

# In situ analysis and visualization of massively parallel simulations of transitional and turbulent flows

**A. Cadiou and M. Buffat and C. Pera and B. Di Pierro and  
F. Alizard and L. Le Penven**

Laboratoire de Mécanique des Fluides et d'Acoustique,  
Université Claude-Bernard Lyon 1/ CNRS/ École centrale de Lyon/ INSA de Lyon,  
43, bd du 11 novembre 1918, 69622 Villeurbanne, France.

E-mail: [anne.cadiou@ec-lyon.fr](mailto:anne.cadiou@ec-lyon.fr)

**Abstract.** The increase of computational resources with the generalization of massively parallel supercomputers benefits to various fields of physics among which turbulence and fluid mechanics, making it possible to increase time and space accuracy and gain further knowledge in fundamental mechanisms. Parametric studies, high fidelity statistics, high resolutions, can be realized. However, this access poses many problems in terms of data management, analysis and visualization. Traditional workflow, consisting of writing raw data on disks and performing post-processing to extract physical quantities of interest, considerably slows down the analysis, if not becomes impossible, because of data transfer, storage and re-accessibility issues. This is particularly difficult when it comes to visualization. Usage has to be revisited to maintain consistency with the accuracy of the computation step and in this context, in situ processing is a promising approach. We developed an in situ analysis and visualization strategy with a hybrid method for transitional and turbulent flow analysis with a pseudo-spectral solver. It is shown to have a low impact on computational time with a reasonable increase of resource usage, while enriching data exploration. Large time sequences have been analyzed. This could not have been achieved with the traditional workflow. Moreover, computational steering has been performed with real-time adjustment of the simulations, thereby getting closer to a numerical experiment process.

## 1. Introduction

Massively parallel computations help gaining more insights into physics by increasing accuracy in space and time resolutions. However, knowledge extraction is getting more and more difficult because analysis traditionally based on post-processing of data written on disks is facing several limitations. Data I/O and transfer has gained less efficiency than computation, so that the data analysis process is now the bottleneck of high performance computing. Traditional workflow has to be revisited in order to circumvent it. In situ analysis and visualization has gained interest in this context, even if many challenges remain [7].

It consists in coupling the analysis with the solver in the simulations, so that both are performed simultaneously, thereby allowing to preserve the space and time accuracy retained in the simulation into the analysis. This can be done in a tightly (co-processing) or loosely coupled (concurrent-processing) way with distinctive advantages and disadvantages [9], but both share the fact that the analysis and visualization are performed before data is written on disks.

Many solutions are explored in the literature, some providing libraries for in-situ visualization (often tightly coupled), others designing an entire framework for new workflows (often loosely coupled). We tackle related issues in both ways, by simply instrumenting our code for an original hybrid in situ analysis in the context of massively parallel simulations. This was applied to our spectral code for the need of physical exploration of transitional and turbulent mechanisms in a highly resolved plane channel flow. The HPC background is presented in 2 and the in situ technique is described in section 3. An application for in situ visualization and computational steering is given in section 4, corresponding to a mesochallenge where half of the local cluster has been used for the demonstration experience [3].

## 2. HPC for the study of transitional and turbulent flows

The study of turbulence and the mechanisms of transition to this state make use of intensive computations since the 70s, with direct numerical simulations of the governing Navier-Stokes equations ([10] [11] [12]). Simulations of fundamental aspects of these phenomena require accuracy with always increasing space and time resolutions, in the exploration of universal scaling laws, self-similarities of well-known features, in the understanding of rare events and anomalous spectra, in the accumulation of various statistics, etc. If universality exists, it is only expected at high Reynolds numbers. Unfortunately, the degrees of freedom grow very rapidly with the Reynolds number and are limited by the available computer resources [15]. Pseudo-spectral methods are retained since long to solve the Navier-Stokes equations because of their exact representation of the spatial derivatives, and this is the numerical method retained in our solver. Nonlinear terms are calculated in the physical space to avoid the calculation of convolution products, so that Fourier transforms have to be performed back and forth to the spectral space. To know the velocity in one location, one must calculate the spectral coefficients in the entire domain, making those methods difficult to implement efficiently on massively parallel supercomputers, because of their global properties. Decomposition of the domain is compulsory in computations involving hundreds of millions to billions of degrees of freedom. This means that lots of communications are needed between processes to access to the global set of variables needed in the spectral transforms. Despite this, modern multi-core architectures can be used with high efficiency, taking advantage of hybrid MPI/OpenMP parallelization and tasks parallelization between the three spatial directions [8]. Our code [2] solves the incompressible Navier-Stokes equations and a reasonable speedup has been obtained on a BlueGene/Q architecture for one billion of degrees of freedom (see figure 1), with 88% of the time spent in computation. One time step takes 0.2s (walltime) on 16384 cores, which is fast for a global method.

With one billion of degrees of freedom, one velocity field consists in 27 Go of disk usage. Managing a large set of data during the simulation and after has always been the case in the field of turbulence, but the I/O bandwidth efficiency did not keep pace with the increasing FLOPS on supercomputers and the data transfer between servers is increasingly time consuming with the growth of data volumes. This considerably slows down the analysis and visualization of the data generated by the simulations. This trend will remain for at least the next decade, enhancing the interest for in situ processing.

## 3. Embedded analysis

Simulations nowadays often consist in multiple runs in batch mode on remote multi-cores massively parallel supercomputers. This traditional workflow results in the production of data files, that are written on disks and transferred on a local data server for post-processing thanks to remote access (via NFS or ssh tunneling) to a local computer server.

In the perspective of exascale computing, this workflow is no longer sustainable because of lack of I/O performance and increasing data generation leading to storage, transfer and re-access

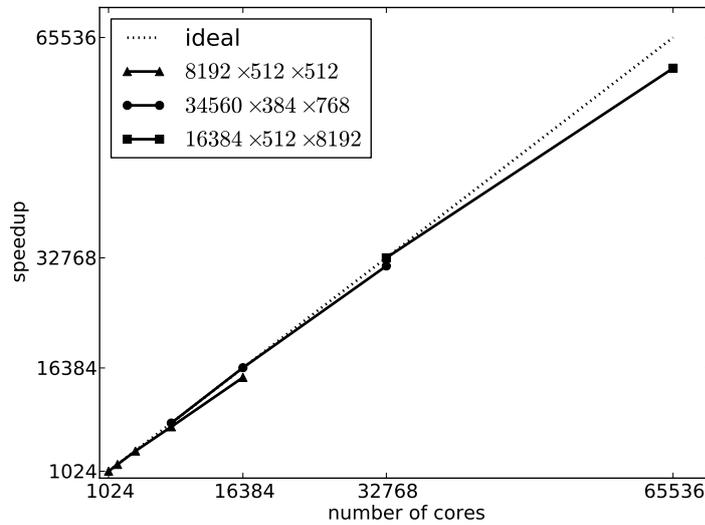


Figure 1: Speedup on JUQUEEN (PRACE)

issues.

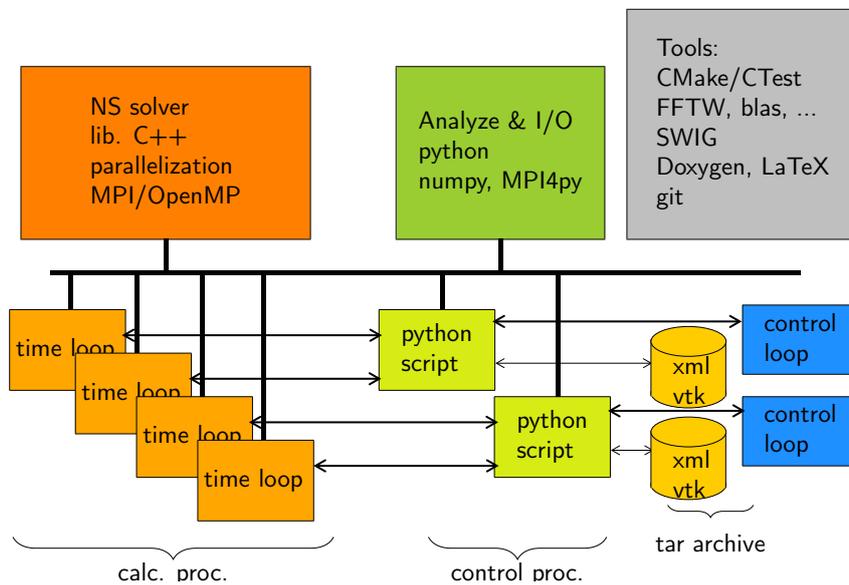


Figure 2: Scheme of the hybrid embedded analysis code.

To minimize the volume of raw data, a part of the analysis is performed within the solver code, during the time steps of the simulation (accumulation of statistics, profiles or spectra extractions, etc.), in a tightly coupled manner. This has two major drawbacks: the analysis shares the resources with the solver, thereby reducing its overall efficiency, and must be defined in the code before running the simulation because of the compilation. Consequently, exploration of the influence of a physical parameters or management of unplanned observation is rather slow

as the simulation must be stopped to modify the analysis accordingly. The hybrid in situ strategy presented here circumvents these drawbacks by embedding the analysis in the code in an asynchronous way (figure 2). The analysis is performed on a distinct set (if needed) of computational cores from the ones used by the solver. There is no need to write the data on disks for asynchronous processing analysis, even if I/O could be entrusted to the analysis. There is two groups of MPI communicators, one for the solver part, the other one for the analysis part. Computational steering can be performed thanks to the fact that the analysis is using python, an interpreted language, while the solver is written in C++. Accuracy of the spatial operators is however preserved because python is actually coupled to the C++ libraries of the code thereby using the same operators than the code. The originality of our approach lies in the fact that the interpreters are launched by the C++ code. Modifications of the analysis can be performed by changing the instructions in the python script during the computation. This also allows to keep the same way of working with post-processing of data generated on disks by the solver (if needed), only no time step is performed in this case. The libsim [14] interface is used to connect VisIt[13] to the analysis communicator for 3D visualization.

#### 4. Example of application : in situ visualization and computational steering

We focus here on the spatio-temporal development of instabilities growing at the entrance of a plane channel under the effect of a small perturbation at the channel entry. The form of the unstable structures will be observed with the  $\Lambda_2$  criterion [6], which is the second eigenvalue of the pressure Hessian. In our solver, the pressure has been eliminated by construction, so that it has to be first calculated from the velocity field. A demonstrator has been done for a channel configuration of size  $L_x/h \times L_y/h \times L_z/h = 75 \times 2 \times 6.4$  with  $5760 \times 192 \times 512$  modes ( $\sim 566$  millions of modes) where  $h$  is the channel half-width, each velocity field occupies  $\sim 14$  Go. If the solver runs on 2048 cores with 4 OpenMP threads per MPI tasks, and I/O would be performed by the solver, each of the 512 processes would have to write its own partition on disk, i.e. each file containing the velocity field would be decomposed into 512 parts. If written on disks, they are directly stored in separate directory per time step, to avoid having inodes limitations. The post-processing of the  $\Lambda_2$  criterion on one velocity field consistent with the spectral accuracy of the solver requires FFTs (back and forth) and is possible (in terms of memory) with only 16 MPI tasks. If the asynchronous embedded analysis only performs I/O of the velocity field on disks, each file containing the velocity field would be partitioned in only 16 parts. Writing one velocity field takes less than 30% of the time spent per solver time step, therefore the global restitution time of the simulation would not be strongly affected by the asynchronous I/Os in this small case because the time stride for tracking the  $\Lambda_2$  criterion is about 50 time steps. The time step of the solver is indeed about 100 times smaller than the physical time step needed for knowledge extraction (linked to the turnover time of the eddies). However, if every velocity field would be written on disks waiting for post-processing, a huge amount of data has to be generated (1.4 To for 100 time records).

A consistent visualization of the isovalues surfaces of this criterion also requires interpolation of the field on a regular grid, as the visualization software (here VisIt [13]) make linear interpolations on the grid points, resulting on a loss of precision at the center of the channel if the Chebychev collocations points are retained. The whole analysis takes about 10 time steps to be calculated on 16 MPI processes, and the output data consists of images (about 1 Mo each), written at each analysis step. With the in situ analysis, only slightly more than 14 Go has to be written on disks (100 Mo of images for 100 time records and 14 Go for checkpointing). More than 3000 state of the  $\Lambda_2$  criterion [4] [1] [5] were recorded thanks to this embedded in situ analysis. They have been obtained concomitantly with the direct numerical simulation. To perform it with the traditional usage, it would have been needed to write, transfer and process more than 42 To.

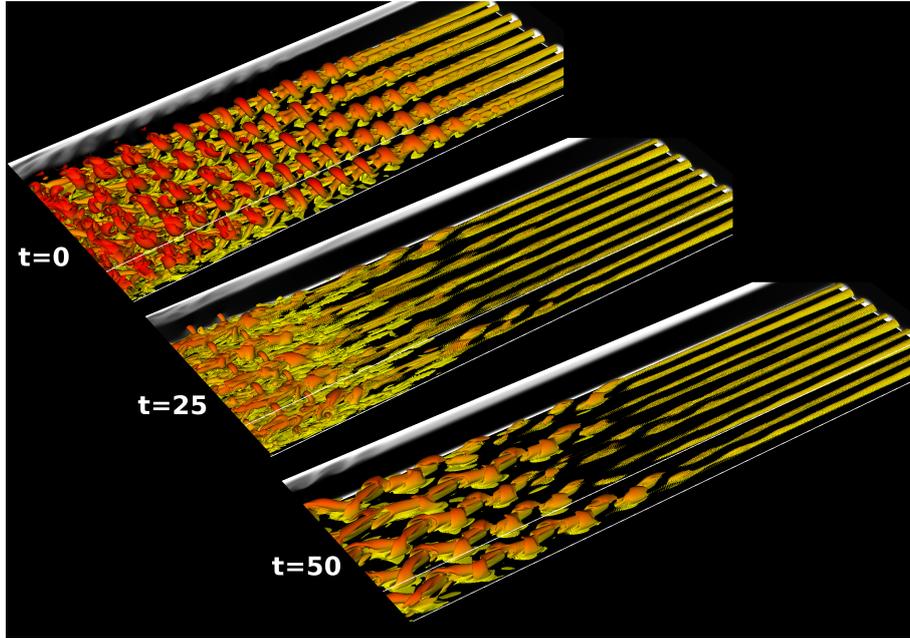


Figure 3: Real-time visualization of the spatio-temporal evolution (from the right to the left) of three-dimensional flow structures ( $\Lambda_2$  criterion) under the control of the amplitude of the perturbation at the entrance of the channel.



Figure 4: Traditional usage, 2048 cores for 512 MPI partitions, 1.4 To of data

Walltime of tightly (figure 5) and hybrid (figure 6) has been compared to the traditional usage, as pictured in the following schemes (figure 4). It has been drawn proportionally for 100 analysis at the same physical time interval  $\Delta t$ .



Figure 5: Tightly coupled in situ solution, when the cores for the solver are shared with the analysis.



Figure 6: Revisited in situ co-processing analysis, solver on 2048 cores for 512 MPI partition, 16 cores for co-processing, 14 Go output data.

In this demonstrator, the amplitude of the perturbation at the entrance of the channel has been modified while the flow was simulated. The resulting impact on the  $\Lambda_2$  criterion and the change from various-like structures from sinuous-like ones has been observed in real (CPU-) time. Our system indeed allow to steer the simulation during runtime and get immediate feedback of the modified parameter influence.

## 5. Conclusion

An asynchronous hybrid (mixed loosely and tightly coupled) in situ method has been implemented on HPC context for a massively hybrid MPI/OpenMP pseudo-spectral code to

analyze transitional and turbulent flows and perform I/O. The implementation consists in the use of python libraries such as numpy, matplotlib, MPI4py to process the data in memory and SWIG to communicate with the C++ solver. In situ visualization is based on the libsim library from VisIt. Interaction with the simulation can be performed thanks to the interpreted character of the python language. Being performed on separate groups of processes with two communicators, the analysis and visualization does not affect the solver efficiency. The asynchronous system has been chosen assuming that communications between both groups of process is only needed on a specific stride of time step, defined by the physics, whereas the time step is generally smaller, imposed by the numerical stability of the numerical scheme. This embedded analysis has been successfully applied to follow the spatio-temporal development of physical instabilities at the entrance of a plane channel, that triggers turbulence. The modification of the type of instabilities depending on the amplitude of the perturbation at the channel entry has been done during the computation and could be analyzed in real CPU time. This kind of embedded analysis is therefore a promising way to bring the simulation closer to the experimental approach.

The authors thank the PRACE project, the national computer center IDRIS from GENCI and the computer center P2CHPD at “Université Claude Bernard Lyon 1”, member of the “Fédération Lyonnaise de Modélisation et Sciences Numériques” (FLMSN), for providing computer facilities.

- [1] M. Buffat, A. Cadiou, L. Le Penven, and Ch. Pera. In-situ analysis and visualization of massively parallel computations. *Int. J. of High Perf. Computing Appl.*, pages 1–15, 2015.
- [2] M. Buffat, L. Le Penven, and A. Cadiou. An efficient spectral method based on an orthogonal decomposition of the velocity for transition analysis in wall bounded flow. *Comput. Fluids*, 42:62–72, 2011.
- [3] A. Cadiou, M. Buffat, and L. Le Penven. In-situ analysis for spectral dns of transitional and turbulent flows. In *Journée d’inauguration des plateformes de la FLMSN*, octobre 2013.
- [4] A. Cadiou, M. Buffat, and L. Le Penven. Visualisation in-situ pour l’étude de la transition sous-critique. In *HPC Magazine*, volume 7, pages 68–69. October 2013.
- [5] A. Cadiou, M. Buffat, B. Di Pierro, L. Le Penven, and C. Pera. A new strategy for analysis and visualization of massively parallel computations of turbulent and transitional flows. In *Turbulence and Interactions*, pages 101–107, Cham, 2018. Springer International Publishing.
- [6] J. Jeong and F. Hussain. On the identification of a vortex. *J. Fluid Mech.*, 285:69–74, 1995.
- [7] J. Kress. In situ visualization techniques for high performance computing. Technical report, University of Oregon, 2017.
- [8] J. Montagnier, A. Cadiou, M. Buffat, and L. Le Penven. Towards petascale simulation for transition analysis in wall bounded flow. *Int. Journ. for Num. Meth. in Fluids*, 72(7):709–723, 2013.
- [9] K. Moreland. The tensions of in situ visualization. *IEEE Computer Graphics and Applications*, 36:5–9, 2016.
- [10] S.A. Orszag. Numerical methods for the simulation of turbulence. *Physics of Fluids*, 12(12), 1969.
- [11] S.A. Orszag and G.S. Patterson. Numerical simulation of three-dimensional homogeneous isotropic turbulence. *Phys. Rev. Lett.*, 28:76–79, 1972.
- [12] P.R. Spalart and A. Leonard. Direct numerical simulation of equilibrium turbulent boundary layer. In *5th Symposium on Turbulent Shear Flows*, 1985.
- [13] VisIt. web site <https://wci.llnl.gov/codes/visit/home.html>.
- [14] B. Whitlock, J.M. Favre, and J.S. Meredith. Parallel in situ coupling of simulation with a fully featured visualization system. In *In Eurographics Symposium on Parallel Graphics and Visualization (EGPGV). Eurographics Association*, 2011.
- [15] M. Yokokawa, K. Itakura, A. Uno, T. Ishihara, and Y. Kaneda. 16.4-Tflops direct numerical simulation of turbulence by a Fourier spectral method on the earth simulator. In *Supercomputing, ACM/IEEE 2002 Conference*, 2002.