RELIABLE OVERNIGHT INDUSTRIAL LES: CHALLENGES AND LIMITATIONS. APPLICATION TO CSP TECHNOLOGIES

***A. Alsalti-Baldellou*[1,2], *G. Colomer*[1], *J. A. Hopman*[1], *X. Álvarez-Farré*[3], *A. Gorobets*[4], *F. X. Trias*[1], *C. D. Pérez-Segarra*[1] *and A. Oliva*[1]**

[1] ***Heat and Mass Transfer Technological Center, Universitat Politècnica de Catalunya – BarcelonaTech, Carrer de Colom 11, 08222 Terrassa (Barcelona), Spain***

[2] ***Termo Fluids SL, Carrer de Magí Colet 8, 08204 Sabadell (Barcelona), Spain***

[3] ***High-Performance Computing Team, SURF, Science Park 140, 1098 XG Amsterdam, The Netherlands***

[4] ***Keldysh Institute of Applied Mathematics, 4A, Miusskaya Sq., Moscow 125047, Russia***

adel@termofluids.com

Abstract

Preserving the operators' symmetries at the discrete level is crucial to enable reliable DNS and LES simulations of turbulent flows. Moreover, real-world applications demand robust and stable numerical methods suitable for complex geometries. In this regard, this work describes TFA, our novel in-house code, which relies on a symmetry-preserving discretisation for unstructured collocated grids that, apart from being virtually free of artificial dissipation, is shown to be unconditionally stable. To ensure cross-platform portability, the implementation of such a discretisation relies on a minimal set of algebraic kernels. Doing this poses challenges that need to be addressed, like the low arithmetic intensity of the sparse matrix-vector product, the reformulation of boundary conditions and flux limiters, or the efficient computation of eigenbounds to determine the time-step. With the aim of analysing the advantages and disadvantages of this "algebraic" approach, a comparison with OpenFOAM, probably the most widespread open-source CFD code, will be made. Finally, a relevant case from the CSP industry will be presented in order to assess the feasibility of overnight industrial simulations.

1 Introduction

In the last decades, CFD has become a standard design tool in many fields, such as the automotive, aeronautical, and renewable energy industries. The driving force behind this is the development of numerical techniques in conjunction with the progress of high-performance computing (HPC) systems. However, progress is nowadays hindered by its legacy from the 90-2000s. The reasons are two-fold. Firstly, the design of digital processors constantly evolves to overcome limitations and bottlenecks. The formerly compute-bound nature of processors led to compute-centric programming languages and simulation codes. However, raw computing power grows faster than the memory bandwidth, turning around the problem and leading to increasingly complex memory hierarchies that make optimising traditional applications a cumbersome task. Moreover, new parallel programming languages emerged to target modern hardware, *e.g.*, CUDA, HIP and oneAPI, and porting algorithms and applications has become restrictive. For this reason, designing more abstract modular codes kept gaining interest. For instance, the PyFR framework (see Witherden et al. (2014)) is mostly based on matrix multiplications and point-wise operations. Another examples are Kokkos and RAJA libraries (see Carter Edwards et al. (2014); Beckingsale et al. (2019)), which provide an abstraction layer aiming at providing performance portability. Remarkably enough, implementing modular codes allow linking to standard libraries optimized for particular architectures, in addition to specialized in-house implementations. Examples of this include cuSPARSE and clSPARSE (see Bell et al. (2009); Greathouse et al. (2016)).

Secondly, legacy numerical methods chosen to solve (quasi)steady problems using RANS models are inappropriate for more accurate (and expensive) techniques such as large-eddy simulation (LES) or direct numerical simulation (DNS). We aim to interlace these two pillars with the final goal of enabling LES and DNS of industrial applications to be efficiently carried out on modern HPC systems while keeping codes easy to port, optimise, and maintain. In this regard, TermoFluids Algebraic (TFA), our novel in-house code, adopts the fully-conservative discretisation for collocated unstructured grids proposed in F. X. Trias et al. (2014): it constitutes a very robust approach that

can be easily implemented in existing codes such as OpenFOAM in E. Komen et al. (2021).

The main recognised limitations of LES in the industry are their computational cost and the wall-clock simulation time. Thanks to the above-explained advent of new computational architectures, the former is becoming less and less critical, whereas the latter is still the most limiting factor precluding LES from being routinely used in the industry. For that to be possible, the consensus is that widespread adoption in the industry begins when a run can be carried out overnight, see, for instance, Löhner et al. (2021). Namely, the industry is governed by shortening design cycles, faster time-to-market, and increased expectations of operability and reliability for established product lines. Therefore, it is willing to spend on hardware and software as long as analysts can obtain meaningful insights in a time commensurate with design cycles. Overnight runs fit this timescale, and in this context, we aim to answer the following question: *can we use LES modelling to simulate complex industrial flows with overnight simulations accurately?*

2 Rethinking CFD for present and future portability

Building codes on top of a minimal set of basic kernels is the cornerstone for portability and optimisation, which became crucial after the increasing variety of computational architectures competing in the exascale race. Moreover, the hybridisation of HPC systems imposes additional constraints since heterogeneous computations are usually needed to engage processors and massively parallel accelerators efficiently. This involves different parallel paradigms and computing frameworks and requires complex data exchanges between computing units. However, legacy CFD codes usually rely on sophisticated data structures and computing subroutines, making portability extremely complex.

In this context, we proposed a completely different approach. That is, making CFD algorithms rely on a very reduced set of algebraic kernels, *e.g.*, the sparse matrix-vector product (`SpMV`), the dot product (`dot`) and the linear combination of vectors (`axpy`), see X. Álvarez-Farré et al. (2018). This imposes restrictions and challenges that need to be addressed, such as the low arithmetic intensity of the `SpMV`, the reformulation of boundary conditions and flux limiters (see N. Valle et al. (2022)) or the efficient computation of eigenbounds to determine the time-step, Δt.

3 Challenges and opportunities

Relying on a minimal set of algebraic kernels enables code portability and facilitates its maintenance and optimisation. However, it comes together with two types of challenges and restrictions. Firstly, *computational challenges* like the low arithmetic intensity of the `SpMV`, which can be alleviated by using the more compute-intensive sparse matrix-matrix product (`SpMM`). This is possible in a great variety of situations, such as with multiple transport equations, in cases with spatial reflection symmetries, parallel-in-time simulations and, in general, whenever dealing with matrices, $\hat{\mathsf{A}} \in \mathbb{R}^{N\times N}$, decomposable as the Kronecker product of a diagonal matrix, $\mathsf{C} \equiv diag(\boldsymbol{c}) \in \mathbb{R}^{K\times K}$, and a sparse matrix, $\mathsf{A} \in \mathbb{R}^{N/K\times N/K}$, *i.e.*, $\hat{\mathsf{A}} = \mathsf{C} \otimes \mathsf{A}$. Indeed, under such circumstances, the standard `SpMV`:

$$\boldsymbol{y} = \hat{\mathsf{A}}\boldsymbol{x} \tag{1}$$

can be replaced with the `SpMM`:

$$(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_K) = \mathsf{A}\,(c_1\boldsymbol{x}_1, \ldots, c_K\boldsymbol{x}_K)\,, \tag{2}$$

where $\boldsymbol{x}_i, \boldsymbol{y}_i \in \mathbb{R}^{N/K}$. By doing so, matrix coefficients are recycled, thus significantly reducing the memory accesses and the memory footprint of the operators.

Secondly, *algorithmic challenges* such as reformulating classical flux limiters (see N. Valle et al. (2022)) or the boundary conditions must also be addressed. The latter can be naturally solved by casting boundary conditions into an affine transformation:

$$\boldsymbol{\varphi}_h \rightarrow \mathsf{A}\boldsymbol{\varphi}_h + \boldsymbol{b}_h, \tag{3}$$

allowing a purely algebraic treatment of virtually all existing boundary conditions both for explicit and implicit time-integration methods. Furthermore, an accurate and portable approach solely relying on the above-mentioned algebraic kernels for bounding the eigenvalues of the convective and diffusive operators has also been proposed.

4 Performance analysis and application to CSP technologies

The performance analysis of the code will be done in two stages. Firstly, we will focus on the performance analysis for a given mesh without considering the results' accuracy. The comparison will be made with the open-source CFD code OpenFOAM. Several factors will be analysed separately and compared, such as exploiting the shared-memory paradigm with OpenMP, increasing the arithmetic intensity through the strategies mentioned above, and the effect of GPUs. In the second stage, a relevant case from the CSP industry (see fig. 1) will be studied as a demonstrative test case to assess the feasibility of overnight industrial simulations.

All the numerical experiments were conducted on the Snellius supercomputer at SURF. The CPU-only nodes on which we ran the tests are equipped with two AMD Rome 7H12 (64 cores, 2.6 GHz, 256 MB L3 cache and 204.8 GB/s memory bandwidth) linked to 256 GB of RAM and interconnected through HDR100 ConnectX-6.

Regarding the domain considered for the experiments, it is analogous to that of fig. 2. As discussed

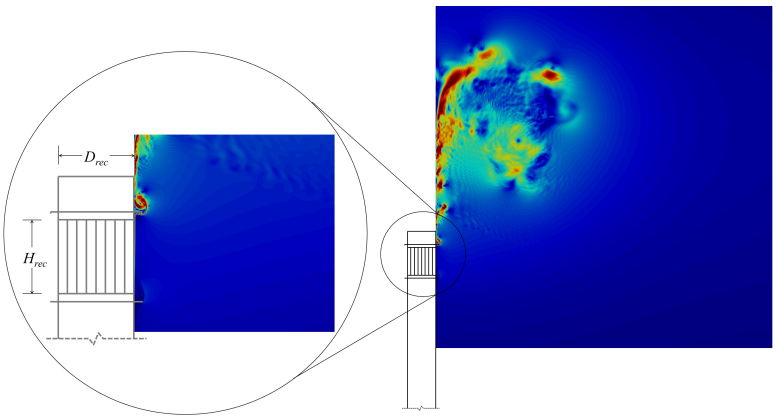


Figure 1: CSP simulation of a central tower receiver.

in Alsalti-Baldellou et al. (2023), we only discretised a fraction of it in accordance with the number of symmetries being exploited. Namely, a half, a quarter and an eighth of the domain when exploiting $s = 1, 2, 3$ symmetries, respectively. Then, given the mesh resolution demands of high-fidelity simulations of the CSP case studied, the discretisation considered had 512 million grid points (see David et al. (2021); David et al. (2023)).

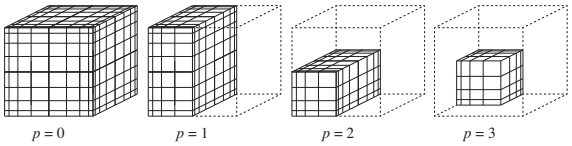


Figure 2: Example of portions meshed when exploiting s reflection symmetries.

To evaluate the parallel efficiency of TFA and compare it to that of OpenFOAM, we benchmarked OpenFOAM's PISO against TFA's fully-explicit fractional step method. Given that both algorithms cannot be directly compared, we limited these preliminary results to analysing how both codes scaled given their implementation. More concretely, given a fixed 512 million problem size, fig. 3 presents the resulting strong scalability plots. On the one hand, the fact that OpenFOAM relies on an MPI-only approach assigning an MPI process to each CPU core results in larger communication overheads and, consequently, poorer parallel efficiencies. On the other hand, TFA scales significantly better thanks to combining distributed and shared-memory parallelism. Indeed, its hybrid MPI+OpenMP parallelisation reduces the required communications, leading to higher efficiencies as the number of processors increases.

In order to assess the feasibility of overnight

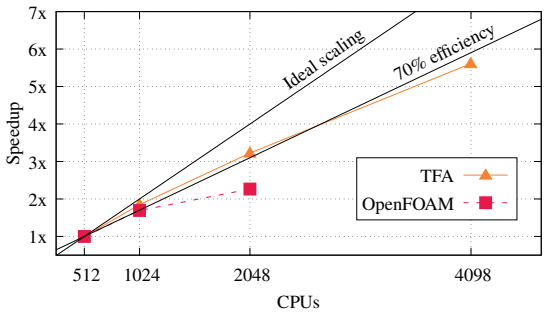


Figure 3: Strong scalability up to 70% efficiency.

LES simulations, the CSP case of fig. 1 has been studied. More concretely, we recalled the excellent weak scaling of TFA, as demonstrated in X. Álvarez-Farré et al. (2018) through the almost perfect weak scalability of `SpMV`. Then, by picking from fig. 3 the CPU work loads of 1024, 2048 and 4096 CPUs, which correspond to 95%, 75% and 65% parallel efficiencies, we could approximate the maximum simulation sizes that TFA could afford in overnight simulations, *i.e.*, taking at most 16 wall-clock hours. This was very convenient given the limited availability of computing resources, as it allowed us to obtain low-cost approximations of the positive impact of exploiting symmetries and leveraging GPUs.

On the one hand, we can infer from fig. 3 the time a time-step takes, $T^{\text{eff}}_{\Delta t}$, at a given parallel efficiency and on a mesh of size N_{ref}. On the other hand, the number of time-steps required to simulate τ time units can be approximated as follows:

$$n_{\Delta t} = \frac{\tau}{\Delta t}. \tag{4}$$

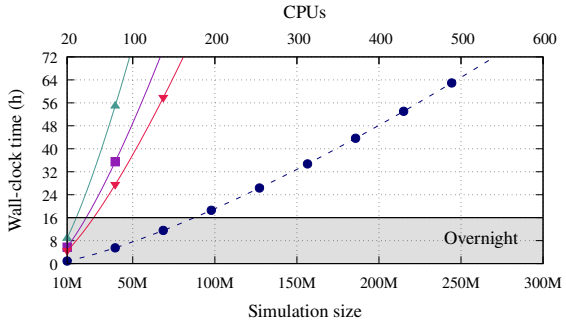
Then, recalling that LES are generally convection-dominated, its time-step can be approximated as follows:

$$\Delta t = \min\left\{\frac{\Delta x_i}{|u_i|}\right\} \simeq \frac{c}{\sqrt[3]{N}}, \tag{5}$$

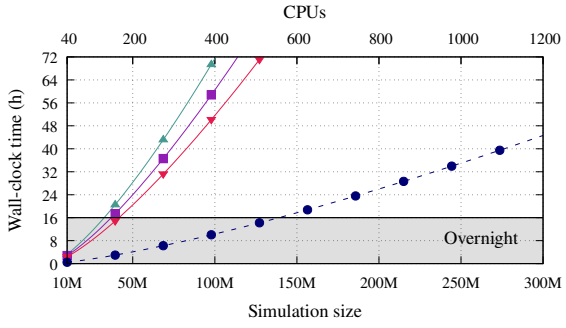
where Δx, u and N stand for the cell length, local velocity and mesh size, respectively. Then, from eq. (5), Δt is (approximately) inversely proportional to $\sqrt[3]{N}$ and, as shown in Trias et al. (2010), the correction constant c typically takes values around 0.3. Therefore, the wall-clock time of a simulation of size N can be approximated as:

$$T_{\text{LES}}(N) \simeq n_{\Delta t} \frac{T^{\text{eff}}_{\Delta t} N}{N_{\text{ref}}} = \frac{\tau T^{\text{eff}}_{\Delta t}}{c N_{\text{ref}}} \sqrt[3]{N^4}. \qquad (6)$$

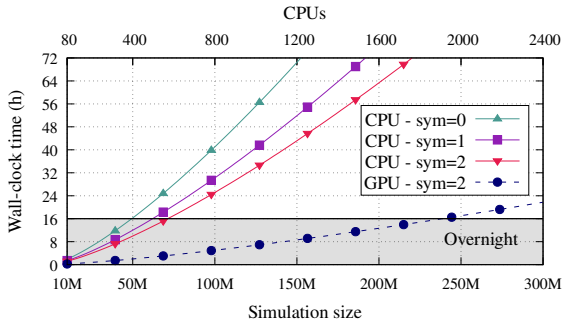
When it comes to τ, it is also shown in Trias et al. (2010) that after around 100 time-units the flow starts to become statistically stationary. Therefore, fig. 4 considers the wall-clock time for simulating $\tau = 150$ time units.



(a) 95% efficiency



(b) 75% efficiency



(c) 65% efficiency

Figure 4: Estimation of largest affordable overnight simulations in up to 16 wall-clock time hours. Results for several efficiencies according to fig. 3.

As expected, increasing the number of CPUs allows for larger overnight simulations, but this comes at the cost of lower efficiencies. According to fig. 4c, restricting ourselves to 70% efficiencies allows for around 50M overnight LES simulations. Then, thanks to exploiting up to two symmetries, 75M simulations become affordable. Nonetheless, as shown in David et al. (2021); David et al. (2023), high-fidelity simulations of CSP applications would easily require 300 to 500 million-sized grids. As a result, leveraging massively-parallel accelerators is imperative. In this sense, one of the critical advantages of TFA is its modular design, which provides support for both CUDA and OpenCL, therefore covering virtually all GPU vendors. Given our limited access to the GPU nodes, the CPU tests could not be extended to GPUs. As a result, we were forced to approximate its performance by applying a conservative speed-up factor based on past results, see X. Álvarez-Farré et al. (2018). More concretely, by assuming a 5x GPU speed-up, fig. 4c illustrates how up to 250M overnight simulations can be tackled.

5 Conclusions

This conference paper presents a symmetry-preserving discretisation for unstructured collocated grids that is virtually free of artificial dissipation and unconditionally stable. The proposed discretisation relies on a minimal set of algebraic kernels to ensure cross-platform portability, making it suitable for complex geometries and modern computational architectures.

The paper addresses several challenges related to the algebraic approach, such as the low arithmetic intensity of the sparse matrix-vector product, reformulating boundary conditions and flux limiters, and efficiently computing eigenbounds to determine the time-step.

The advantages and disadvantages of the proposed approach are analysed by comparing our in-house code, TFA, with OpenFOAM. The performance analysis demonstrates that the algebraic approach in TFA scales better thanks to its hybrid MPI+OpenMP parallelisation, leading to higher efficiencies as the number of processors increases.

Furthermore, this paper evaluates the feasibility of overnight industrial LES simulations. With this aim, we recall a relevant case from the CSP industry to estimate the largest affordable overnight simulations. As it is shown, leveraging massively-parallel accelerators such as GPUs becomes crucial for meeting the industry demands, and we estimate that TFA could tackle up to 250M overnight simulations.

Overall, this work shows promising results for enabling reliable DNS and LES simulations of turbulent flows in industrial applications with robust and stable numerical methods on modern high-performance computing systems. It demonstrates the potential of the algebraic approach to achieve performance portability while proposing strategies to increase the arithmetic intensity of the simulations.

Regarding future work, we aim to extend the scalability tests both on CPUs and GPUs. Additionally, we plan to enrich the comparison between TFA and OpenFOAM by studying their cost vs accuracy.

Acknowledgments

A.A.B., G.C., J.A.H., F.X.T., C.D.P.S. and A.O. were financially supported by the competitive R+D project RETOtwin (PDC2021-120970-I00), given by MCIN/AEI/10.13039/501100011033 and European Union Next GenerationEU. A.A.B. has also been supported by the predoctoral grants DIN2018-010061 and 2019-DI-90, given by MCIN/AEI/10.13039/501100011033 and the Catalan Agency for Management of University and Research Grants (AGAUR), respectively. J.A.H. has also been supported by the predoctoral grant 2022 FI_B1 00204, given by AGAUR. Calculations were carried out on Snellius supercomputer at SURF. The authors thankfully acknowledge these institutions.

References

Alsalti-Baldellou, A., X. Álvarez-Farré, F. X. Trias, and A. Oliva (2023). "Exploiting spatial symmetries for solving Poisson's equation". In: *Journal of Computational Physics* 486, p. 112133.

Beckingsale, David A. et al. (2019). "RAJA: Portable Performance for Large-Scale Scientific Applications". In: *2019 IEEE/ACM International Workshop on Performance, Portability and Productivity in HPC (P3HPC)*, pp. 71–81.

Bell, Nathan and Michael Garland (2009). "Implementing sparse matrix-vector multiplication on throughput-oriented processors". In: *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, pp. 1–11.

Carter Edwards, H., Christian R. Trott, and Daniel Sunderland (2014). "Kokkos: Enabling manycore performance portability through polymorphic memory access patterns". In: *Journal of Parallel and Distributed Computing* 74(12). Domain-Specific Languages and High-Level Frameworks for High-Performance Computing, pp. 3202–3216.

David, M., A. Toutant, and F. Bataille (Apr. 2021). "Investigation of thermal large-eddy simulation approaches in a highly turbulent channel flow submitted to strong asymmetric heating". In: *Physics of Fluids* 33(4), p. 045104.

David, M., A. Toutant, and F. Bataille (Mar. 2023). "Thermal large-eddy simulation methods to model highly anisothermal and turbulent flows". In: *Physics of Fluids* 35(3), p. 035106.

E. Komen, J. A. Hopman, E. M. A. Frederix, F. X. Trias, and R. W. C. P. Verstappen (2021). "A symmetry-preserving second-order time-accurate PISO-based method". In: *Computers & Fluids* 225, p. 104979.

F. X. Trias, O. Lehmkuhl, A. Oliva, C.D. Pérez-Segarra, and R. W. C. P. Verstappen (2014). "Symmetry-preserving discretization of Navier-Stokes equations on collocated unstructured meshes". In: *Journal of Computational Physics* 258 (1), pp. 246–267.

Greathouse, Joseph L., Kent Knox, Jakub Poła, Kiran Varaganti, and Mayank Daga (Apr. 2016). "clSPARSE: A Vendor-Optimized Open-Source Sparse BLAS Library". In: *IWOCL '16: Proceedings of the 4th International Workshop on OpenCL*. ACM: New York, NY, USA.

Löhner, R., C. Othmer, M. Mrosek, A. Figueroa, and A. Degro (2021). "Overnight industrial LES for external aerodynamics". In: *Computers & Fluids* 214, p. 104771.

N. Valle, X. Álvarez-Farré, A. Gorobets, J. Castro, A. Oliva, and F. X. Trias (2022). "On the implementation of flux limiters in algebraic frameworks". In: *Computer Physics Communications* 271, p. 108230.

Trias, F.X., A. Gorobets, M. Soria, and A. Oliva (2010). "Direct numerical simulation of a differentially heated cavity of aspect ratio 4 with Rayleigh numbers up to 1011 – Part I: Numerical methods and time-averaged flow". In: *International Journal of Heat and Mass Transfer* 53(4), pp. 665–673.

Witherden, F. D., A. M. Farrington, and P. E. Vincent (2014). "PyFR: An open source framework for solving advection–diffusion type problems on streaming architectures using the flux reconstruction approach". In: *Computer Physics Communications* 185(11), pp. 3028–3040.

X. Álvarez-Farré, A. Gorobets, F. X. Trias, R. Borrell, and G. Oyarzun (2018). "HPC^2 – A fully portable algebra-dominant framework for heterogeneous computing. Application to CFD". In: *Computers & Fluids* 173, pp. 285–292.