kirby, R., "High resolution schemes for conservation laws with locally varying time steps," *SIAM Journal on Scientific Computing*, Vol. 22, No. 6, 2001, pp. 2256–2284.

[16]Tan, Z., Zhang, Z., Huang, Y., and Tang, T., "Moving mesh methods with locally varying time steps," *Journal of Computational Physics*, Vol. 200, No. 1, 2004, pp. 347–367.

[17]Fumeaux, C., Baumann, D., Leuchtmann, P., and Vahldieck, R., "A generalized local time-step scheme for efficient FVTD simulations in strongly inhomogeneous meshes," *IEEE Transactions on Microwave Theory and Techniques*, Vol. 52, No. 3, 2004, pp. 1067–1076.

[18]AVBP, "AVBP Code: www.cerfacs.fr/cfd/avbp_code.php and www.cerfacs.fr/cfd/CFDPublications.html," 2009.

[19]Moureau, V., Lartigue, G., Sommerer, Y., Angelberger, C., Colin, O., and Poinsot, T., "Numerical methods for unsteady compressible multi-component reacting flows on fixed and moving grids," *Journal of Computational Physics*, Vol. 202, No. 2, 2005, pp. 710–736.

[20]Schonfeld, T. and Rudgyard, M., "A cell-vertex approach to local mesh refinement for the 3-D Euler equations," *AIAA, Aerospace Sciences Meeting and Exhibit, 32 nd, Reno, NV*, 1994.

[21]Colin, O. and Rudgyard, M., "Development of high-order Taylor-Galerkin schemes for LES," *Journal of Computational Physics*, Vol. 162, No. 2, 2000, pp. 338–371.

[22]Lele, S., "Compact finite difference schemes with spectral-like resolution," *Journal of Computational Physics*, Vol. 103, No. 1, 1992, pp. 16–42.

[23]Liu, J., Pope, G., and Sepehrnoori, K., "A high-resolution finite-difference scheme for nonuniform grids," *Applied Mathematical Modelling*, Vol. 19, No. 3, 1995, pp. 162–172.

[24]Wang, R., Feng, H., and Spiteri, R., "Observations on the fifth-order WENO method with non-uniform meshes," *Applied Mathematics and Computation*, Vol. 196, No. 1, 2008, pp. 433–447.

[25]Ducros, F., Comte, P., and Lesieur, M., "Large-eddy simulation of transition to turbulence in a boundary layer developing spatially over a flat plate," *Journal of Fluid Mechanics*, Vol. 326, 1996, pp. 1–36.

[26]Senoner, J., García, M., Mendez, S., Staffelbach, G., Vermorel, O., and Poinsot, T., "Growth of Rounding Errors and Repetitivity of Large-Eddy Simulations," *AIAA Journal*, Vol. 46, No. 7, 2008, pp. 1773–1781.