

Numerical Properties and Multi-GPU implementation of a Spatially High Order Finite Volume solver for compressible flows

Jean-Marie Le Gouez, Onera CFD Department

Special thanks to Alain Lerat, emeritus professor at ENSAM, consultant for the CFD Department



return on innovation

Multi-GPU implementation of a CFD solver

with contributions from :

Matthieu Lefebvre, former PHD student, for GPU data models and Cuda programming,

Jean-Matthieu Etancelin, HPC support engineer at the ROMEO Computing Center, Université de Champagne Ardennes, for the GPU optimization of inner kernels and cluster exchanges

Thanks to Nikolay Markovskiy dev-tech at NVIDIA research Center, GB

and to Carlos Carrascal, Research Master intern

Numerical properties and multi-GPU implementation of a CFD solver

- Context
- Projects at the CFD department on software modularity, interoperability, code reuse, capacity for evolution, performance
- NXO numerical scheme : status, deployment,
- NextFlow GPU prototype : Development stages, data models, programming languages, co-processing tools, validation and performance
- On-going work
- Outlook

General context : CFD at Onera

The recognized legacy platforms for Research and Industrial applications :

- **elsA** : aerodynamics and aeroelasticity, in direct and adjoint modes, automatic shape optimization, numerous turbulence models, Zonal Detached Eddy Simulation (ZDES)

- **Cedre** : combustion and aerothermics, 2-phase flows, heat radiation, structural thermics and thermomechanics (**Zebulon** Onera code)

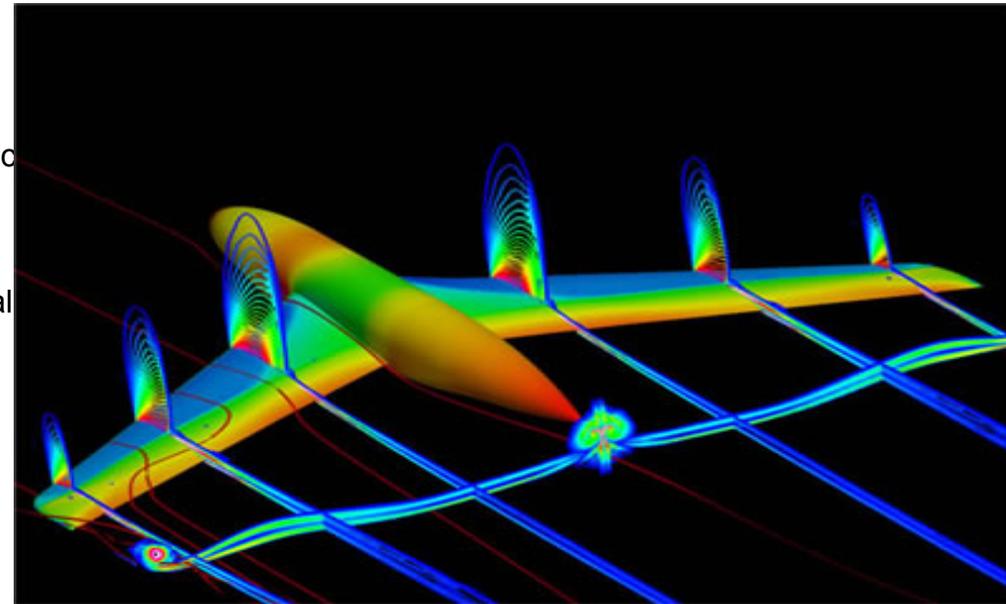
- **Sabrina/Funk** : unsteady aerodynamics on multi-block structured grids, optimized for LES and aeroacoustics

- Their associated tools for application productivity; **Cassiopée** project : links with CAD, overlapping grids management, AMR patches, grid deformation tools, scripting and modularity,...

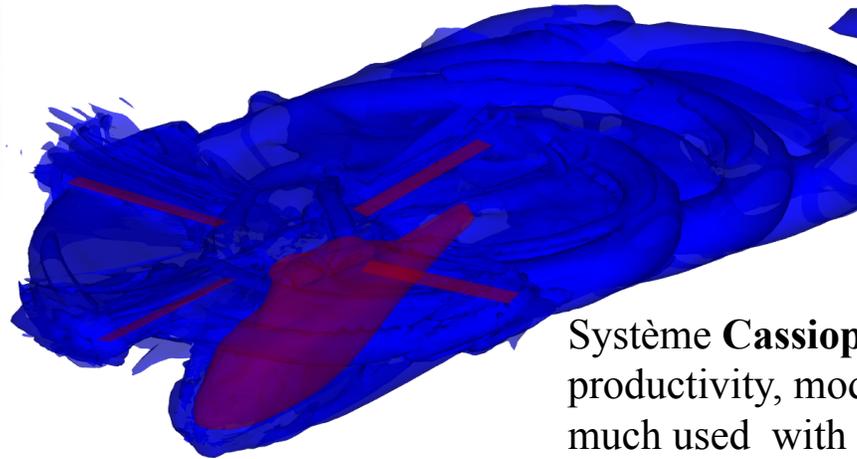
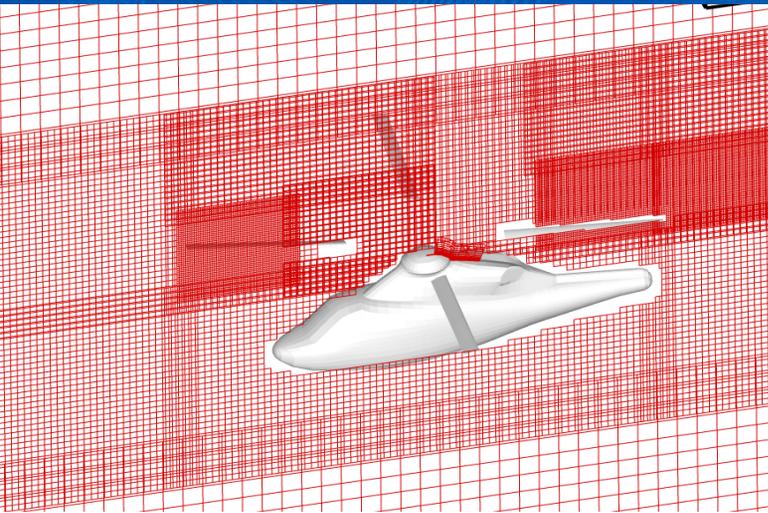
- The multi-solver coupling libraries (collaborations with Cerfacs : **Open-Palm**, **CWIPI**)

- *Hundreds of thousands of code lines, mix of Fortran, C++, python*

- *Important usage by the aerospace industries*



General context : CFD at Onera



Systeme **Cassiopee** for application productivity, modularity and coupling, much used with the elsA solver, complementary solvers, partly OpenSource

The expectations of the external users:

- **Extended simulation domains:** → effects of wake on downstream components : blade-vortex, fuselage interaction on helicopters, thermal loadings by reactor jets on composite structures,
- **Model of full systems and not only the individual components :** multi-stage turbomachinery internal flows, couplings between the combustion chamber and the turbine aerodynamics, ...
- **More multi-scale effects :** representation of technological effects to improve the overall flow system efficiency : grooves in the walls, local injectors for flow / acoustics control,
- **Advanced usage of CFD :** adjoint modes for automatic shape optimization and grid adaptation, uncertainty management, input parameters defined as pdf,

CFD at Onera

Expectations from the internal users :

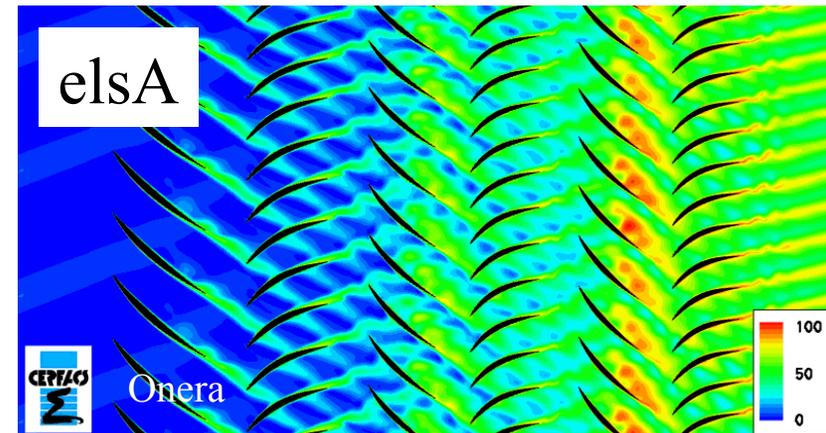
- to develop and validate state of the art physical models : transition to turbulence, wall models, sub-grid closure models, flame stability,
- to propose novel designs in rupture for aeronautics in terms of aerodynamics, propulsion integration, noise mitigation, ...
- to tackle the CFD grand challenges

➔ New classes of numerical methods, less dependent on the grids, more robust and versatile,

➔ Computational efficiency near the hardware design performance, high parallel scalability,

Decision to launch research projects :

- ➔ On deployment of the DG method for complex cases : **AGHORA** code
- ➔ On modular multi-solver architecture within the **Cassiopee** set of tools



Projects on novel application architectures

HPC prototype « FAST » : I.Mary, J.M. Le Gouez, S. Péron, C. Benoit, D. Blaise, T. Renaud

Developed in an Onera federative project with aeroelasticity and combustion departments

Experiment with:

HPC specific implementations of solvers and post-processing / HPC technologies

Modular architecture (components)

Innovative mesh strategies : Octree, hierarchical / embedded grids, Immersed BC

Must provide indications for future CFD code

Choice of modularity is made to :

Limit inter-dependency

Ease evolution / replacement / choice of modules, code reuse

Ease component inter-operability

Modules can Compile / install / run independently

They read and write a Standard data

Projects on novel application architectures / challenges

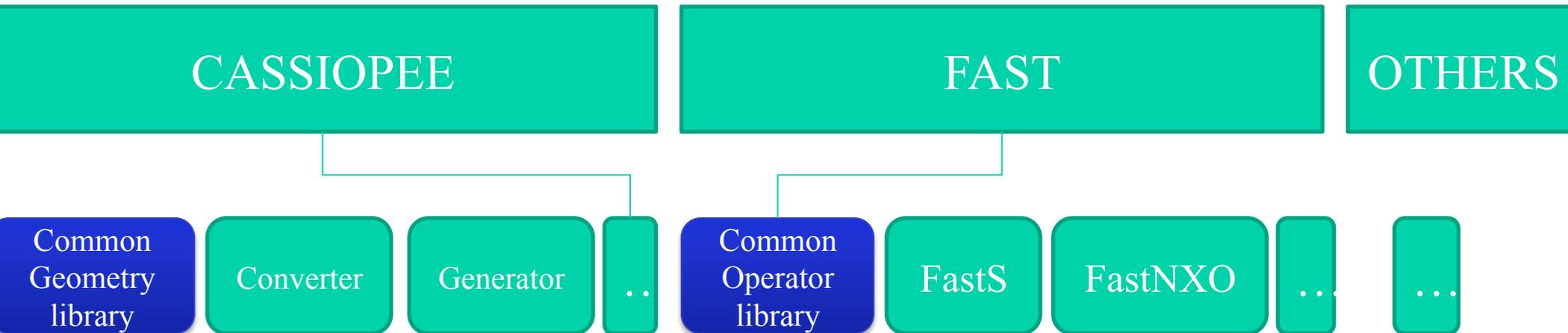
The open standard choice

- Makes possible the work of teams with minimum coordination
- Ease reuse and sustainability

The functional choice

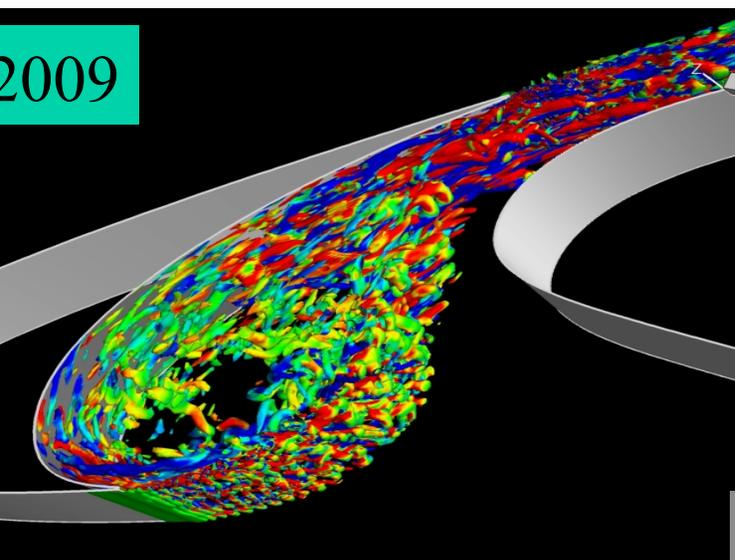
- Diminishes border effect
- Enables Workflow scripting and tasking

CGNS/Python environment

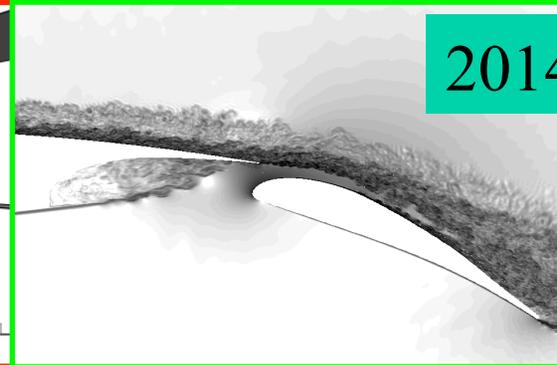
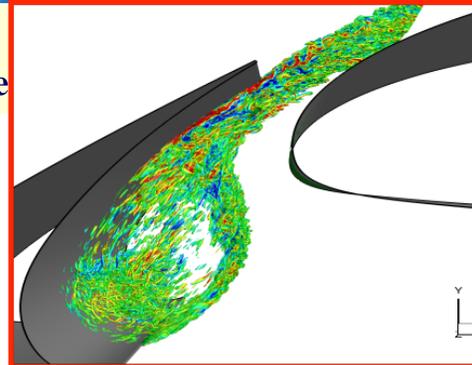


The parallel performance barrier in distributed computing :

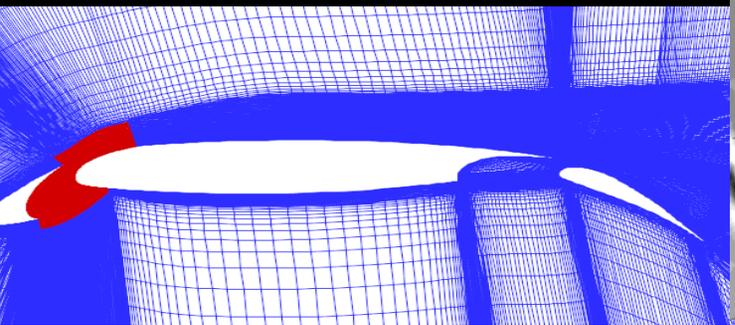
Improvement of predictive capabilities in the last 5 year : RANS / zonal LES of the flow around a High-Lift wing



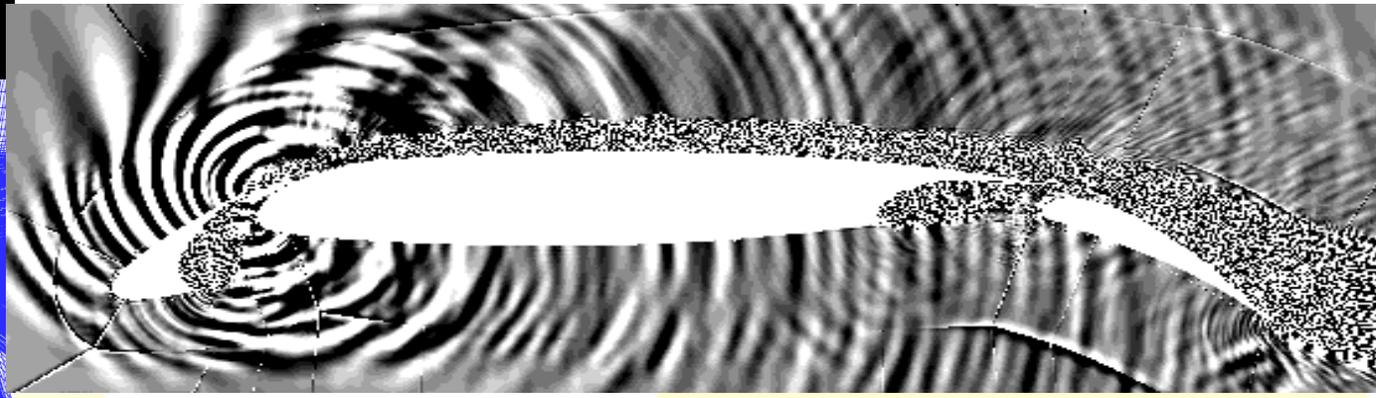
Mach 0.18
Rey 1 400 000/corde



- Optimized on a CPU architecture MPI / OpenMP / vectorization
- CPU ressources for 70ms of simulation : JADE computer (CINES)
- $N_{xyz} \sim 2\ 600\ Mpts$ 4096 cores / 10688 domains $T_{CPU} \sim 6\ 200\ 000\ h$ *Residence time : 63 days*

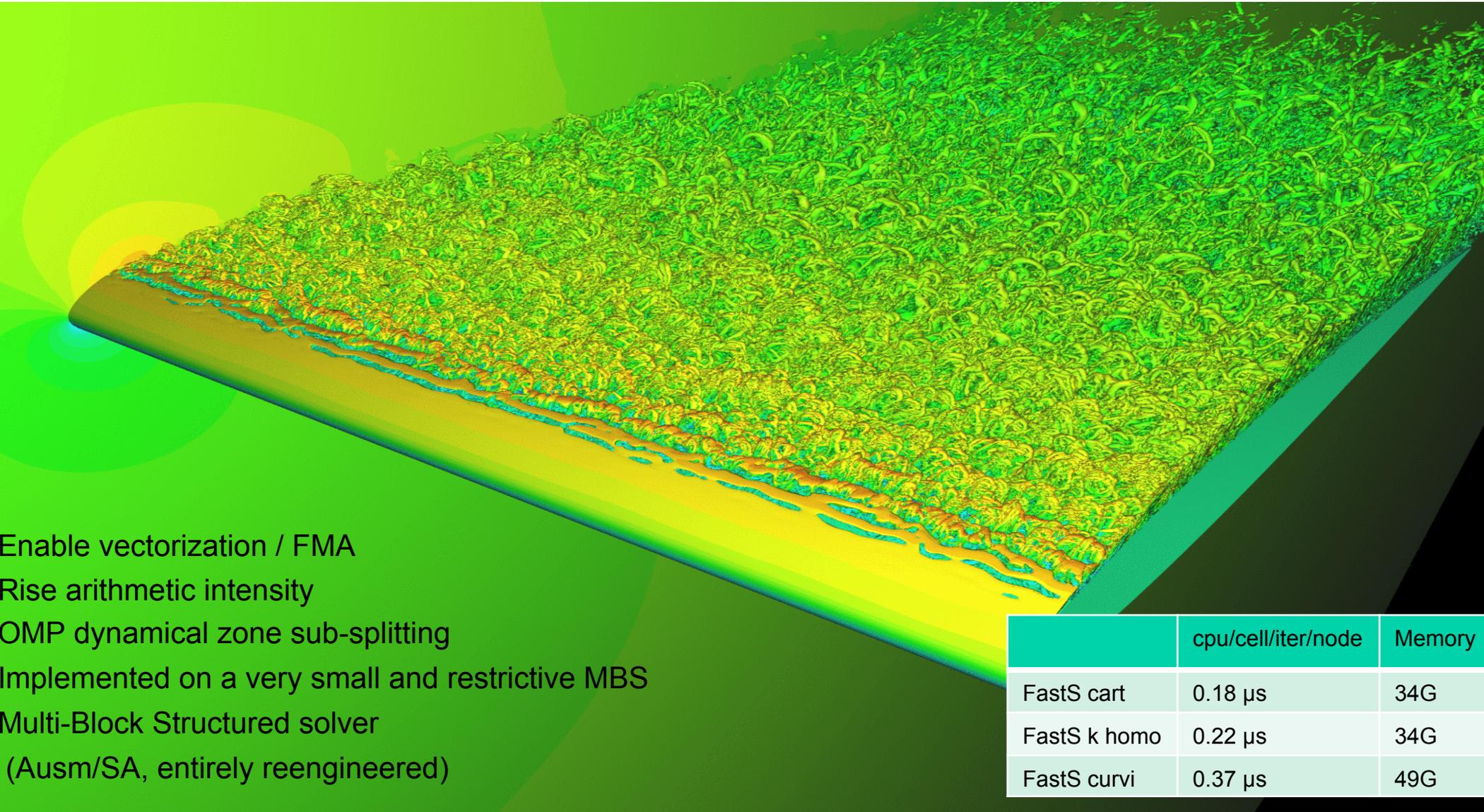


2D steady RANS and
2D LES 7,5 Mpts



LEISA project DLR / Onera cooperation
FUNK software

The parallel performance barrier in distributed computing of 2nd order FV : MPI / OpenMP



Enable vectorization / FMA
Rise arithmetic intensity
OMP dynamical zone sub-splitting
Implemented on a very small and restrictive MBS
Multi-Block Structured solver
(Ausm/SA, entirely reengineered)

	cpu/cell/iter/node	Memory
FastS cart	0.18 μ s	34G
FastS k homo	0.22 μ s	34G
FastS curvi	0.37 μ s	49G

The application productivity barrier : accuracy versus gridding time, grid management issues, wall treatment options

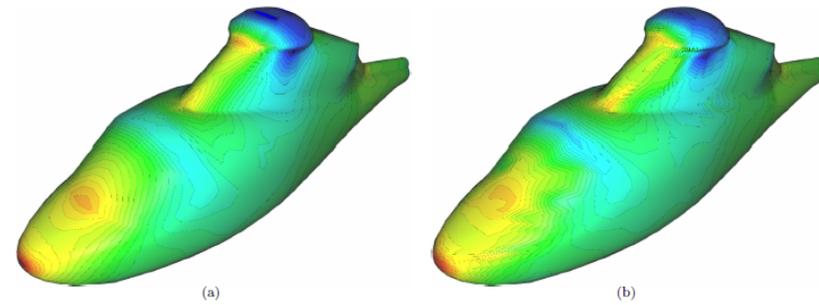
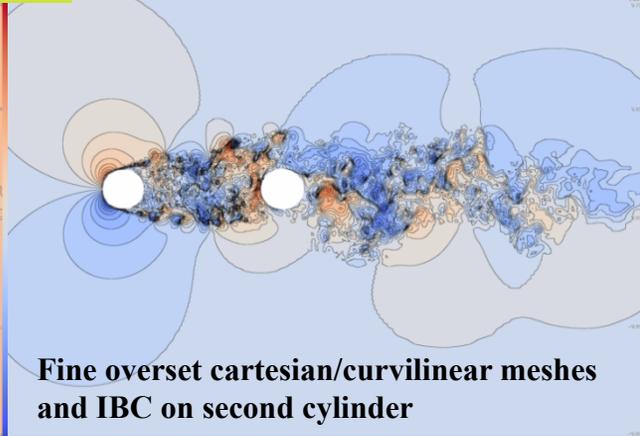
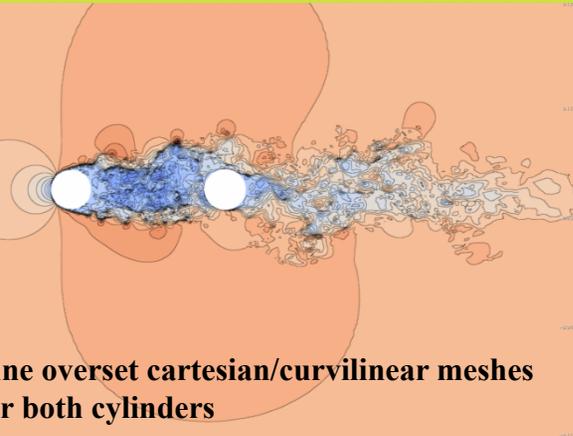
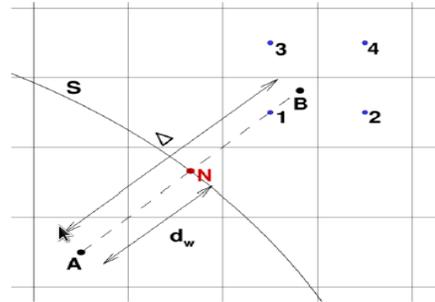
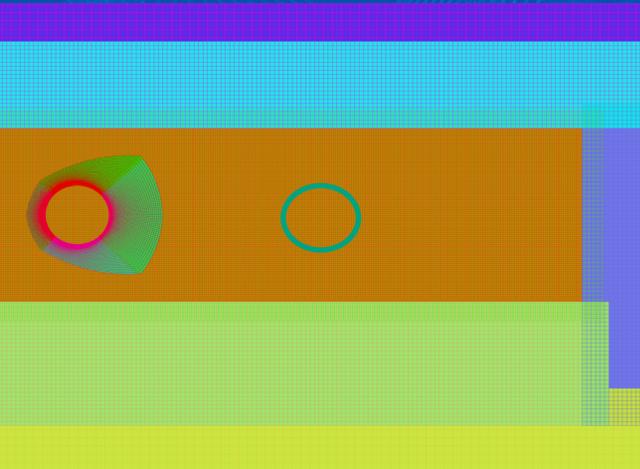
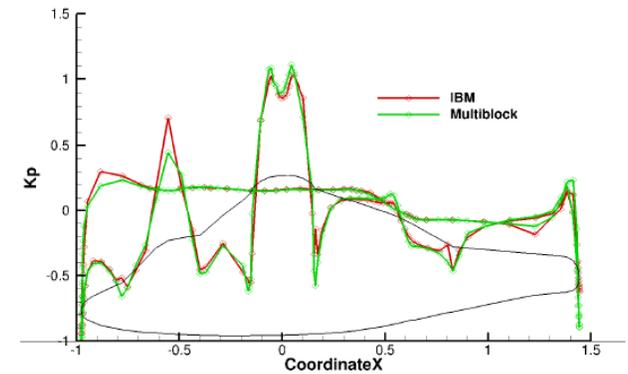


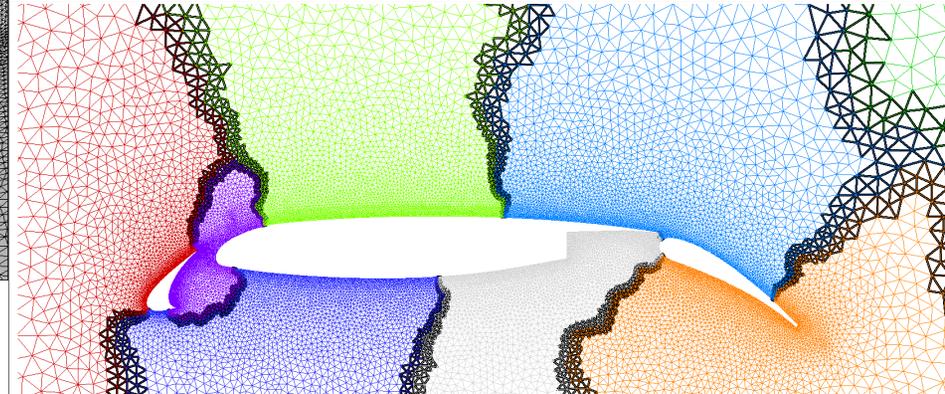
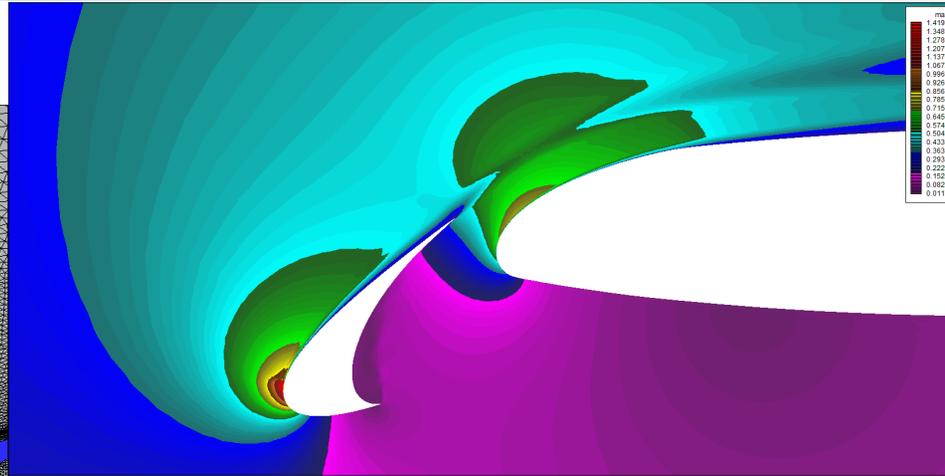
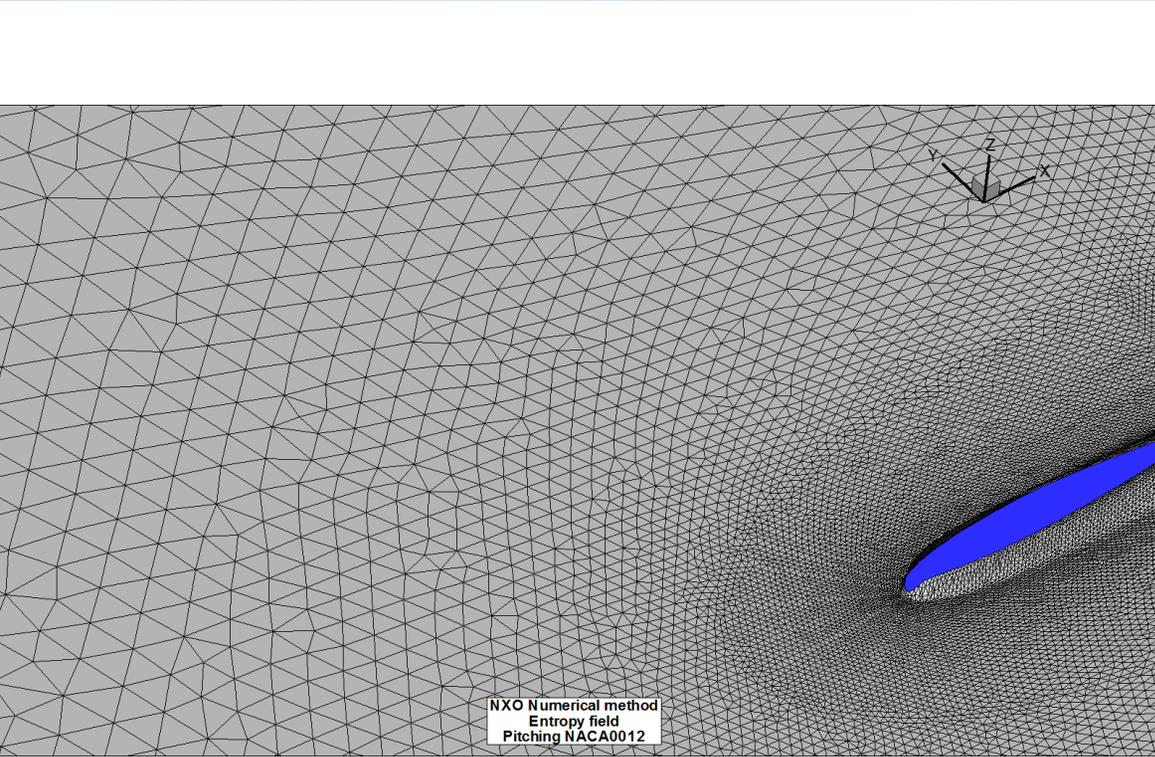
Figure 22. Density contours on the fuselage: (a) conformal mesh; (b) IBM approach.



Comparison conformal mesh /IBC on Robin fuselage
 50 times more isotropic cells for the IBC model, $y^+=100$
 Wall model, No gridding time, faster solver

Cassiopée set of tools, modules for overset grids, IBC, cartesian solver

The algorithmic efficiency barrier : accuracy versus number of dofs (DG, HO FV, FD), wide stencils methods, coarse partitionning / node, shared memory programming



NextFlow : Spatially High-Order Finite Volume method for RANS / LES
Demonstration of the feasibility of porting these algorithms on heterogeneous architectures : TESLA GPU

NASA AMS Seminar April 5th 2016

Basics of the NXO scheme

Euler or Navier-Stokes for perfect gas law of state
1 dof per cell and per equation (Volume average)

Polynomial Reconstruction algorithm for the conservative variables *or flux density fields*

Preprocessor phase : Weighted Least-Square polynomial degree adapts to the “quality” of the stencils

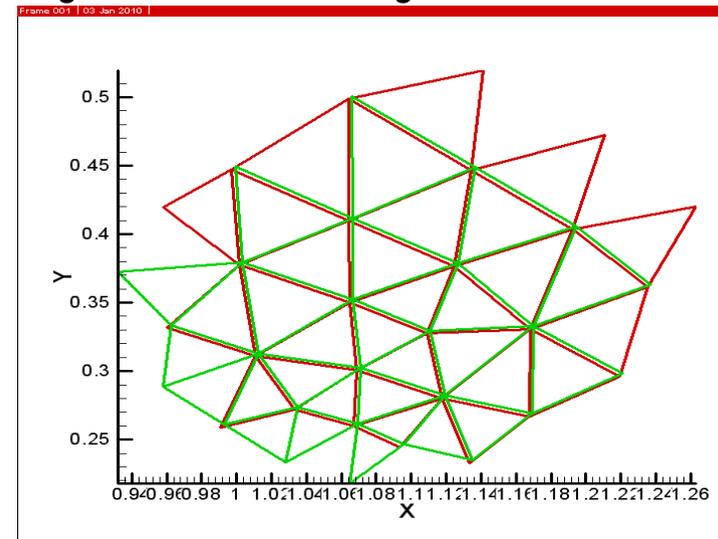
→ Gives the interpolation coefficients of conservative variable fields from volume averages to surface averages

For the Euler fluxes :

*cell-centered stencil (in red) for the reconstruction
projection on one of the faces of this cell*

For the diffusive fluxes :

*Interface-centered stencil (union red-green)
Projection of the gradients of the polynomial*



NextFlow : Spatially High-Order Finite Volume method for RANS / LES

Demonstration of the feasibility of porting these algorithms on heterogeneous architectures : TESLA GPU

NASA AMS Seminar April 5th 2016

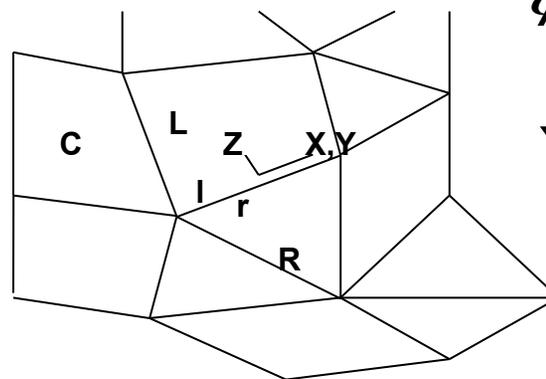
H.O. Full 3D Volume to face interpolation : Reconstruction and projection

$$\phi_a(X, Y, Z) = a_{\{ijk\}} X^i Y^j Z^k$$

$$\psi = \sum_{s=1}^{ns} \varpi_s \left(\Omega_{c(s)} \bar{\phi}_{c(s)} - \int_{\Omega_{c(s)}} \phi_a(X, Y, Z) dV \right)^2$$

Reconstruction error functional

ns = Stencil size : nb of monomials + 50%



$$\hat{\phi}_l = \sum_{c=1}^{ns} \lambda_c \bar{\phi}_c$$

$$\hat{\nabla} \phi_l = \sum_{c=1}^{ns} \bar{\mu}_c \bar{\phi}_c$$

/ Reconstruction in Left stencil centred on L

$$\frac{\partial \psi}{\partial a_{\{ijk\}}} = \sum_{s=1}^{ns} \varpi_s \left(-2 \Omega_{c(s)} \mathfrak{R}_{c(s)}^{ijk} \bar{\phi}_{c(s)} + 2 \mathfrak{R}_{c(s)}^{ijk} a_{\{ijk\}} + 2 \mathfrak{R}_{c(s)}^{ijk} \sum_{\substack{\{i'j'k'\} \\ \{ijk\} \neq \{i'j'k'\}}} \mathfrak{R}_{c(s)}^{i'j'k'} a_{\{i'j'k'\}} \right) = 0$$

$$\mathfrak{R}_c^{ijk} = \int_{\Omega_c} X^i Y^j Z^k dV : \text{Volume moment of order } ijk$$

$$\mathbf{M}_{\{ijk\}, c(s)} \bar{\phi}_{c(s)} + P_{\{ijk\}, \{i'j'k'\}} a_{\{i'j'k'\}} = 0 \implies a_{\{ijk\}} = \mathbf{K}_{\{ijk\}, c(s)} \bar{\phi}_{c(s)}$$

FV NXO method : Reconstruction and projection

$$\mathbf{M}_{\{ijk\},c(s)} \bar{\phi}_{c(s)} + \mathbf{A}_{\{ijk\},\{i'j'k'\}} \mathbf{a}_{\{i'j'k'\}} = 0$$

$$\mathbf{a}_{\{ijk\}} = \mathbf{K}_{\{ijk\},c(s)} \bar{\phi}_{c(s)}$$

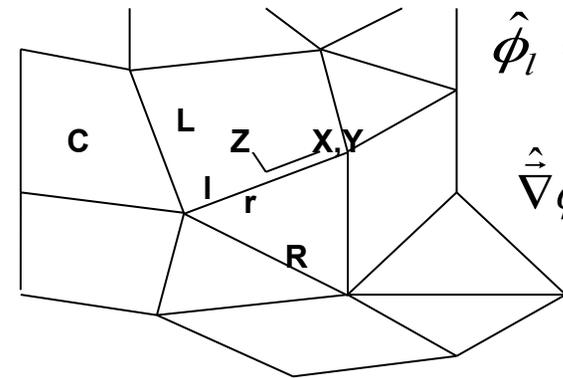
2/ Projection on the interface

$$\hat{\phi}_L = \frac{1}{S_{LR}} \int_{\partial\Omega_{LR}} \phi_a(X, Y, Z) dS = \frac{\int_{\partial\Omega_{LR}} X^i Y^j Z^k dS}{S_{LR}} \mathbf{a}_{\{ijk\}} = \mathbf{v}_{\{ijk\}} \mathbf{a}_{\{ijk\}}$$

$$\hat{\nabla} \phi_L = \frac{1}{S_{LR}} \int_{\partial\Omega_{LR}} \vec{\nabla} \phi_a(X, Y, Z) dS = (i v_{\{i-1jk\}} \vec{e}_X + j v_{\{ij-1k\}} \vec{e}_Y + k v_{\{ijk-1\}} \vec{e}_Z) \mathbf{a}_{\{ijk\}} = \vec{\eta}_{\{ijk\}} \mathbf{a}_{\{ijk\}}$$

$$\hat{\phi}_L = \mathbf{v}_{\{ijk\}} \mathbf{a}_{\{ijk\}} = \mathbf{v}_{\{ijk\}} \mathbf{K}_{g\{ijk\},c} \bar{\phi}_c = \lambda \bar{\phi}_c$$

$$\hat{\nabla} \bar{\phi}_L = \vec{\eta}_{\{ijk\}} \mathbf{a}_{\{ijk\}} = \vec{\eta}_{\{ijk\}} \mathbf{K}_{g\{ijk\},c} \bar{\phi}_{c(s)} = \vec{\mu} \bar{\phi}_c$$



$$\hat{\phi}_l = \sum_{c=1}^{ns} \lambda_c \bar{\phi}_c$$

$$\hat{\nabla} \phi_l = \sum_{c=1}^{ns} \vec{\mu}_c \bar{\phi}_c$$

FV NXO method : Inviscid fluxes options

$\bar{\phi}$: Volume average

$\hat{\phi}$: surface average

$\hat{\phi}(\bar{\phi})$: NXO scheme

$$\frac{\partial(\Omega \bar{W})}{\partial t} + \sum_{n=1}^{ni} S_n (\hat{F}_i - \hat{F}_v)_n = 0$$

2 options for the inviscid fluxes
Characteristic Upwind or centred

$$\hat{F}_i = F_i(\hat{W}_l(\bar{W}), \hat{W}_r(\bar{W}))$$

Upwind scheme : one average flux evaluation from the left and right extrapolated average conservative variables, characteristic splitting 'state upwind'

$$\hat{F}_i = \hat{F}_{il}(\bar{F}) - \tau \theta_{\max} (\hat{W}_l(\bar{W}) - \hat{W}_r(\bar{W}))$$

*Centred scheme : interpolation of the cell-average flux density tensor in all cells of the stencil to the interface
+ a stabilization term*

Main inaccuracy sources :

$$\hat{F} = F(\hat{W})$$

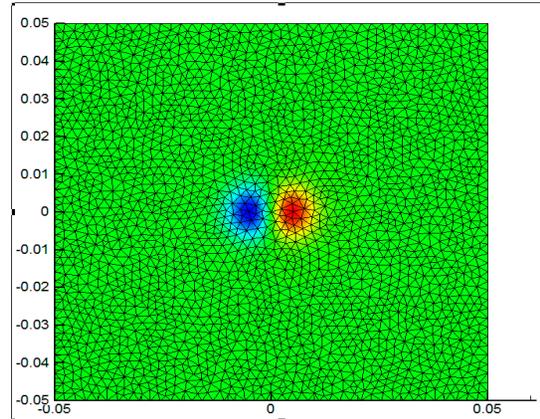
Upwind scheme

$$\bar{F} = F(\bar{W})$$

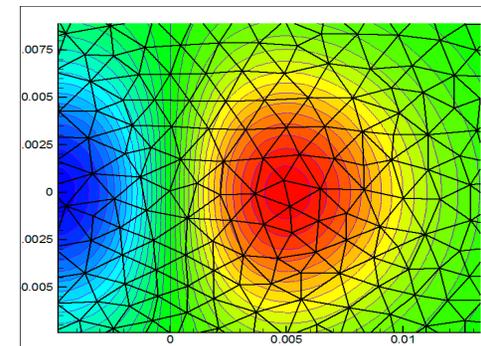
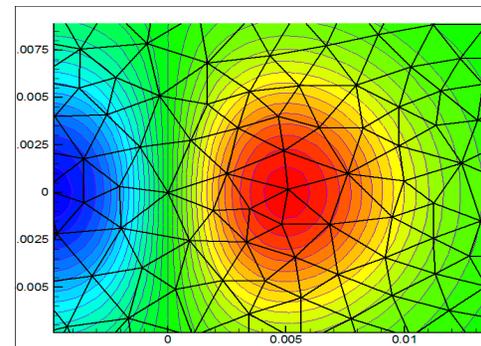
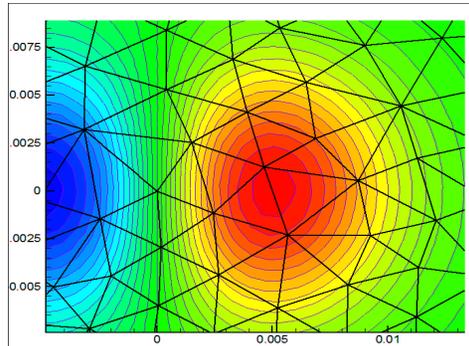
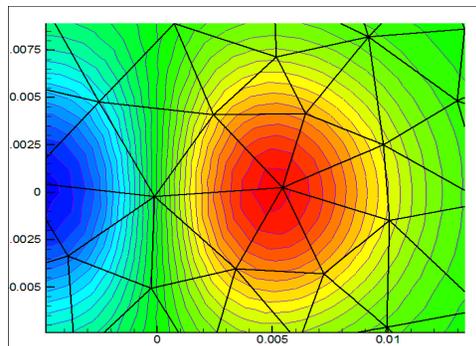
Centred scheme

Non linear effects in flux integration

Total error over the grid due to the flux evaluation from single projected values
(interface integral of the reconstructed polynomial)



Field of y-momentum



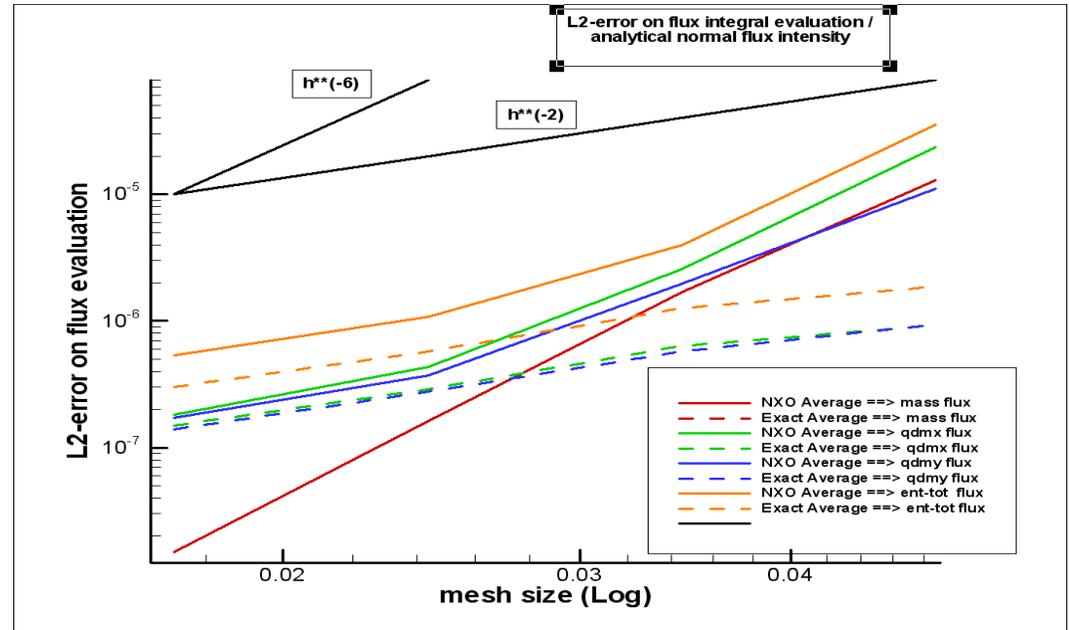
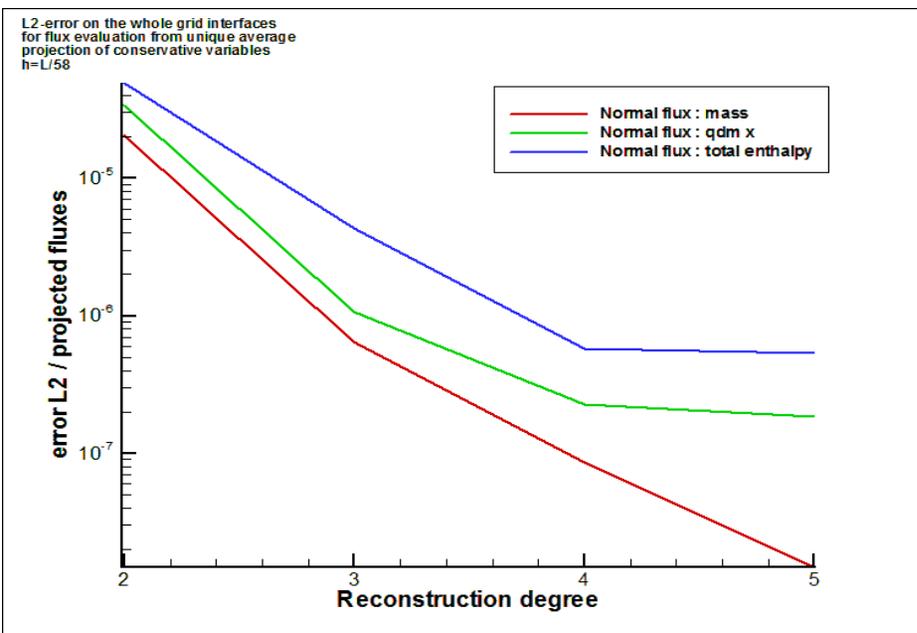
Test case : Initial field for the isentropic vortex transport on the interfaces

Comparison between :

- Analytical expressions for the conservative variables and the normal flux densities
- High order polynomial reconstructions in the cells and along the interfaces from exact cell averages

Non linear effects in flux integration

Total error over the grid due to the flux evaluation from one single projected value (interface integral of the reconstructed polynomial)



Total error over the cell interfaces of the fine grid
Upwind scheme

Convergence with the reconstruction degree

Mesh convergence index for reconstruction k5 :
Optimal order for the fluxes of continuity equation (linear in the conservative variables), loss for other equations

Gain with respect to the second order process of replacing the integrand by its mean value (dotted lines for the analytical evaluations, solid for the NXO method)

High order chimera interpolation

Evaluation of the integral of the reconstructed polynomial over a moving target cell

$$\mathbf{a}_{\{ijk\}} = \mathbf{K}_{\{ijk\},c} \bar{\phi}_{c(s)}$$

The array \mathbf{K} was stored on all candidate source stencils

It gives the coefficients of the polynomial in the source reference frame, from the discrete values in the stencil

$$\widehat{\phi}_t = \frac{1}{V_t} \int_{\Omega_t} \phi_{a,s}(X - X_s, Y - X_s, Z - X_s) dV = \frac{\int_{\Omega_t} (X - X_s)^i (Y - Y_s)^j (Z - Z_s)^k dV}{V_t} a_{\{ijk\}}$$

is the volume average of the polynomial on the target cell

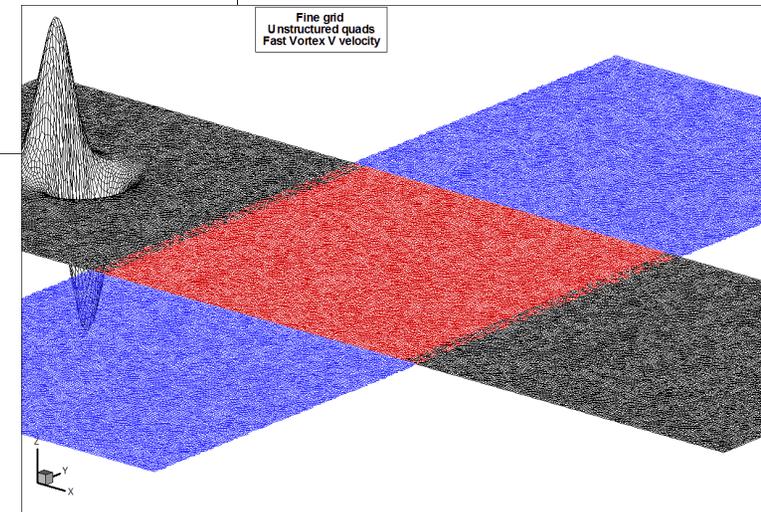
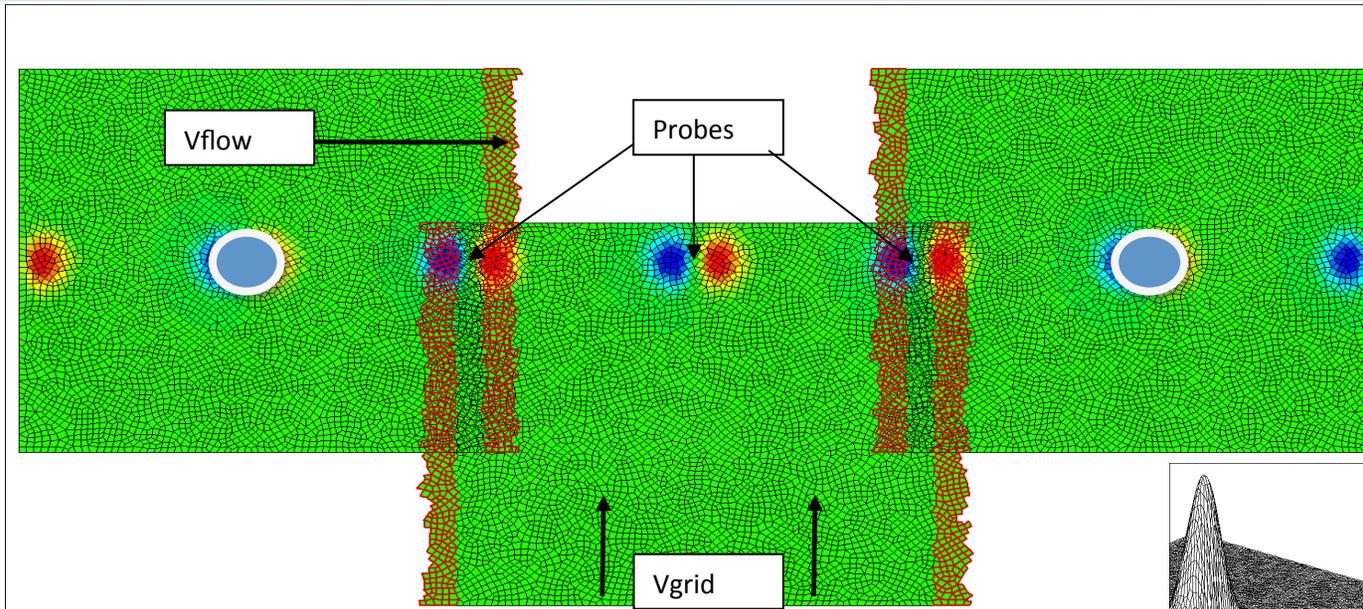
$$\widehat{\phi}_t = \frac{\int_{\Omega_t} (X - X_t + \delta_X)^i (Y - Y_t + \delta_Y)^j (Z - Z_t + \delta_Z)^k dV}{V_t} a_{\{ijk\}} \quad \mathfrak{R}_t^{ijk} = \int_{\Omega_t} (X - X_t)^i (Y - Y_t)^j (Z - Z_t)^k dV$$

The high-order volume moments are known if the local reference frame of the target cell :
need to transport then to the source-local reference frame (linear combination)

$$(\delta_X, \delta_Y, \delta_Z) = (X_t - X_s, Y_t - Y_s, Z_t - Z_s) \quad \widehat{\phi}_t = \beta_{\{i'j'k'\}}^{\{ijk\}} \mathfrak{R}_t^{\{i'j'k'\}} a_{\{ijk\}} = \chi_{\{ijk\}} a_{\{ijk\}}$$

High order chimera interpolation

Test case of the convection of an isentropic in grids in relative motion



Reference fluid conditions : atmospheric air $P=100\ 000\text{Pa}$, $T=300\text{K}$

2 flow conditions :

« Slow vortex » Mach 0.05, Vortex strength $1/50 \rightarrow$ core depression $0,07\text{Pa}$

« Fast vortex » Mach 0.5, Vortex strength $1/5 \rightarrow$ core depression 698Pa

2 types of grids : Cartesian, unstructured quads

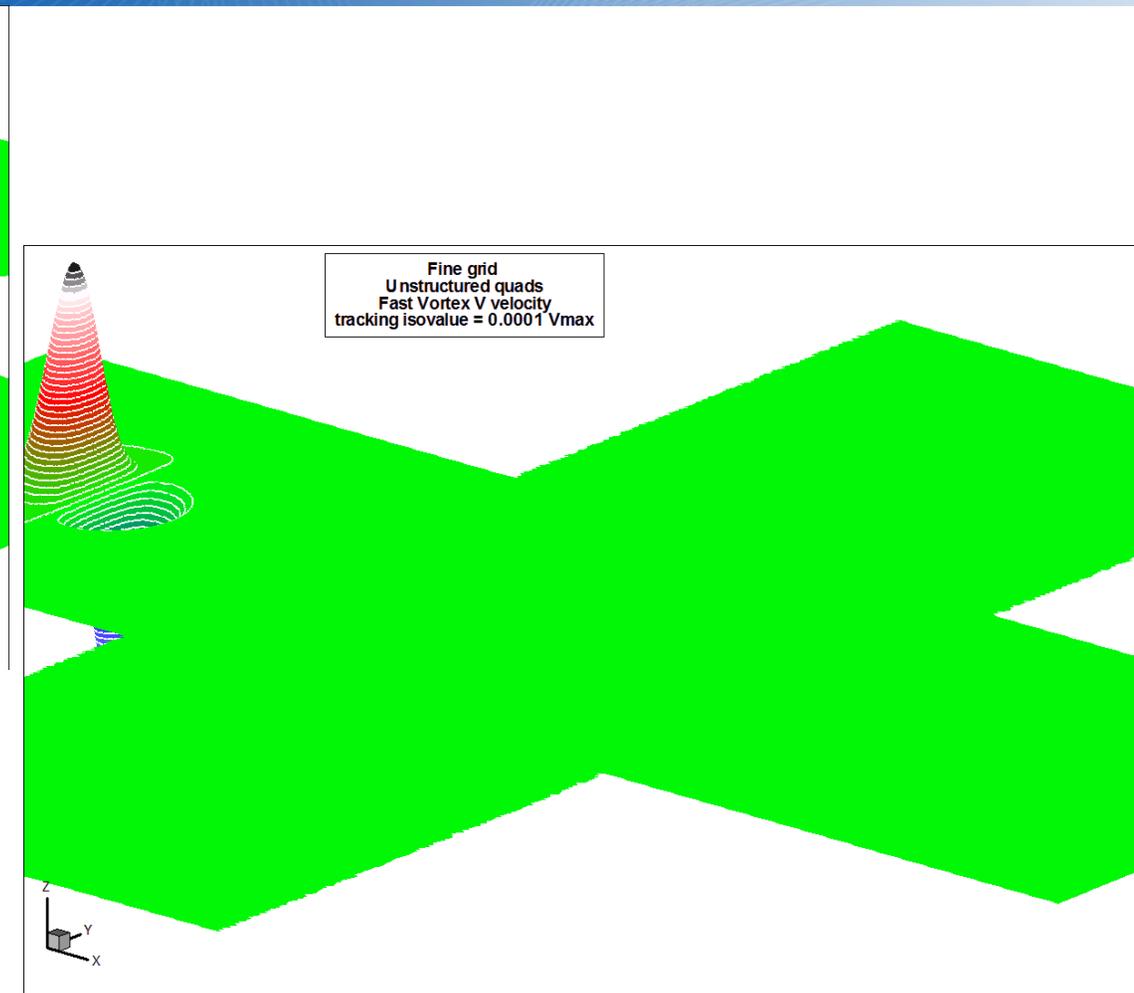
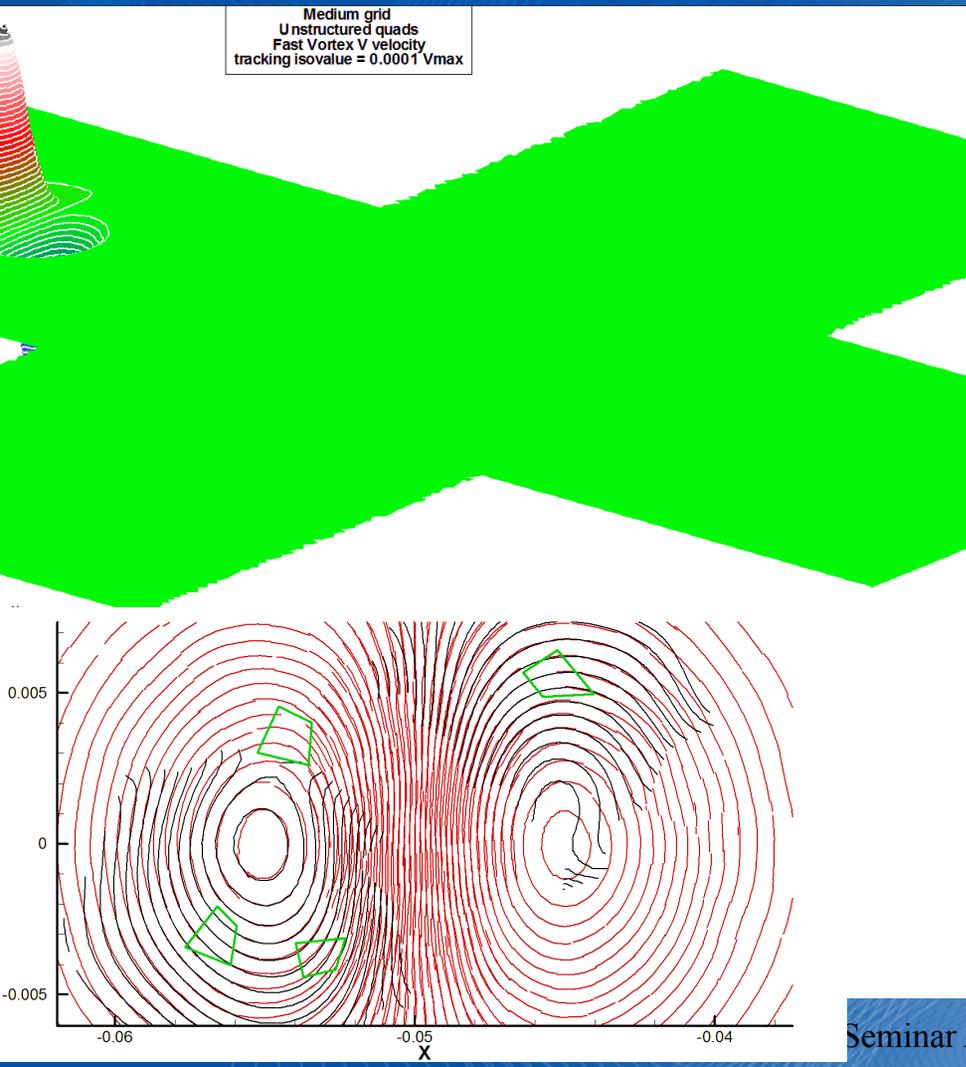
3 mesh sizes : $L/32$, $L/64$, $L/128$

3 velocities of the central grid : 0, 50, 300 m/s

NASA AMS Seminar April 5th 2016

High order chimera interpolation

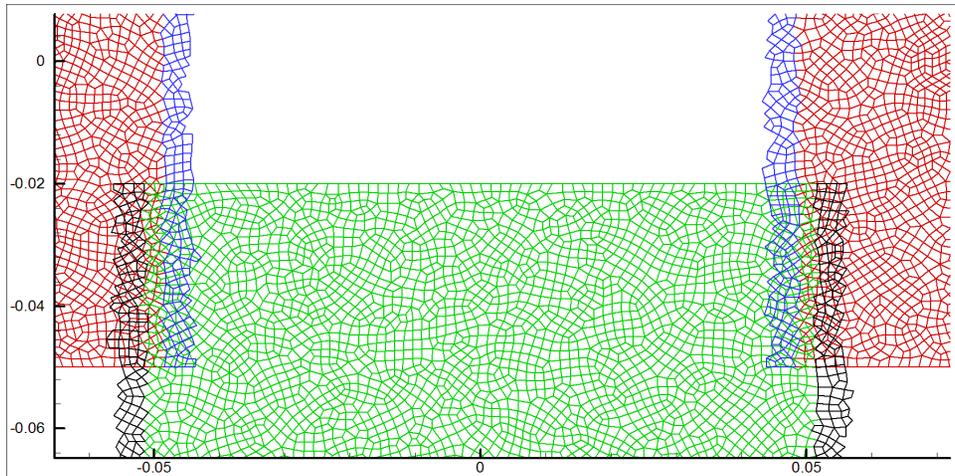
Test case of the convection of an isentropic in grids in relative motion



Seminar April 5th 2016

High order chimera interpolation

Test case of the convection of an isentropic vortex in grids in relative motion



Overset grids, with buffer zone where the solution is computed twice, and interpolated zone (target cells in blue and black) : width equal to the number of neighbours of the stencil build-up

Convection over distance $2L$ from left to right, 3 grid sizes

Time integration of the Euler equations : R-K 3 stages (Shu-Osher), CFL = 0.55

L2 and Linf error on the final flow field with respect to the exact analytical solution

Grid convergence : fields of U and V (L2 norm error), P (L2 and Linf)

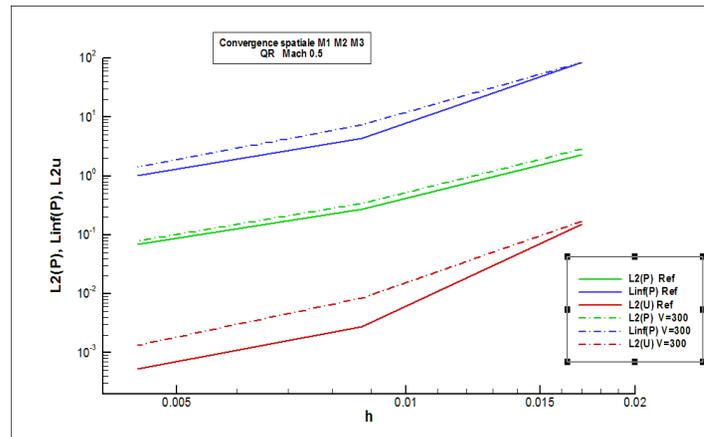
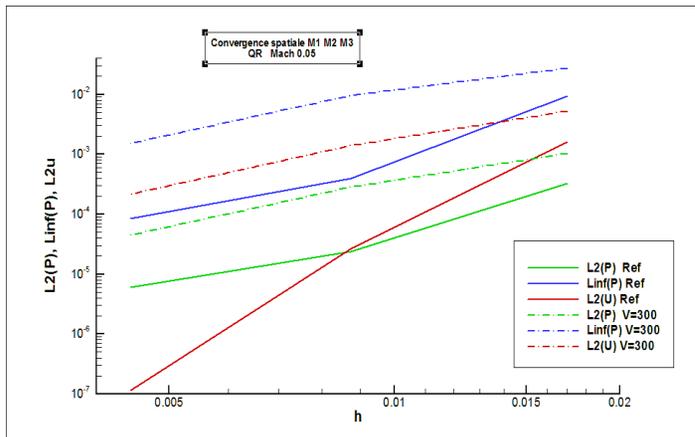
Cartesian grids
Reconstruction k_5
Fixed grids and maximum grid velocity 300m/s

Left : slow vortex Cv index

L2(U) : from 6 (fixed grid) to 2,5 (Vg=300m/s)
L2(P) : from 2,5 - 2,5
Linf(P) : from 2,9 - 2,5

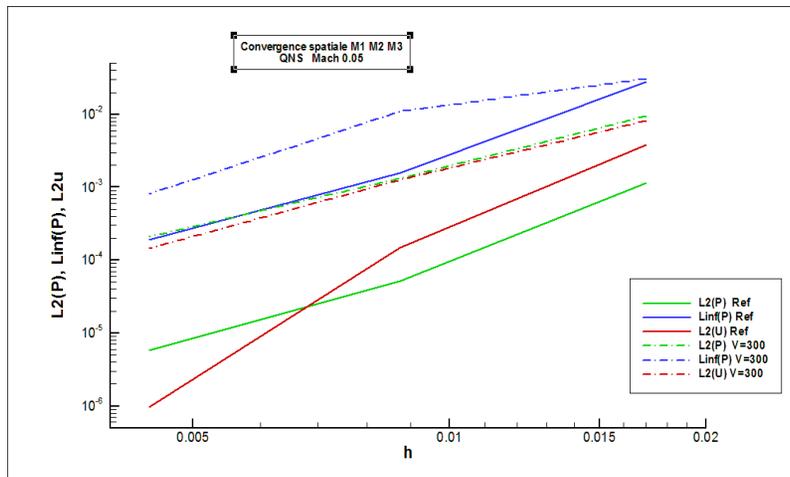
Right : Fast vortex

L2(U) : from 5 (fixed grid) - 2,5 (Vg=300m/s)
L2(P) : from 2,8 - 2,3
Linf(P) : from 3,3 - 2,5



High order chimera interpolation

Test case of the convection of an isentropic vortex in grids in relative motion



Grid convergence

Unstructured quads

Reconstruction k5

Fixed grids and maximum grid velocity 300m/s

Slow vortex Cv index

$L2(U)$ 6,5(fixed grid)	-	3,2 ($Vg=300m/s$)
$L2(P)$ 4,4	-	2,5
$Linf(P)$ 4,2	-	2,8

Large convergence indices for the velocity components : 6 for fixed grids, 2.5 -3 for fast moving grids

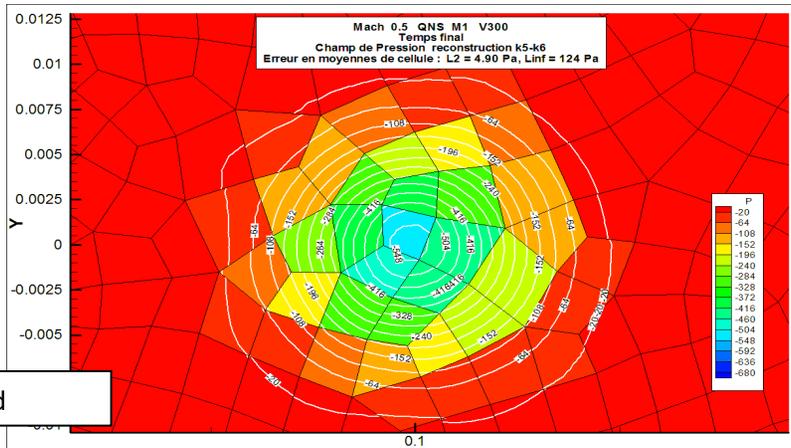
Convergence indices for the pressure : 3 - 4 for fixed grids, 2.5 - 3 for moving grids

The convergence is similar on unstructured grids with respect to cartesian ones, even slightly better on pressure

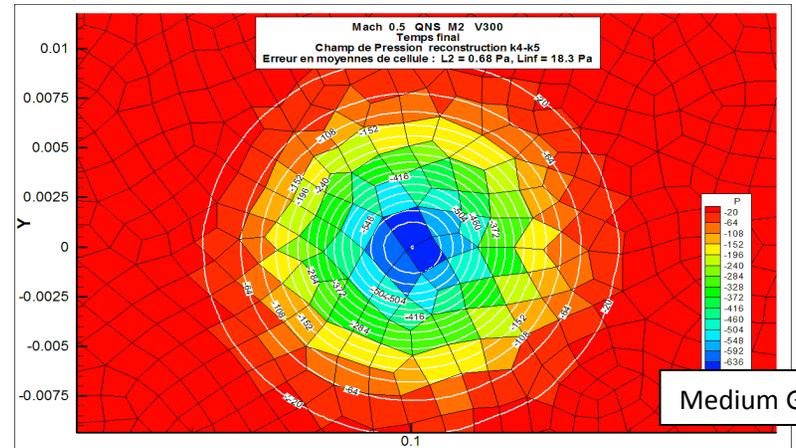
High order chimera interpolation

Test case of the convection of an isentropic vortex in grids in relative motion

Fast vortex case maximum grid velocity unstructured quads



Coarse Grid



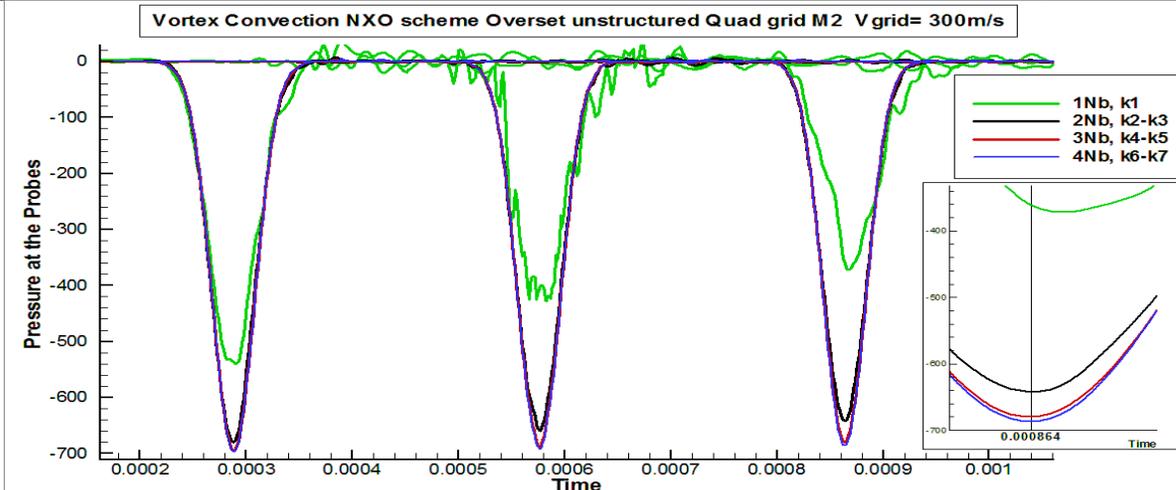
Medium Grid

Final Pressure field (cell average and reconstructed polynomials)

Reconstruction k5
 Maximum grid velocity 300m/s
 Depression = 79% on coarse grid,
 97% on medium grid
 99,6% on fine grid

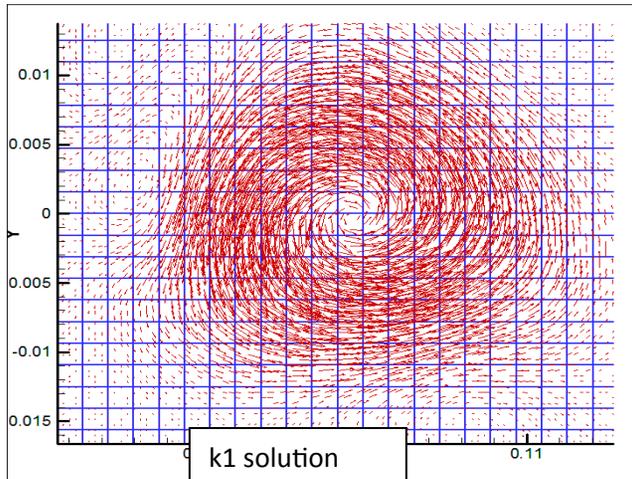
Pressure on the 3 probes on medium grid
 for different reconstruction degrees →

Fast convergence with the spatial order (close-up on probe 3)
 Dispersion errors canceled from k3 up
 Small dissipation (signal lost from probe 1 to probe 3)

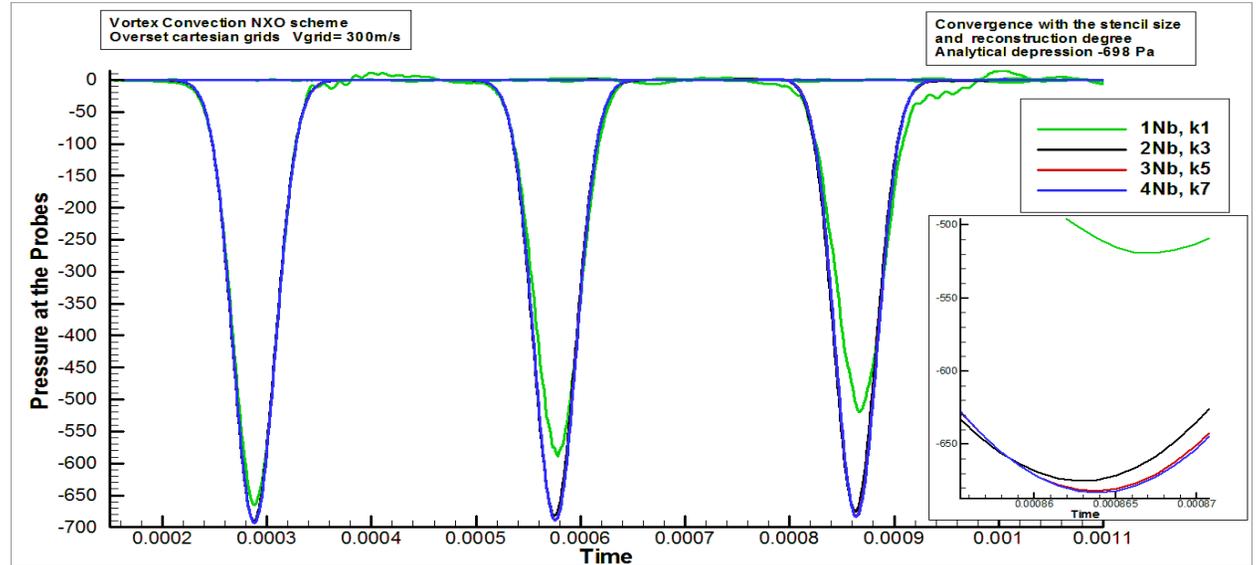
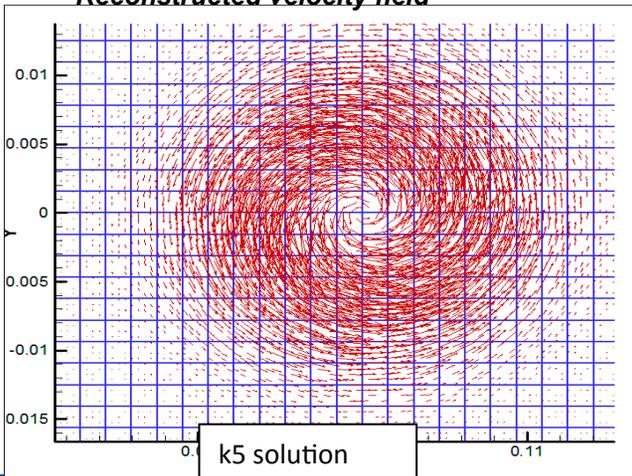


High order chimera interpolation

Test case of the convection of an isentropic vortex in grids in relative motion



Reconstructed velocity field



Pressure on probes : Fast convergence with the spatial order (close-up on probe 3)
Dispersion errors canceled from k5 up
Small dissipation (signal lost from probe 1 to probe 3)

Dispersion is higher than on unstructured grids, dissipation is lower

High order chimera interpolation

Test case of the convection of an isentropic vortex in grids in relative motion

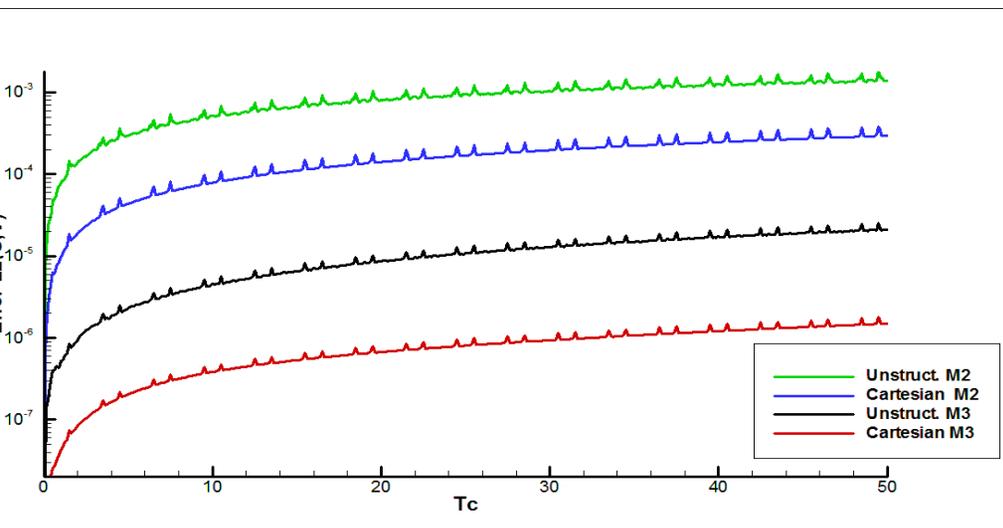
Computation on the medium and fine grids pursued for 50 L/V : crossing of the 2 overset grid boundaries followed by one periodic boundary, 16 times

The overset grid is fixed but shifted in Y to have a non conformal overset interpolation.

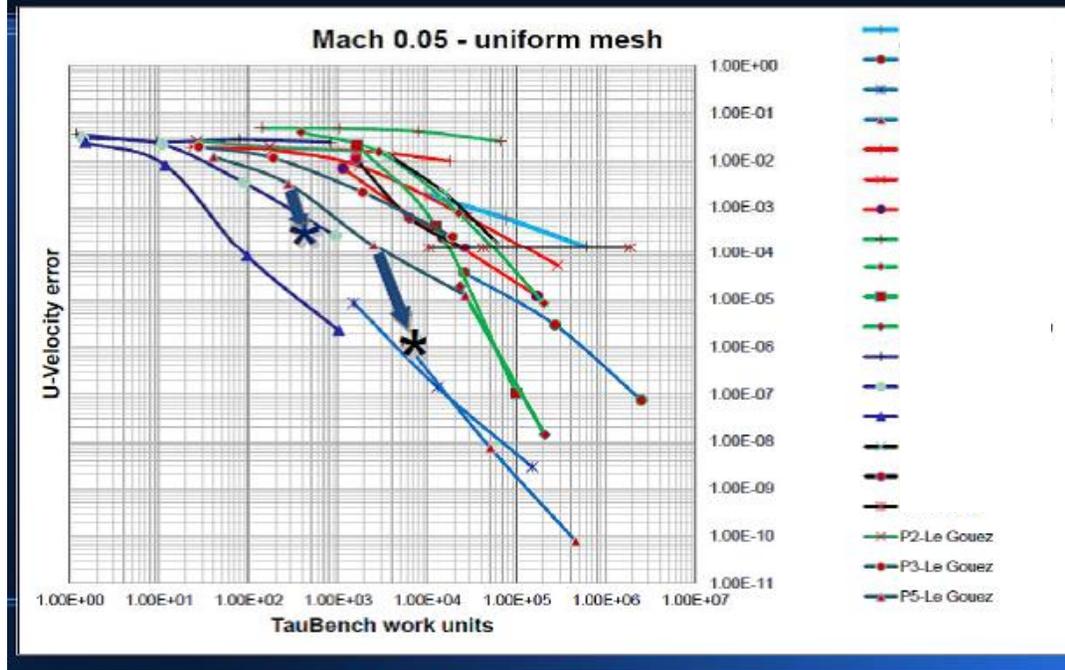
The centred scheme for the fluxes is used.

Comparison with the results of the 2nd HO Workshop (NXO scheme on conformal cartesian periodic grids, with the characteristic upwind scheme)

On the fine grid M3 the error on velocity drops from 1,84e-4 to 1,50e-6 for a cost multiplied by 3 with the centred scheme



Case 1.6 Vortex transport by uniform flow



From the Ref. 2, Convection of an isentropic vortex
 Final error on the velocity components, Conformal Cartesian grids, Slow vortex, 50 Tc
 Stars : centered scheme, red triangles : results obtained with the upwind scheme on grids M2 and M3,

Multi-GPU implementation of a High Order Finite Volume solver

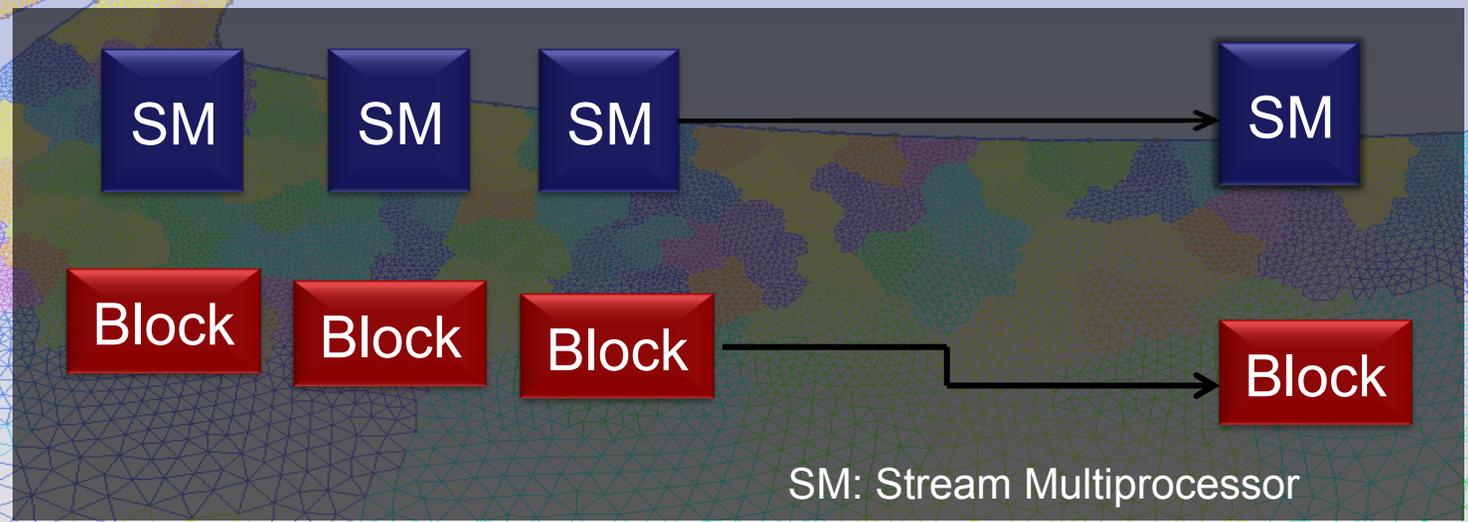
Stages of the project

- Initial porting with the same data model organization than on the CPU
- Generic refinement of coarse triangular elements with curved faces : hierarchy of grids
- Multi-GPU implementation of a highly vectorized model : extruded in the span direction and periodic
- On-going work on a 3D generalization of the preceding phases : embedded grids inside a regular distribution (Octree-type)
- Outlook

1st Approach: Block Structuration of a Regular Linear Grid, cache effects

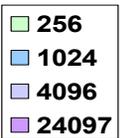
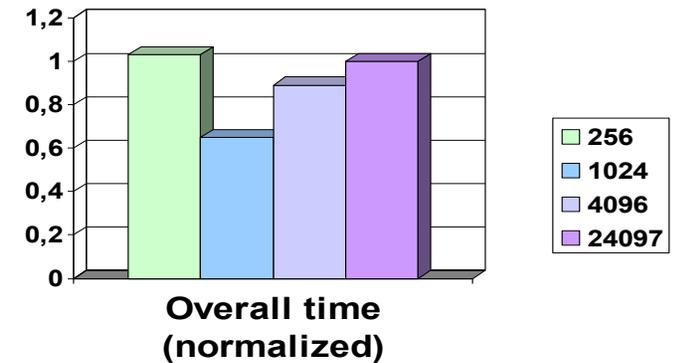
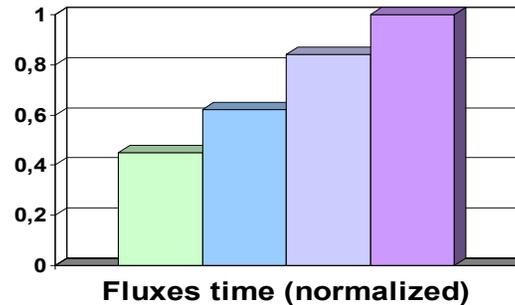
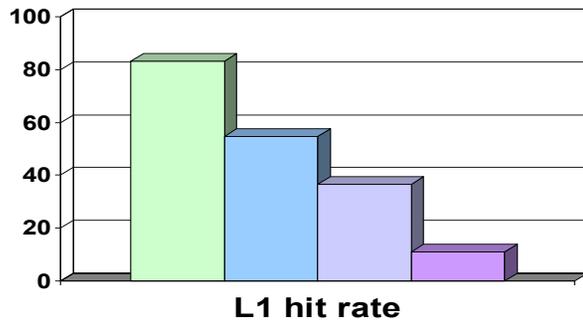
Partition the mesh into small blocks

Map the partitions on the GPU scalable structure



Relative advantage of the small block partition

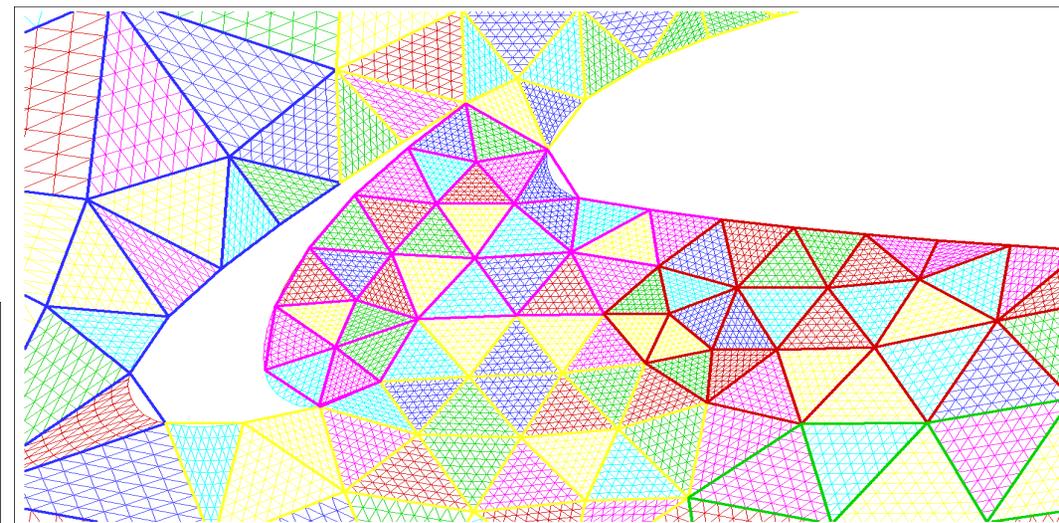
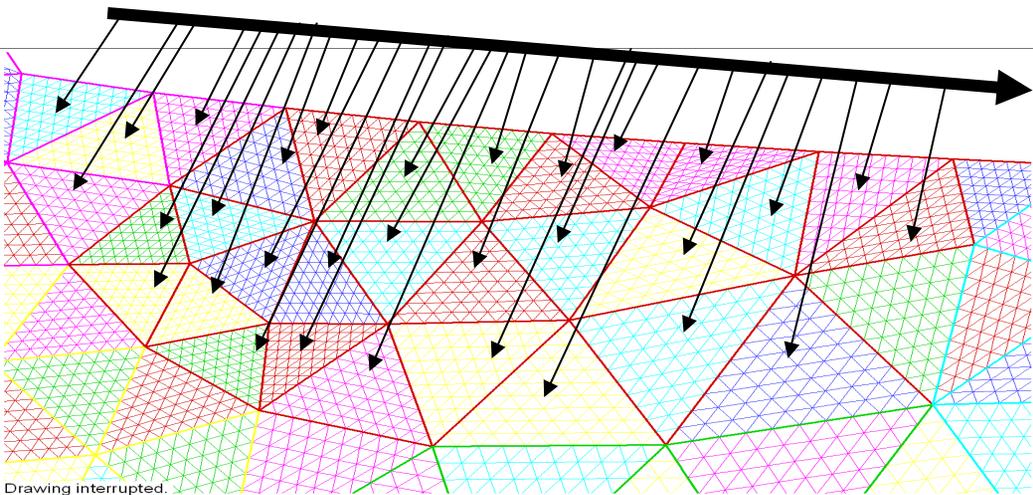
- .Bigger blocks provide
 - . Better occupancy
 - . Less latency due to kernel launch
 - . Less transfers between blocks
- .Smaller blocks provide
 - . Much more data caching



. Final speedup on a TESLA M2050 wrt. to 2 hyperthreaded Westmere CPU: ~2

Imposing a sub-structuration to the grid and data model (inspired by the 'tesselation' mechanism in surface rendering)

Unique grid connectivity for the inner algorithm
Optimal to organize data for *coalescent memory access*
during the algorithm and communication phases
Each coarse element in a block is allocated to an inner thread
(threadId.x)



Hierachical model for the grid : high order (quartic polynomial) triangles generated by gmsh refined on the GPU
the whole fine grid as such could remain unknown to the host CPU

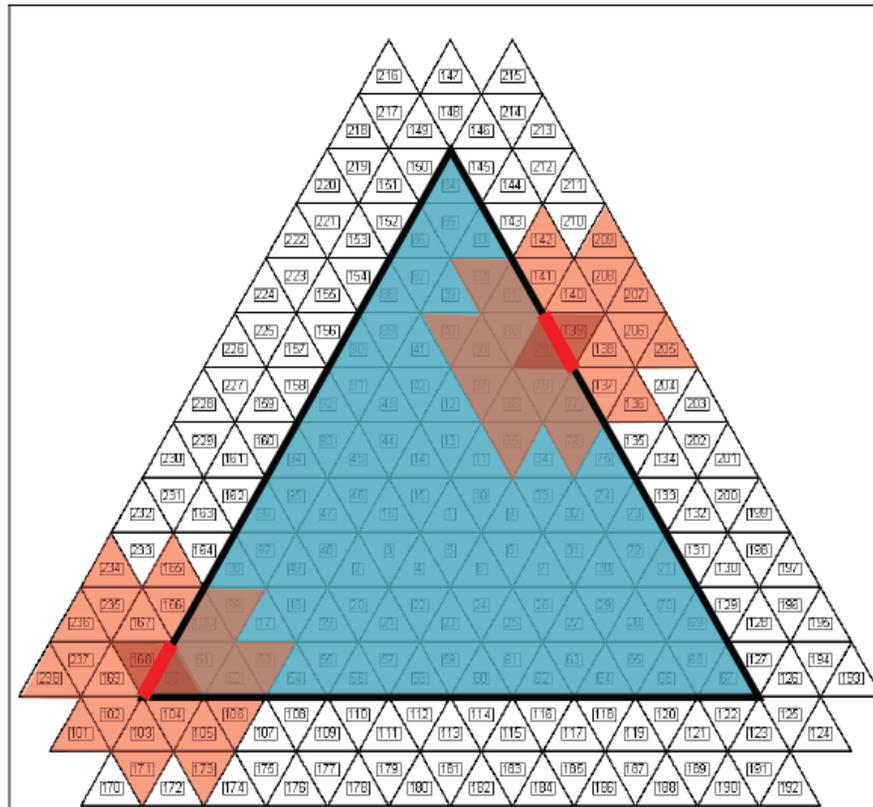
Stencils and Ghost cells

Coupling mechanisms :

Identify ghost cells with real fine cells from neighbor coarse cells (transfer its metrics)

or :

do an overset grid interpolation to obtain the volume average of the conserved variables on the extruded ghost cells



Reference element to be mapped on curvilinear cells

Code structure

Preprocessing

Mesh generation and block and generic refinement generation

Solver

Allocation and initialization of data structure from the modified mesh file

Fortran

Computational routine

Fortran

GPU allocation and initialization binders

C

Computational binders

C

CUDA kernels

CUDA

Time stepping

Data fetching binder

C

Postprocessing

Visualization and data analysis

Version 2 : Measured efficiency on Tesla K20C (with respect to 2 Cpu Xeon 5650, OMP loop-based)

First results on a K20C : Max. Acceleration = 38 wrt to 2 Westmere sockets

Further improvement of the Westmere CPU efficiency : OpenMP task-based

the blocks are refined on the CPU also, then the K20C GPU / CPU acceleration drops to 13 (1 K20c = 150 Westmere cores)

In fact this method is memory bounded, and GPU bandwidth is critical.

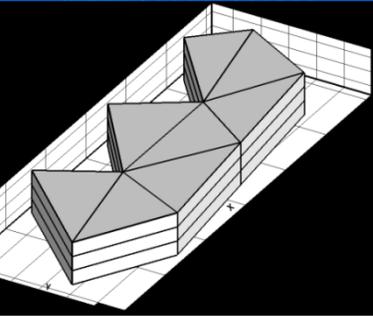
More CPU optimisation needed (cache blocking, vectorisation ?)

Flop count : around 80 Gflops DP /K20C

These are valuable flop, not $Ax=b$, but full non linear Riemann solver flop with high order (4th, 5th) extrapolated values, characteristic splitting, ... : it requires a very high memory traffic to permit theses flops : wide stencils method

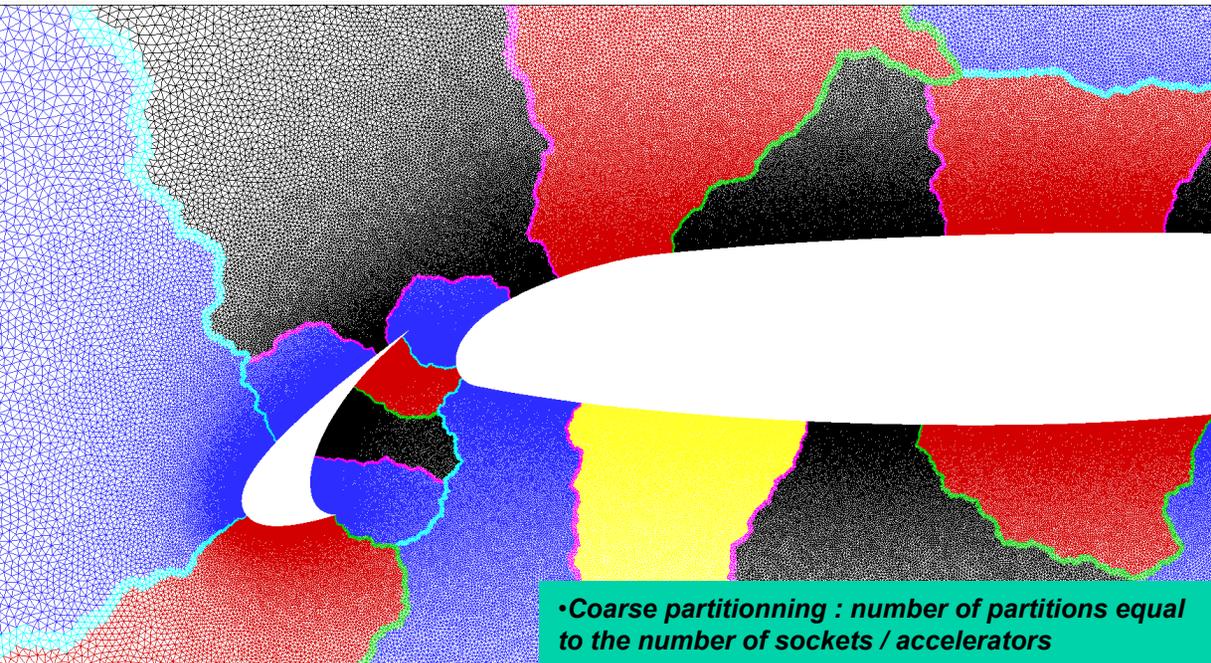
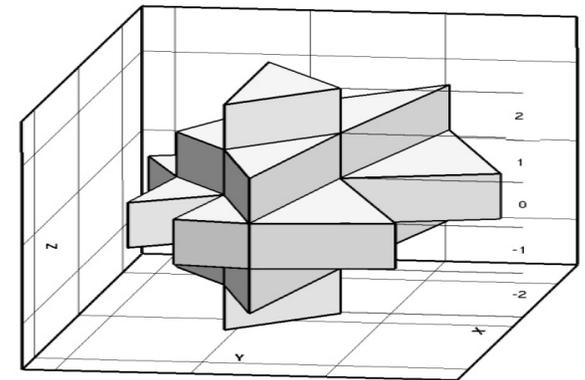
Thanks to the NVIDIA dev-tech department for their support, “ my flop is rich”

Version 3 : 2.5D periodic spanwise (circular shift vectors), MULTI-GPU / MPI



High CPU vectorisation (all variables are vectors of length 256 to 512) in the 3rd homogeneous direction

Full data parallel Cuda kernels with coalesced memory access



•Coarse partitioning : number of partitions equal to the number of sockets / accelerators

Objective : one Billion cells on a cluster with only 64 TESLA K20 or 16 K80

(40 000 cells * 512 spanwise stations per partition : 20 million cells addressed to each TESLA K20)

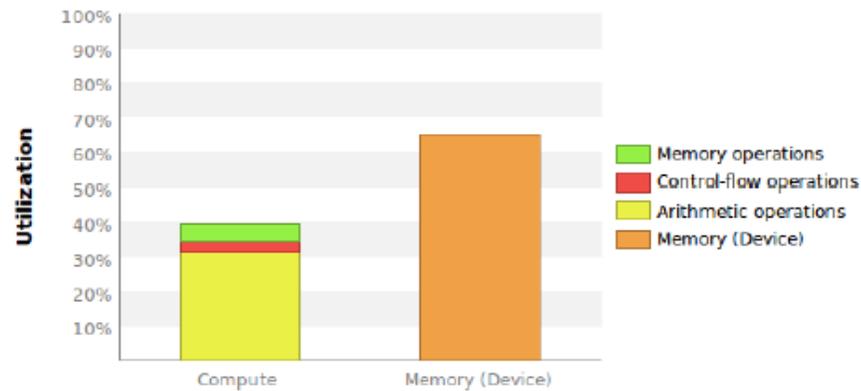
The CPU (MPI / Fortran, OpenMP inner loop-based) and GPU (GPUDirect / C/ Cuda) versions are in the same executable, for efficiency and accuracy comparisons

Version 3 : 2.5D periodic spanwise (cshift vectors), MULTI-GPU / MPI

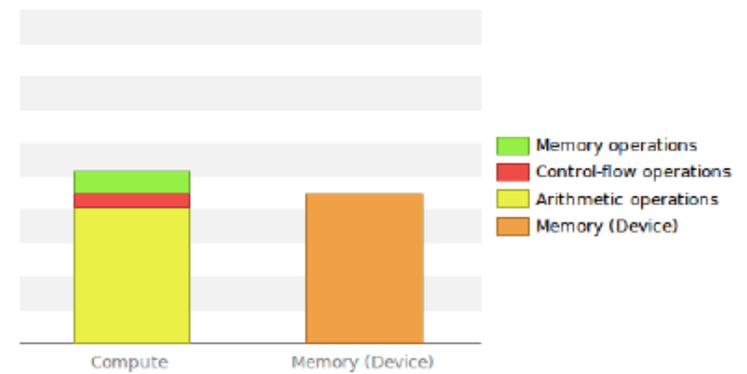
Initial performance measurements

PERFORMANCE LIMITERS

Three top kernels



iorfo=1,2



iorfo=3

Initial Kernel Optimization and analysis performed by NVIDIA DevTech

RESULTS

Comparison with Original code (1 partition)

Original	Current optimizations
Compute : 85.0936872959137 12.8738024234772	Compute : 84.5988802909851 5.95152688026428
Newtime : 2.962398529052734E-002 3.664016723632812E-003	Newtime : 2.988314628601074E-002 3.657817840576172E-003
Newiter : 0.201013803482056 2.434015274047852E-002	Newiter : 0.196562051773071 3.784275054931641E-002
Dtloc : 7.01542568206787 0.916594982147217	Dtloc : 7.00739192962646 0.480051517486572
Timeres : 1.90137290954590 0.292651176452637	Timeres : 1.89031934738159 0.300009727478027
Fluxes : 69.2380857467651 9.93227958679199	Fluxes : 68.8473780155182 4.28139758110046
Fluxbal : 3.26401877403259 0.913108825683594	Fluxbal : 3.21450090408325 0.439968824386597
Update : 3.44401168823242 0.791139841079712	Update : 3.41265749931335 0.408583641052246
Messages : 1.347064971923828E-004 2.384185791015625E-005	Messages : 1.873970031738281E-004 1.502037048339844E-005

8 IVB cores vs K40(ECC ON, GPU Boost ON)

12 NVIDIA

Proposed strategy for further optimization of performances: Increase occupancy, reduce registers' use, reduce amount of operations with global memory

Version 3 : Kernel granularity revised to optimize register use, Overlapping of communications with computations at the centers of the partitions, local memory usage and inner-block thread collaboration

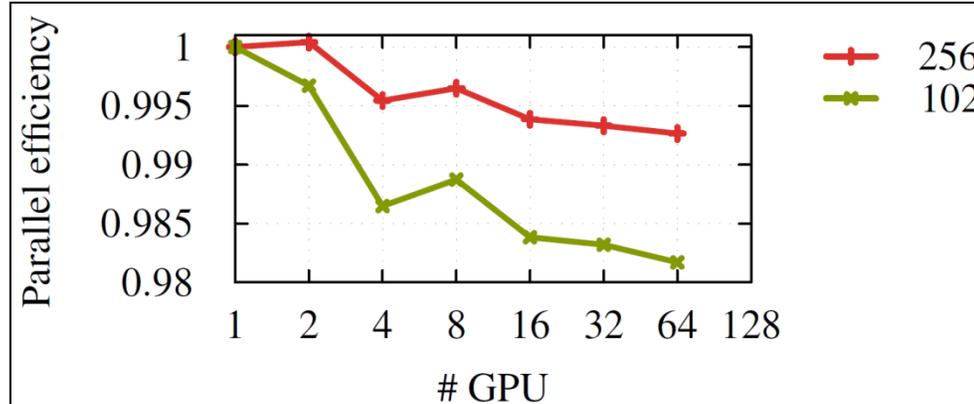
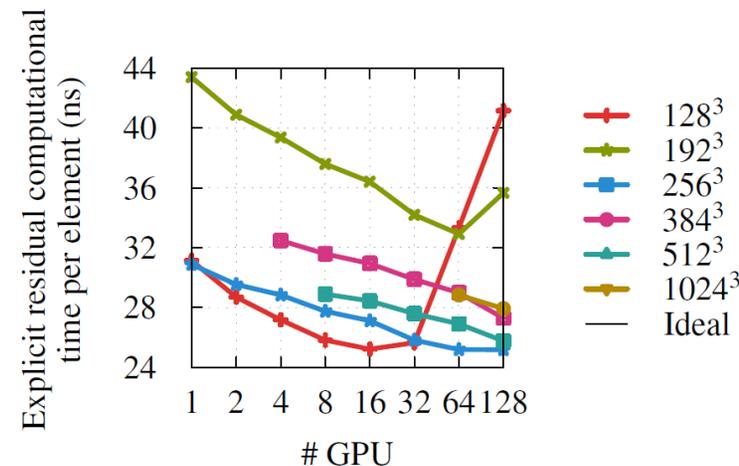
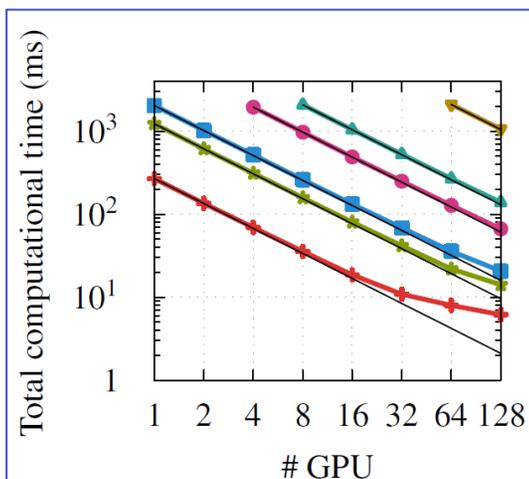
Performance on 1 GPU K20Xm (860 Double Precision units) = 20 Ivy-Bridge (160DP cores, good vector performance of the CPU implementation)

RAYLOR-GREEN Vortex

Scalability analysis with up to one billion cells and 4th degree polynomial reconstruction (5 dof per cell, stencil size 68 cells), on 1 to 128 Gpu K20Xm

High performance : 10 ns to compute one set of 5 fluxes on an interface from a wide stencil of 68 cells : 180 GBytes/s, 170 Gflops DP

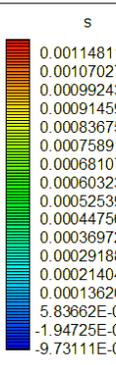
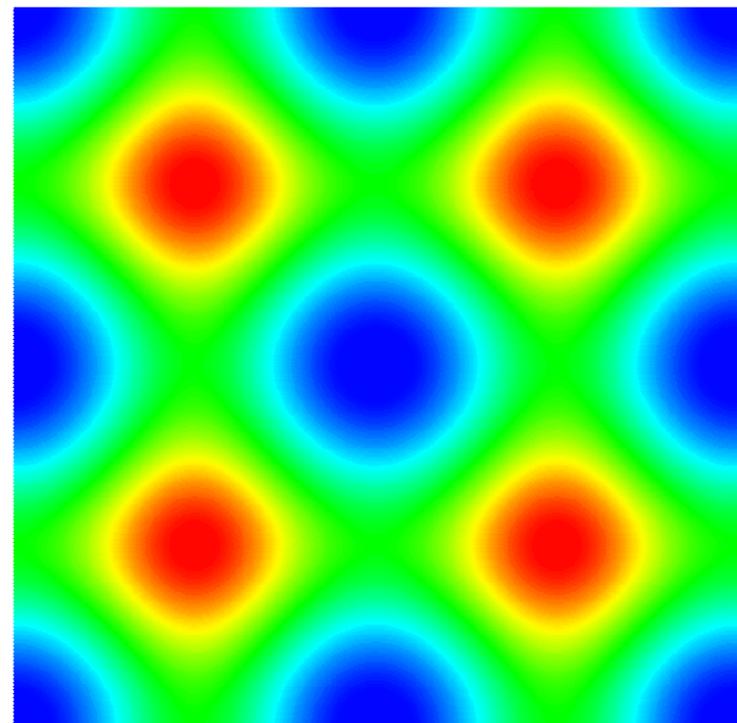
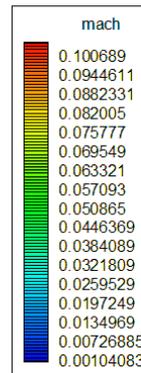
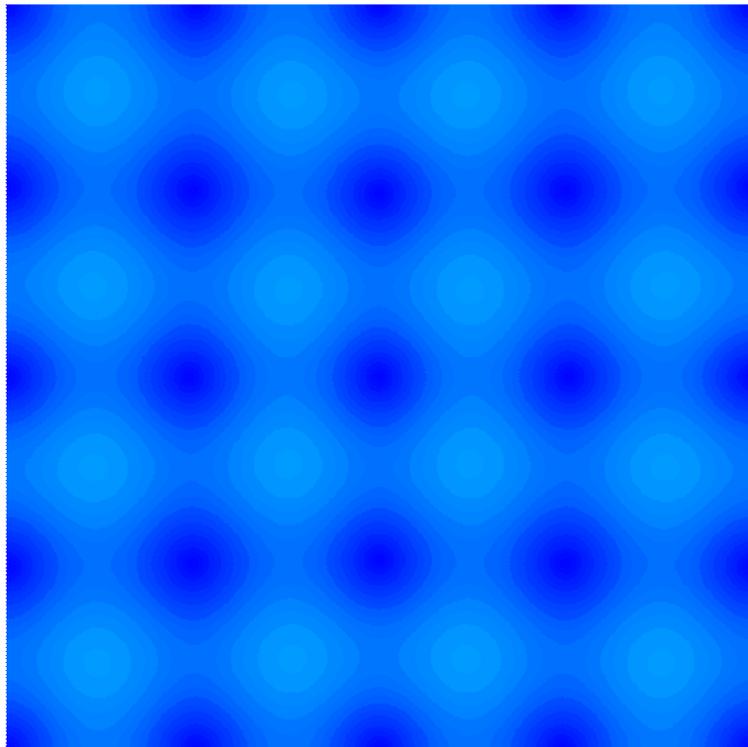
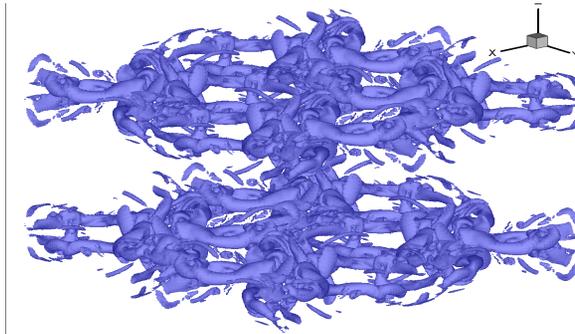
Scalability drops only for extreme degraded usage : small grid 128³ cells on more than 32 GPUs, over 30% of cells to exchange data on interfaces



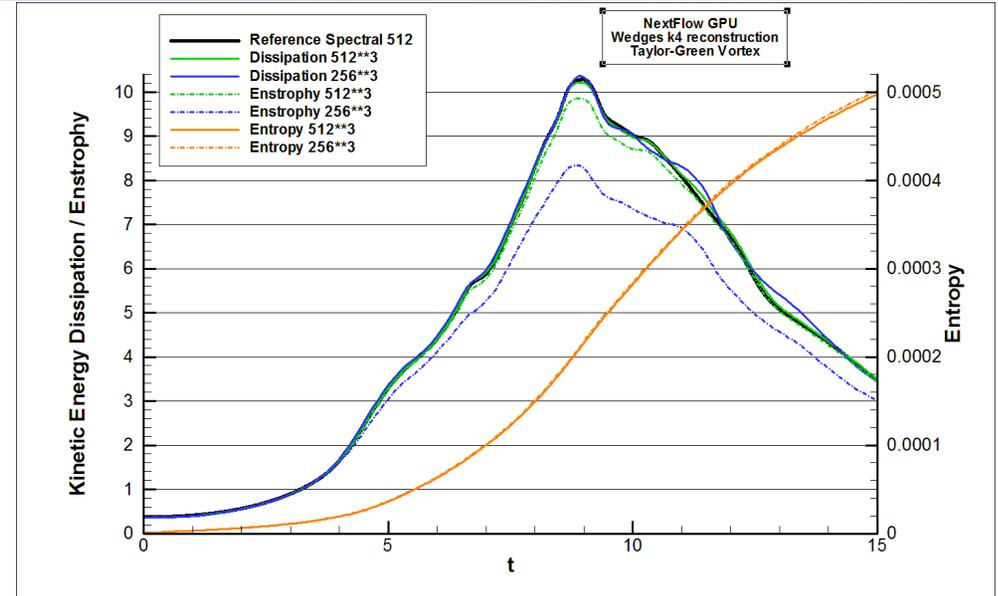
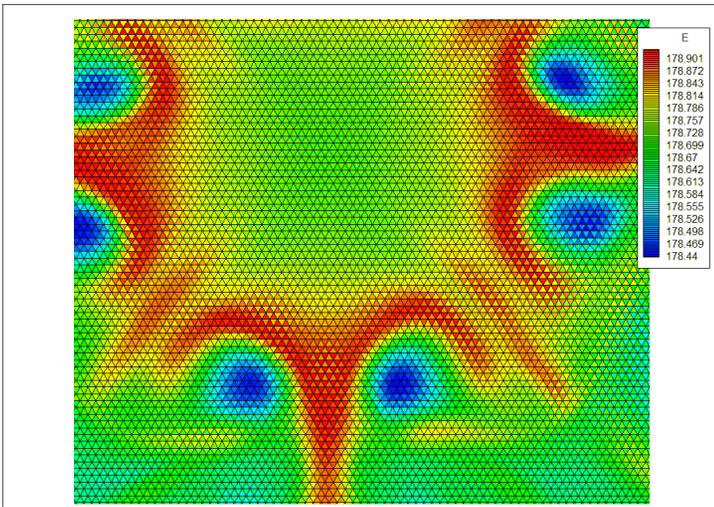
High Order CFD Workshop

Case 3.5 Taylor-Green Vortex

Comparison of time derivatives of enstrophy and scaled
Dissipation of kinetic energy
Occurrence of an acoustic phenomenon



GPU implementation of the NextFlow solver



Taylor-Green Vortex $Re_\tau = 1600$
Computations on wedges

Performance on each K20Xm GPU :

in k3 1,8e-8 s per RHS, 0.36s for 20 000 000 cells

in k4 2,5e-8 s per RHS, 0.50s for 20 000 000 cells

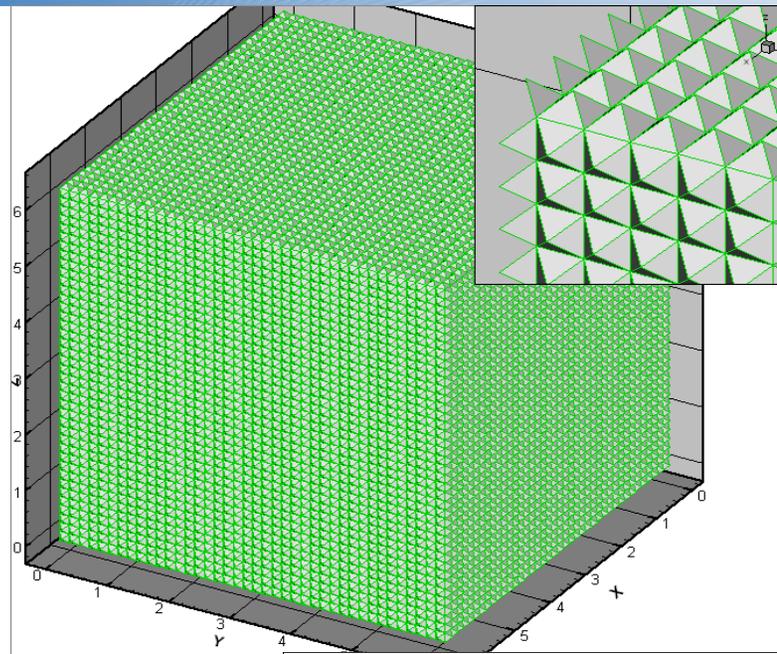
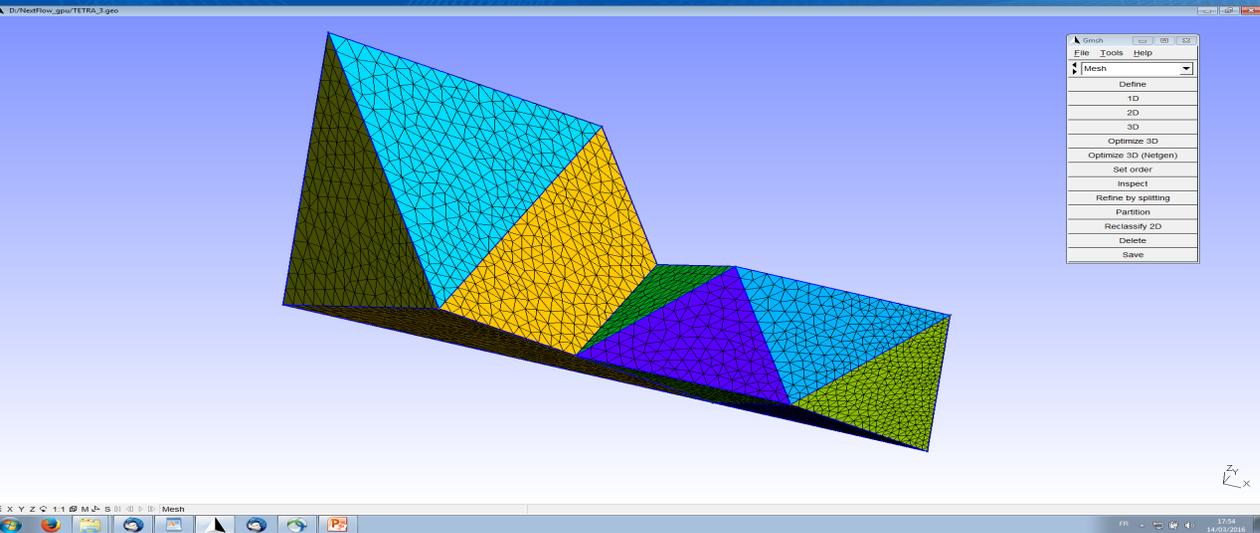
Taylor Green vortex 256**3 - wall-clock = 12 hours on 16 IVY-Bridge processors (total 128 cores) : 1600 hours CPU Intel core
25 minutes on 16 Tesla K20M GPU

By comparison, at the 1st HO CFD workshop , this case requested between 1100 and 3300 Intel core Cpu hours, depending on the numerical method

Taylor Green vortex 512**3 - wall-clock : 4 hours on 16 Tesla K20M GPUs

On-going work

Hierarchical grids based on the generic refinement of a coarse grid of Octree type



All tets are identical, only oriented differently in space

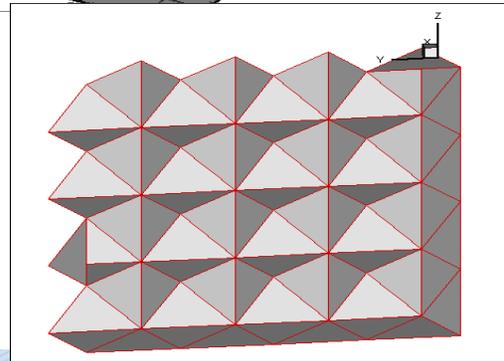
from a grid of very coarse « structured tets » : perform a refinement based on a simple

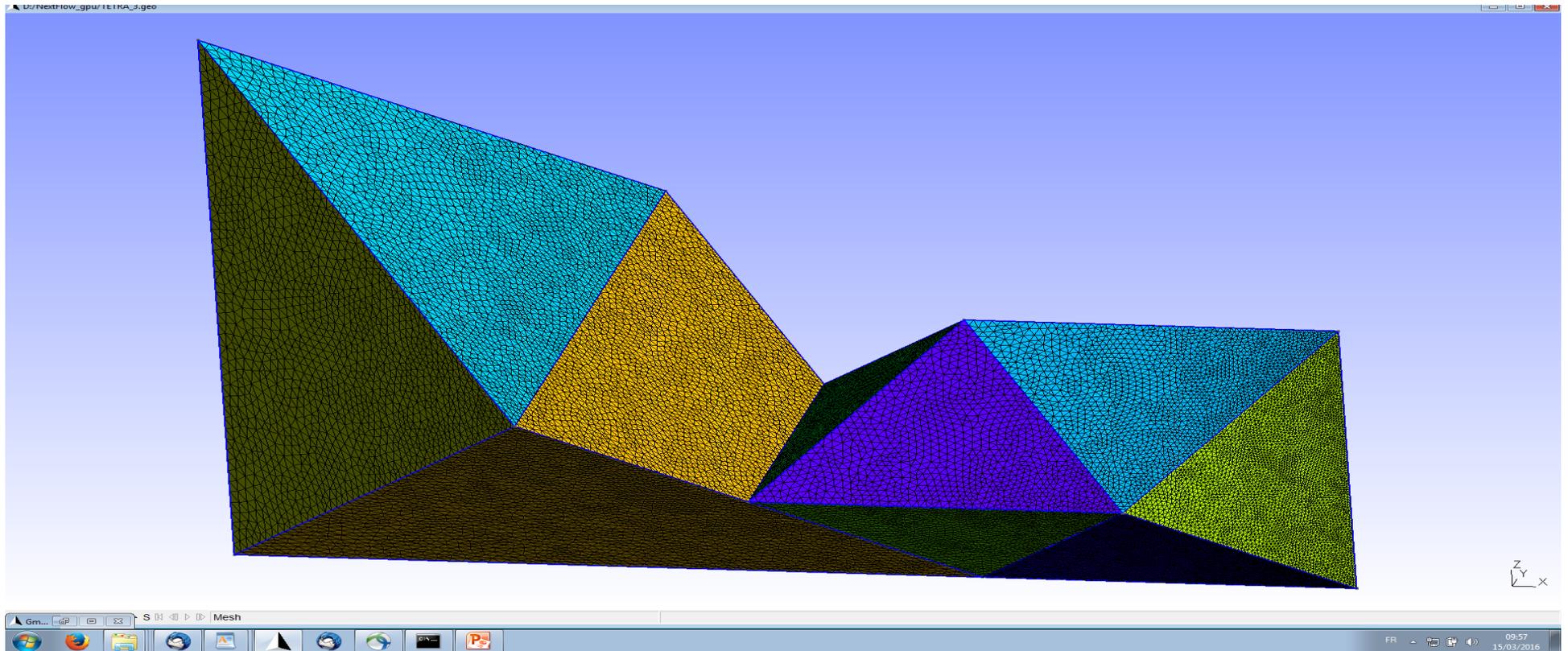
criterion (distance to an object) : $8, 8^2, 8^3 \rightarrow$ Tet-tree 'Coarse' grid , managed, partitioned on the cluster by the thread 0 of each node

Each coarse tet of any size is filled dynamically with small tets : finite volumes for the solver

The size of the inner grid is adapted dynamically to the solution by refinement fronts crossing the coarse edges

The coarse tets are clustered by refinement level : these sets are allotted to the multiprocessors of the accelerators available on nodes





➤ Wall boundary conditions are Immersed Boundary conditions or CAD–Cut cells with curved geometry

➤ Reduced set of filling grids are generated on these simple models (the « tet-farm ») : inner connectivity list, coefficients of the scheme, ghost-cell layers and their correspondence with the inner numbering of next filling grid, HO projection coefficients of the fields when the grid refinement level changes in a coarse element :

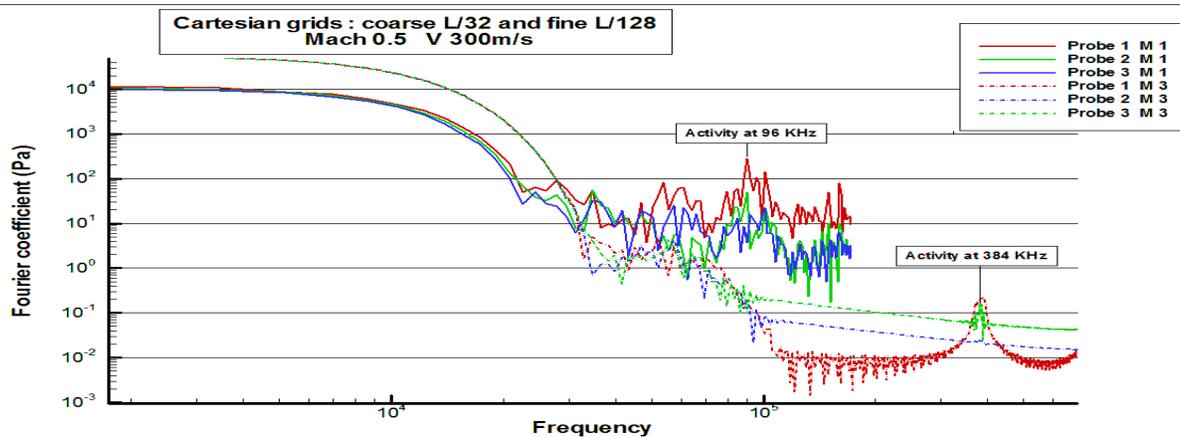
➤ common data model stored on the GPUs and accessed in a coalesced way

Conclusion

A number of preparatory projects enabled to acquire a good expertise on the porting of CFD solvers, their compute intensive kernels and interfaces, and the best organization of the data models for multi-GPU performance.

A project of full software deployment was started for a variety of CFD options and complex 3D geometries, with adaptive grid refinement, on embedded grid systems, without the need for a body-fitted meshing tool.

Cassiopée modules handle the coarse grid management (partitioning, clustering of zones of identical sub-grids, tasking on the accelerators, refinement indices)

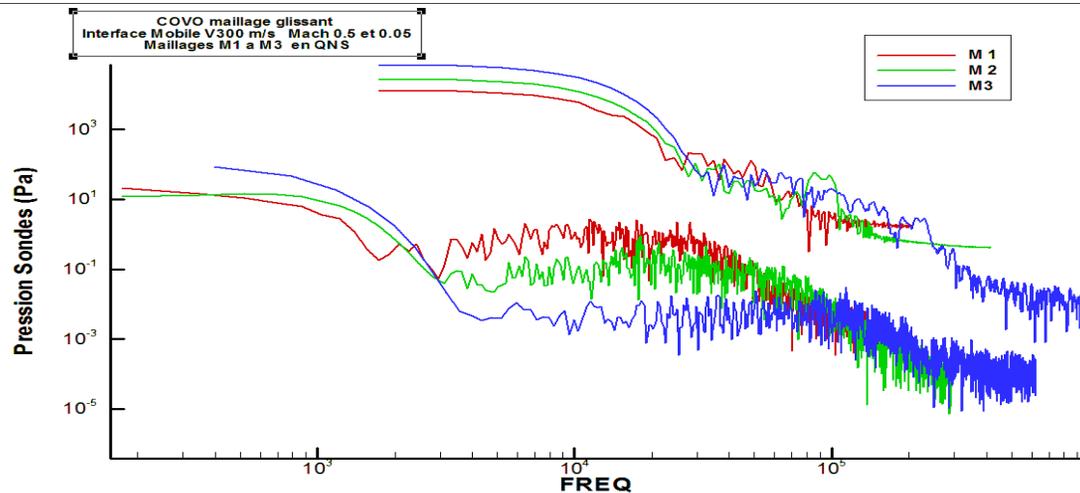


Effect of grid refinement : coarse and fine Cartesian grid

Main frequency = V_{grid}/h : change of source stencil for each target cell

The activity is higher on the probe 1

The peak drops by a factor 1000 from the coarse to the fine grid



Differences between the fast and slow vortex cases on unstructured grids

Fast vortex : higher Fourier coefficients all mesh sizes M1 to M3

Probe 1

Main frequency = h/V_{grid} hardly visible, small peaks shifted to higher frequencies and lower levels than on cartesian grids

More broadband behavior for the slow vortex, mesh convergence also very high : damping by a factor 500 from coarse to fine grid