ECCOMAS 2008

30 June - 4 July, 2008
Venice, Italy

DNS of an air-filled differentially heated cavity of aspect ratio 4 at Ra-numbers from $6{,}4 \times 10^8$ to 10^{11} on MareNostrum supercomputer

F.X. Trias, M. Soria, A. Gorobets and A. Oliva

Centre Tecnològic de Transferència de Calor (CTTC), Universitat Politècnica de Catalunya (UPC)
C/ Colom 11, 08222 Terrassa, Barcelona, Spain, E-mail: cttc@cttc.upc.edu

Presentation outline

1. **Numerical methods for DNS**
   - Symmetry-preserving discretization of Navier-Stokes equations
   - Pressure-velocity coupling. Poisson equation
   - Poisson solver: extended algorithm for fully-3D geometry
2. **Poisson solver**
   - Scalable KSFD algorithm
   - Parallel performance on MareNostrum supercomputer
3. **DNS of turbolence in natural convection**
   - Problem definition: Differentially Heated Cavity
   - DNS results for Ra up to 10^{11} and meshes up to 111M nodes
4. **Conclusions and Future Research**
   - Conclusions
   - Future research

Governing equations

Incompressible Navier-Stokes equations

$$
\begin{aligned}
\nabla \cdot \mathbf{u} &= 0 \\
\partial_t \mathbf{u} + (\mathbf{u} \cdot \nabla)\, \mathbf{u} &= \frac{Pr}{Ra^{0,5}} \nabla^2 \mathbf{u} - \nabla p + f \\
\partial_t T + (\mathbf{u} \cdot \nabla)\, T &= \frac{1}{Ra^{0,5}} \nabla^2 T
\end{aligned}
$$

where $f = (0, 0, RaPrT)$ (Boussinesq approximation)

Finite-volume discretization on an **arbitrary staggered mesh** can be written by

$$
\Omega_s \frac{d\boldsymbol{u}_s}{dt} + \mathsf{C}\left(\boldsymbol{u}_s\right) \boldsymbol{u}_s + \mathsf{D}\boldsymbol{u}_s + \Omega_s \mathsf{G}\boldsymbol{p}_c = \mathbf{0}_s \qquad \mathsf{M}\boldsymbol{u}_s = \mathbf{0}_c
$$

- Ω_s is a diagonal matrix with the sizes of **control volumes**.
- The matrices $\mathsf{C}\left(\boldsymbol{u}_s\right)$ and D are the **convective** and **diffusive** operators, respectively.
- G represents the discrete **gradient** operator
- M is the integral of the **divergence** operator

Symmetry-preserving discretization

Idea behind: It is important to preserve symmetry properties of the underlying differential operators.

$$\Omega_s \frac{d\boldsymbol{u}_s}{dt} + \mathsf{C}\left(\boldsymbol{u}_s\right)\boldsymbol{u}_s + \mathsf{D}\boldsymbol{u}_s + \Omega_s \mathsf{G}\boldsymbol{p}_c = \boldsymbol{0}_s$$

$$\mathsf{M}\boldsymbol{u}_s = \boldsymbol{0}_c$$

$\Longrightarrow$ It can be shown that the **convective matrix** $\mathsf{C}\left(\boldsymbol{u}_s\right)$ has to be **skew-symmetric**,

$$\mathsf{C}\left(\boldsymbol{u}_s\right) + \mathsf{C}^*\left(\boldsymbol{u}_s\right) = 0$$

and the discrete **gradient operator** G has to be exactly equal to

$$\mathsf{G} = -\Omega_s^{-1}\mathsf{M}^*$$

to preserve the inviscid invariants (e.g. kinetic energy) in a discrete sense.

Pressure-velocity coupling (1/3)

Semi-discrete momentum equation is expressed as

$$\frac{\partial \boldsymbol{u}_s}{\partial t} = \boldsymbol{R}\left(\boldsymbol{u}_s\right) - \mathsf{G}\boldsymbol{p}_c$$

where $\boldsymbol{R}\left(\boldsymbol{u}_s\right) = -\mathsf{C}\left(\boldsymbol{u}_s\right)\boldsymbol{u}_s - \mathsf{D}\boldsymbol{u}_s + \boldsymbol{f}_s$

- **Time discretization:**
  - Central difference is used for the time derivative term
  - Fully explicit second-order Adams-Bashforth scheme for $\boldsymbol{R}\left(\boldsymbol{u}_s\right)$
  - Implicit first-order Euler scheme for pressure-gradient term and mass-conservation equation
- **Spatial discretization:** symmetry-preserving discretization.

Pressure-velocity coupling (2/3)

Time-discrete system to be solved:

$$\begin{aligned} \frac{\boldsymbol{u}_s^{n+1} - \boldsymbol{u}_s^n}{\Delta t} &= \frac{3}{2}\boldsymbol{R}^n - \frac{1}{2}\boldsymbol{R}^{n-1} - \mathsf{G}\boldsymbol{p}_c^{n+1} \\ \mathsf{M}\boldsymbol{u}_s^{n+1} &= \boldsymbol{0}_c \end{aligned}$$

Predictor velocity is defined as $\boldsymbol{u}_s^p = \boldsymbol{u}_s^n + \Delta t \left(\frac{3}{2}\boldsymbol{R}^n - \frac{1}{2}\boldsymbol{R}^{n-1}\right)$

$\Longrightarrow$ Then, the unknown velocity is $\boldsymbol{u}_s^{n+1} = \boldsymbol{u}_s^p - \mathsf{G}\tilde{\boldsymbol{p}}_c$

To evaluate $\tilde{\boldsymbol{p}}_c = \Delta t \boldsymbol{p}_c^{n+1}$, **mass conservation** equation **is imposed**

$$\mathsf{M}\boldsymbol{u}_s^{n+1} = \mathsf{M}\boldsymbol{u}_s^p - \mathsf{M}\mathsf{G}\tilde{\boldsymbol{p}}_c = \boldsymbol{0}_c$$

Pressure-velocity coupling (3/3)

Recalling that $\mathsf{G} = -\Omega_s^{-1}\mathsf{M}^*$, this leads to the following **Poisson equation**

$$\underbrace{-\mathsf{M}\Omega_s^{-1}\mathsf{M}^*}_{\mathsf{L}}\tilde{\boldsymbol{p}}_c = \mathsf{L}\tilde{\boldsymbol{p}}_c = \mathsf{M}\boldsymbol{u}_s^p$$

that must be solved to evaluate $\tilde{\boldsymbol{p}}_c$ and then $\boldsymbol{u}_s^{n+1}$

- Note that the **Laplacian operator** is approximated by the matrix

  $$\mathsf{L} = -\mathsf{M}\Omega_s^{-1}\mathsf{M}^*$$

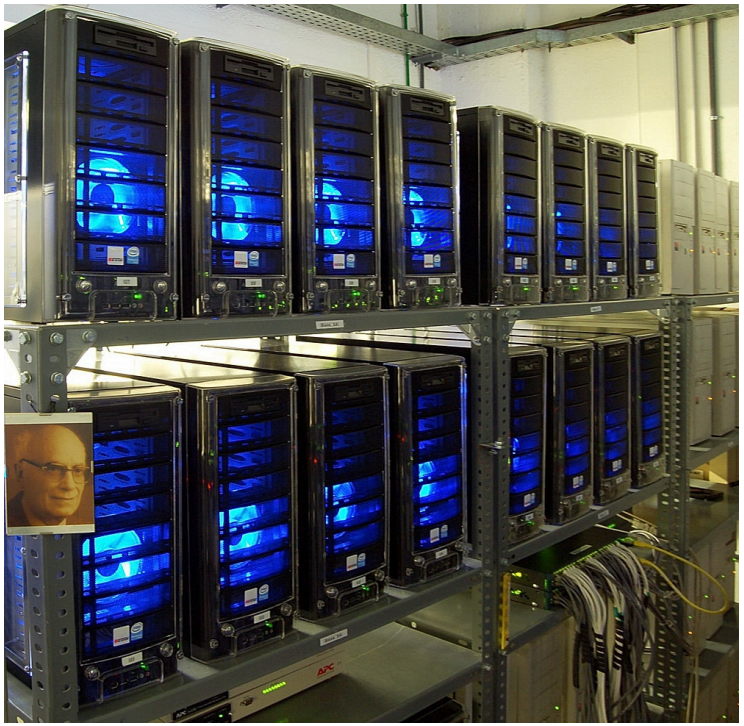
  which is **symmetric** and **negative-definite**, **like the continuous Laplacian operator**.

- This approach is similar in all the segregated formulations for incompressible flows (DNS/LES).

Poisson solver for both PC clusters and supercomputers

Low cost **PC clusters** are loosely coupled parallel computers:

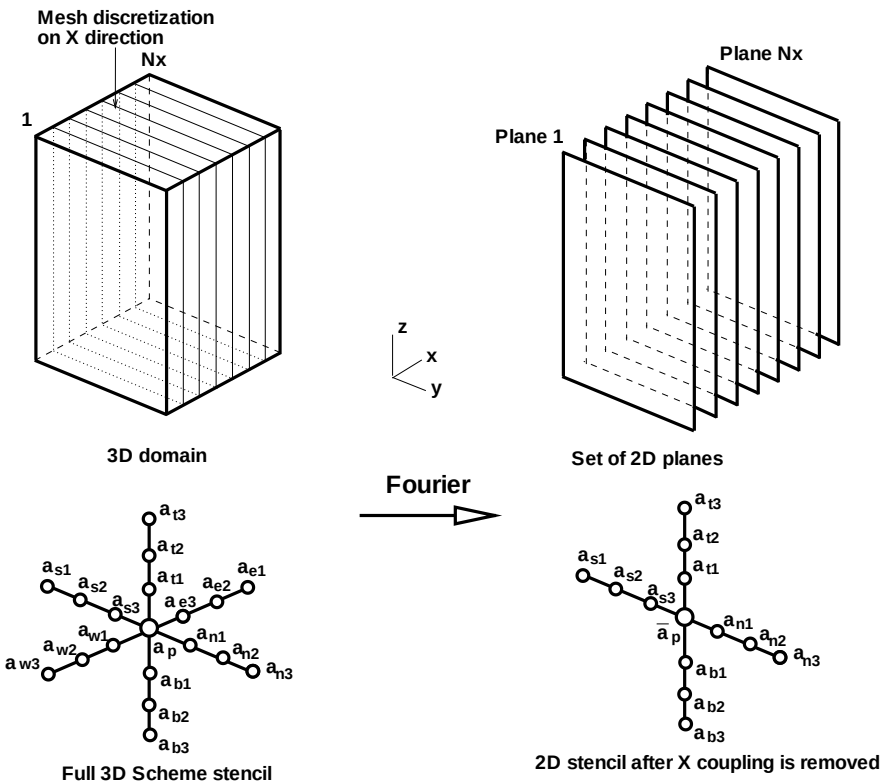
- Good ratio **CPU power / cost**
- Low bandwidth and **high latency** network



Supercomputers

- Much bigger **number of CPU** and much higher price per CPU
- High bandwidth and **low latency** network

Krylov-Schur-Fourier Decomposition: scalable algorithm

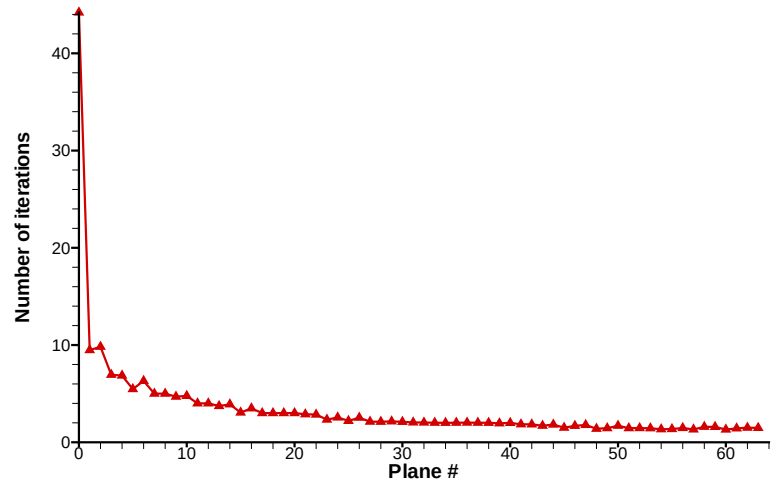


- **KSFD** is a combination of a **FFT-based** method and **Krylov** method of Conjugated Gradients preconditioned with **Direct Schur** method:
  - **Fourier** diagonalization is applied to reduce the 3D problem in a set of 2D problems.
  - The mesh must be uniform in x-direction.
  - Each 2D problem is solved with a direct **DSD** method or **CG** iterative method.
- **DSD** is a Direct Schur method
- After pre-processing, DSD solves Poisson equation to **machine accuracy** with just **ONE** reduction-type communication
- The matrix A has to be used many times with different right-hand-sides (like in case of incompressible DNS)
- Schur interface matrix grows fast with number of CPU hence method is not well scalable

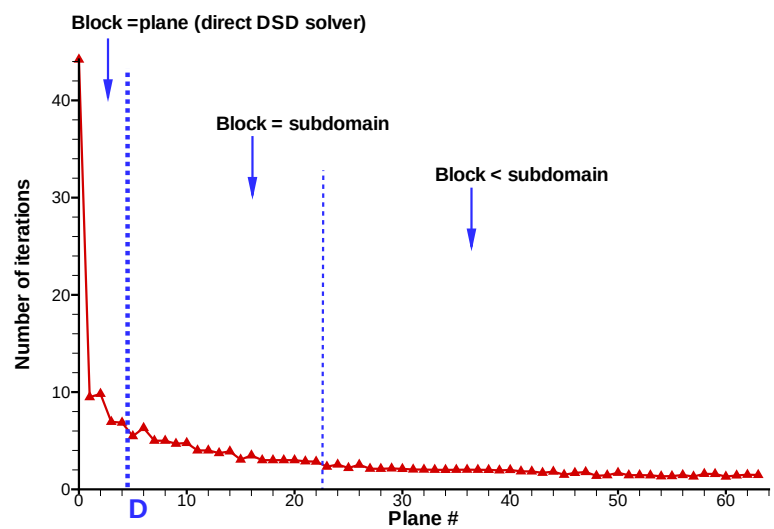
Flexible KSFD configuration

FFT provides a set of 2D independent systems with significantly different condition numbers

- Planes with number $1 \leq i \leq D$ are solved directly by **DSD** method
- Planes with number $D < i \leq N_x$ are solved using CG with local **LU** preconditioner
- **Small** D - to perform well with big meshes and big number of CPU
- **Big** D - to perform well with smaller meshes and small number of CPU

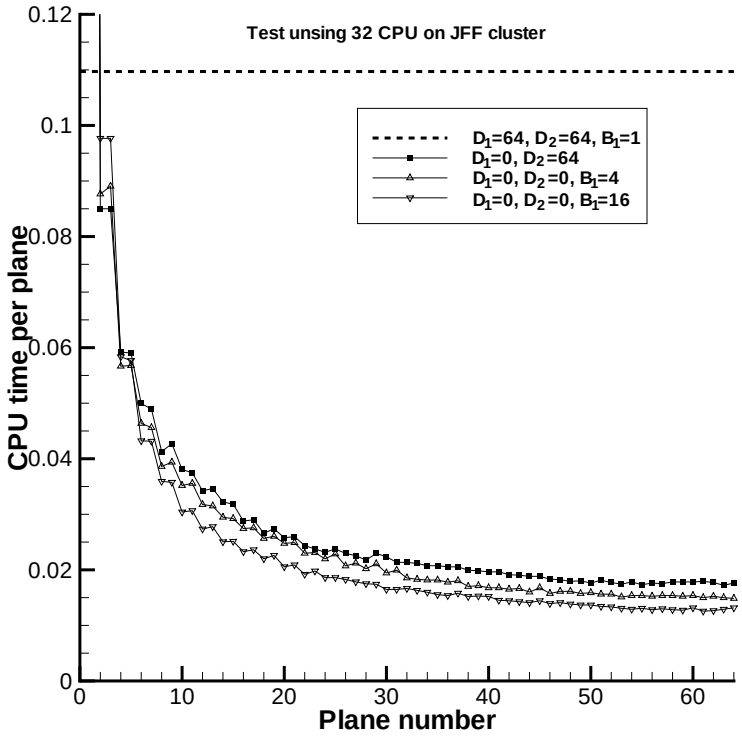


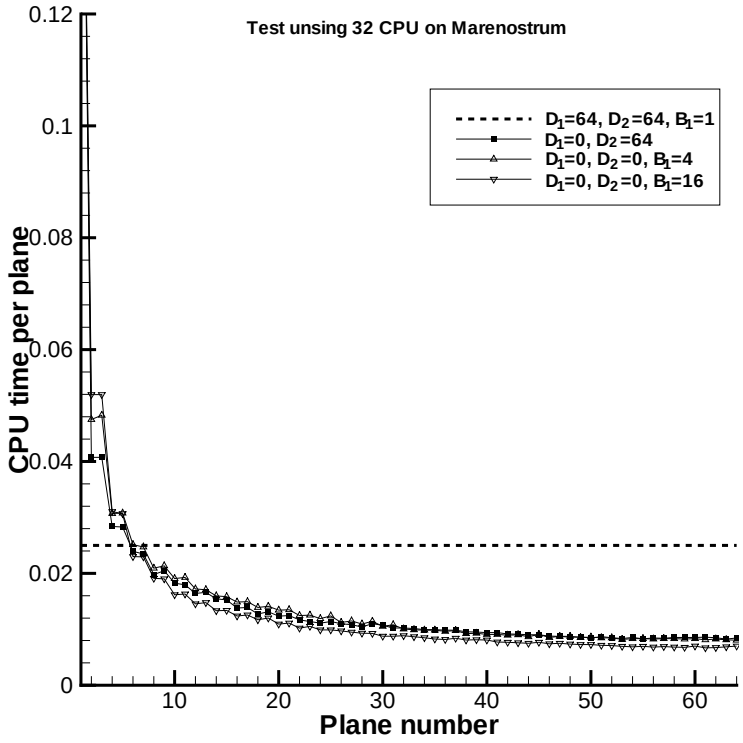
Typical distribution of the number of iterations



Example of KSFD adaptation for different parallel systems

By changing D parameter solver can be optimized for particular parallel system

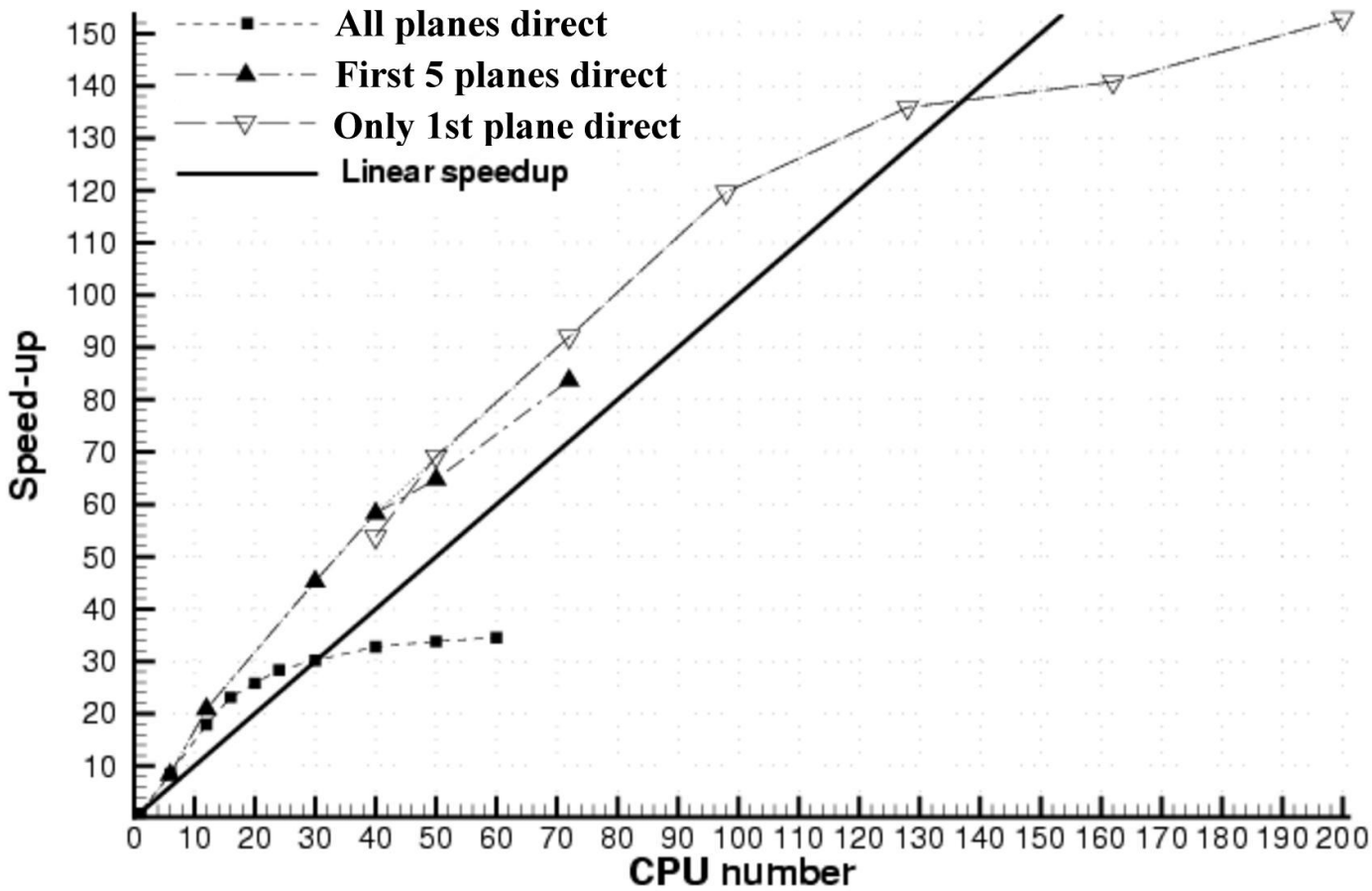




One of the several basic configurations can be chosen for each plane independently to get better performance.

KSFD adaptation for different CPU number (MareNostrum)

Mesh size 32x170x320 ($1{,}7M$ nodes), domain decomposition only on Y and Z directions



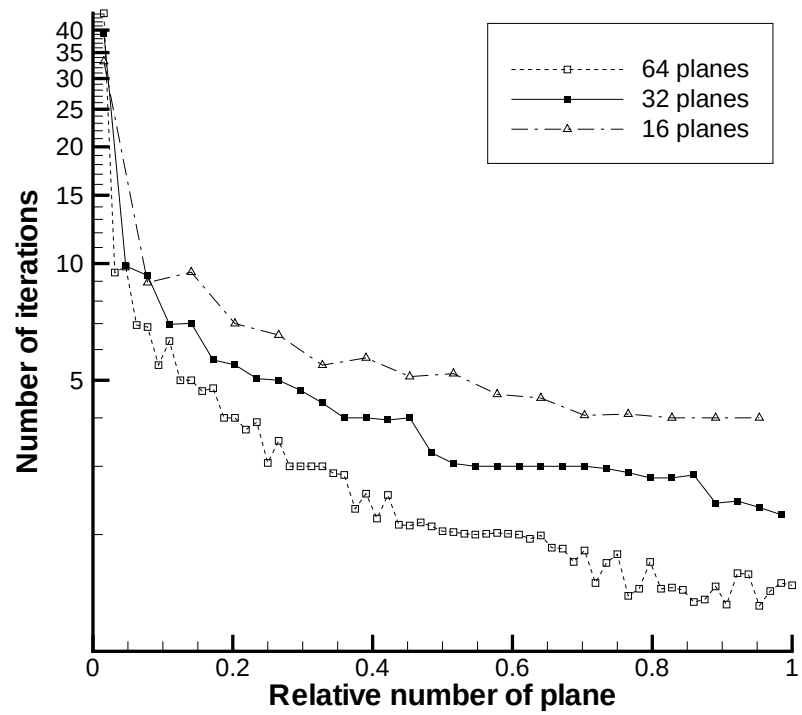
Scalability

Number of iterations **decreases** when number of planes increases

This makes algorithm **well scalable** - the number of nodes grows but the number of iterations **decreases!**.

The first plane accumulates bad convergence, but it is solved with DSD.

For example:
With mesh 16x40x80 average number of iterations is **5.75** but with 8 times bigger mesh 32x80x160 it is **5.2**

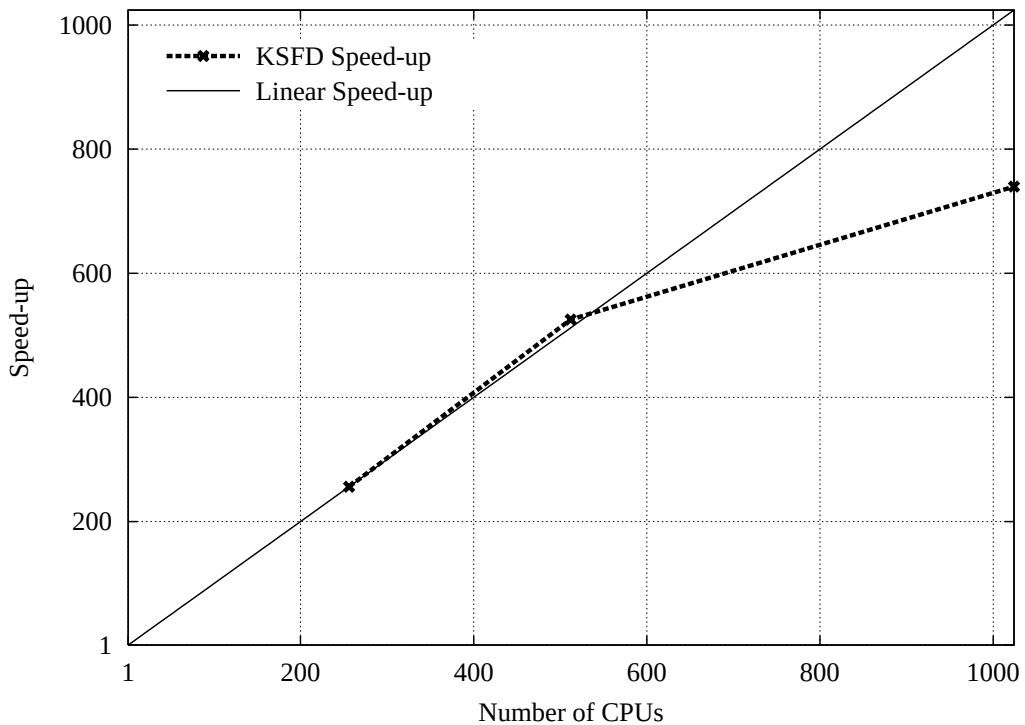


Parallelization along X direction

DSD even for 1 plane is a limitation for CPU number

Decomposition on X direction gives P_x times bigger number of CPU

- FFT data is replicated within 1D CPU groups along X direction
- Then solution of planes is done within 2D CPU groups on Y and Z directions



Mesh size 111**M** nodes (128x680x1280), size of 2D groups 128 CPU,
sizes of 1D groups are 2, 4 and 8 for 256, 512 and 1024 CPU respectively

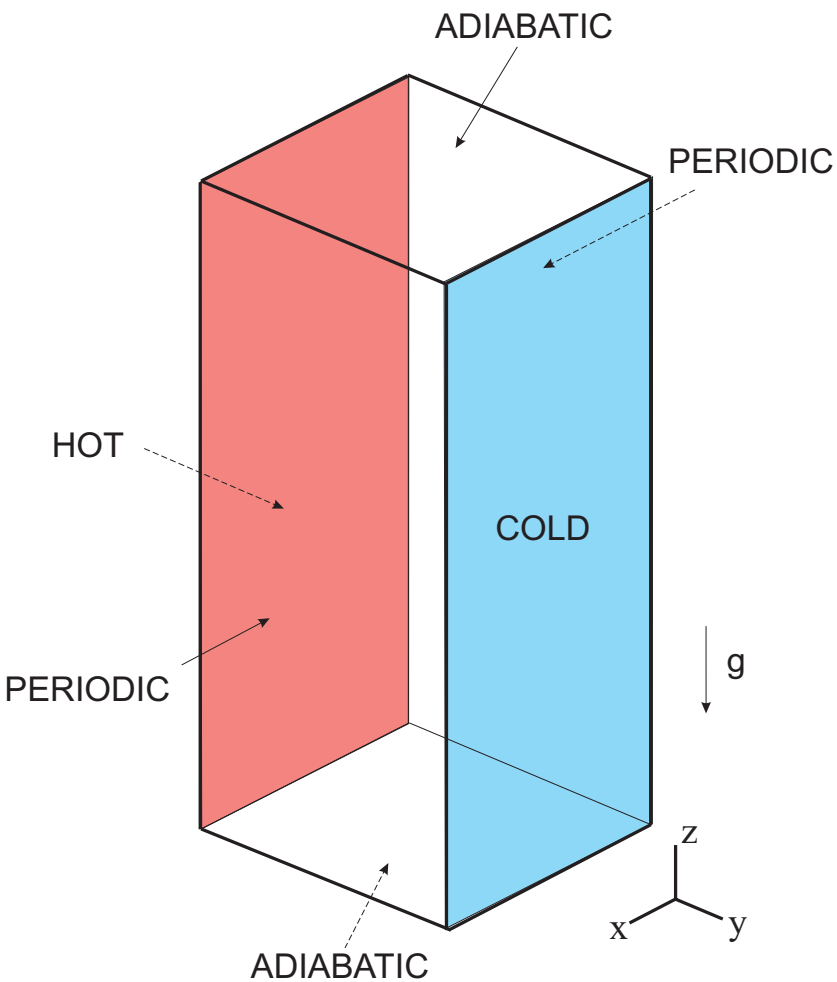
Details of implementation

Algorithm

- **FFT decomposes 3D problem into set of 2D problems**
  1 all gather communication in 1D groups on X direction
- **CG algorithm solves the set of 2D problems**
  1. Call to multi-preconditioner
     - call to multi DSD solver for planes with numbers $1, .., D$
       1 all reduce communication within 2D groups
     - call to multi local LU for planes with numbers $D + 1, .., N_x$
  2. Inner CG algebraic operations:
     - multi m.v.p. 1 halo update point-to-point communication in 2D groups
     - multi dot products all reduce communication in 2D groups
- **inverse FFT restores solution of the 3D problem**
  1 all gather communication in 1D groups on X direction

Multi - planes are grouped to have 1 communication episode for all planes

Problem definition: Differentially Heated Cavity



- **Mesh size** 128x680x1280, 111×10^8 **nodes**
- **DNS** carried out on **Marenostrum** supercomputer using 512 CPU
- **4-th order** space approximation

Boundary conditions:

- **Isothermal vertical walls**
- **Adiabatic horizontal walls**
- **Periodic** boundary conditions in the x-direction, orthogonal to the main flow

Dimensionless governing numbers:

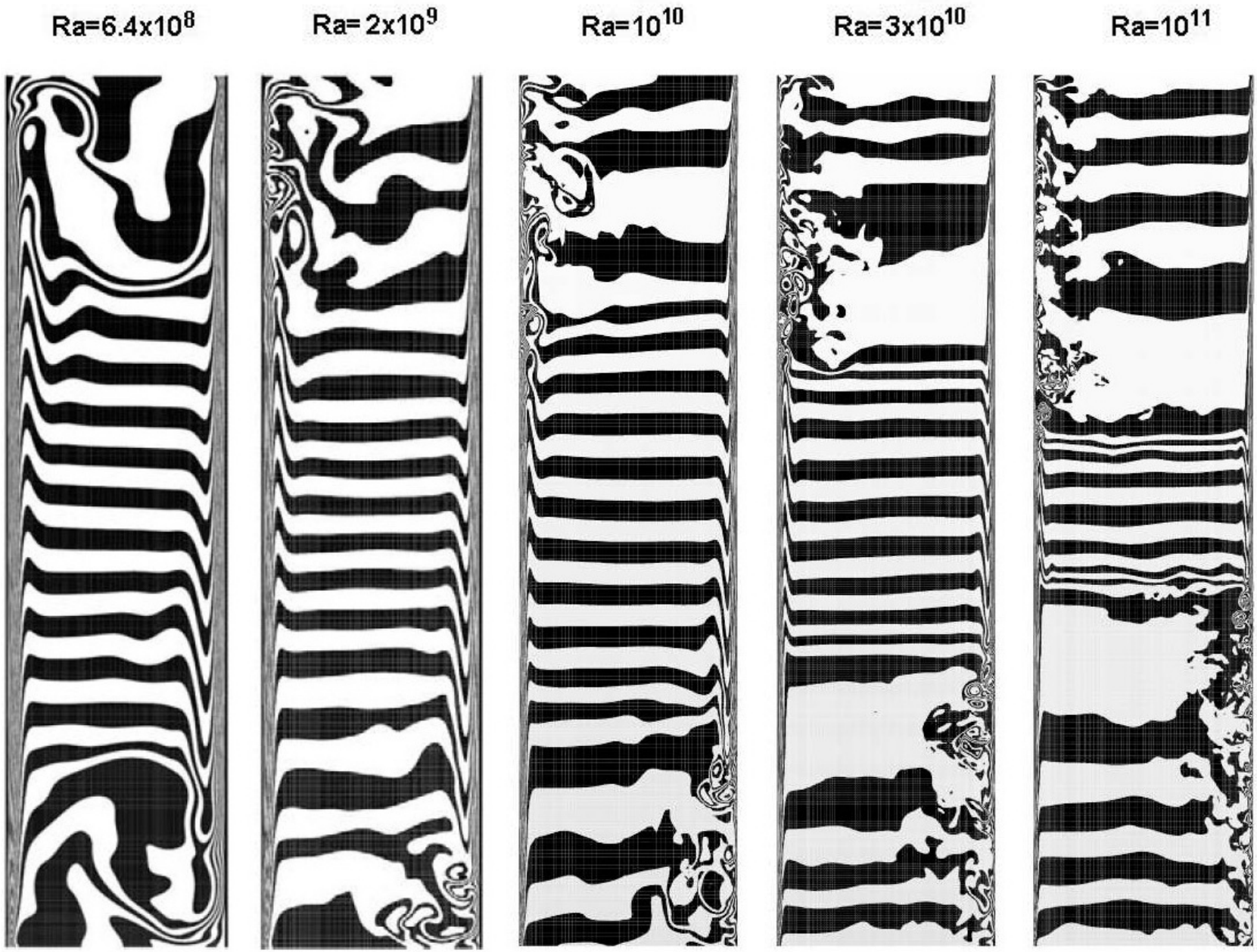
- $Ra_z = \frac{\beta \Delta T L_z^3 g}{\alpha \nu}$ =**10**^{11}
- $Pr = \frac{\nu}{\alpha}$ = **0.71** (air)
- Height aspect ratio $A_z = \frac{Lz}{Ly}$ =**4**

$$Ra = 2 \times 10^9$$

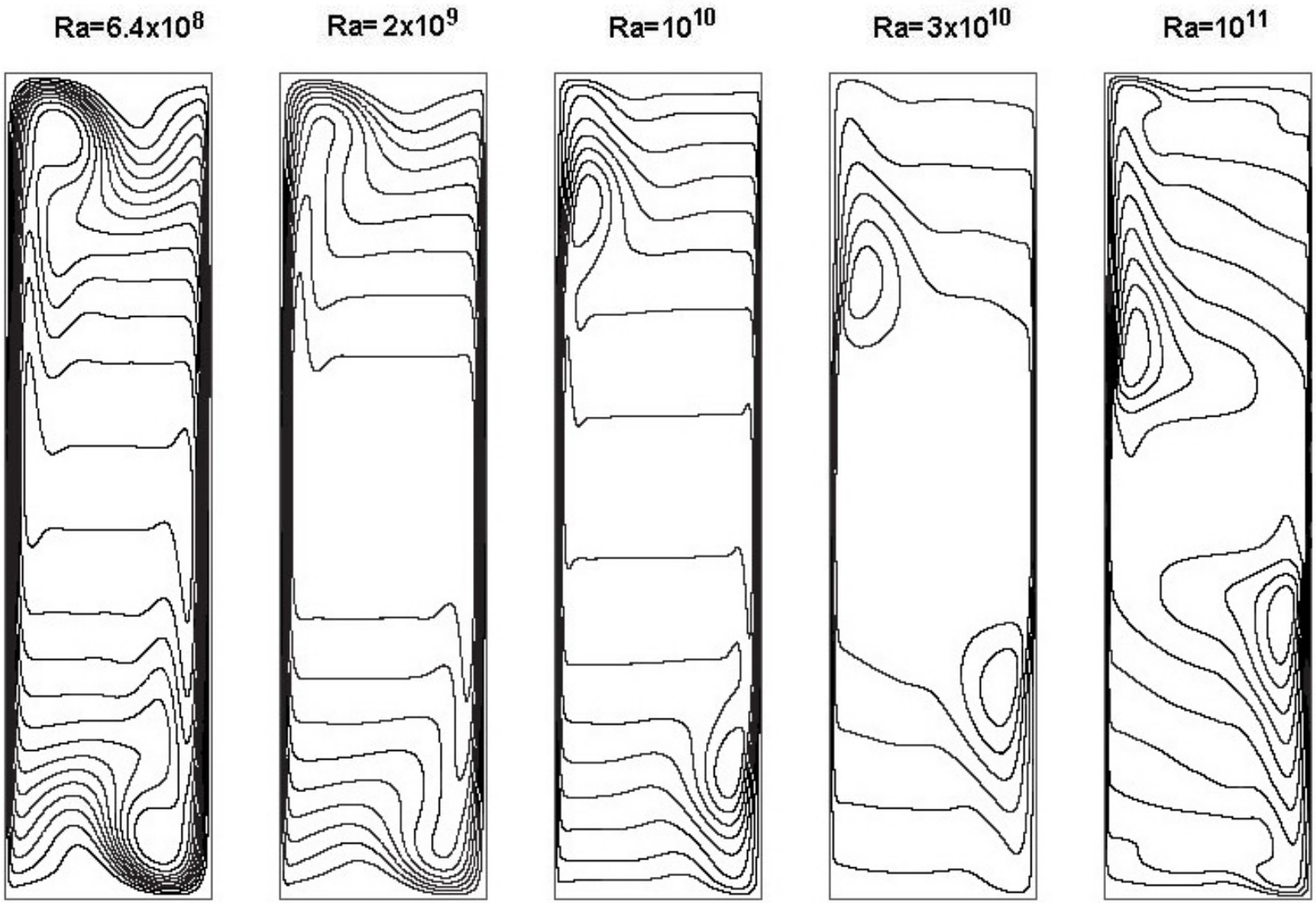
$$Ra = 10^{10}$$

$$Ra = 10^{11}$$

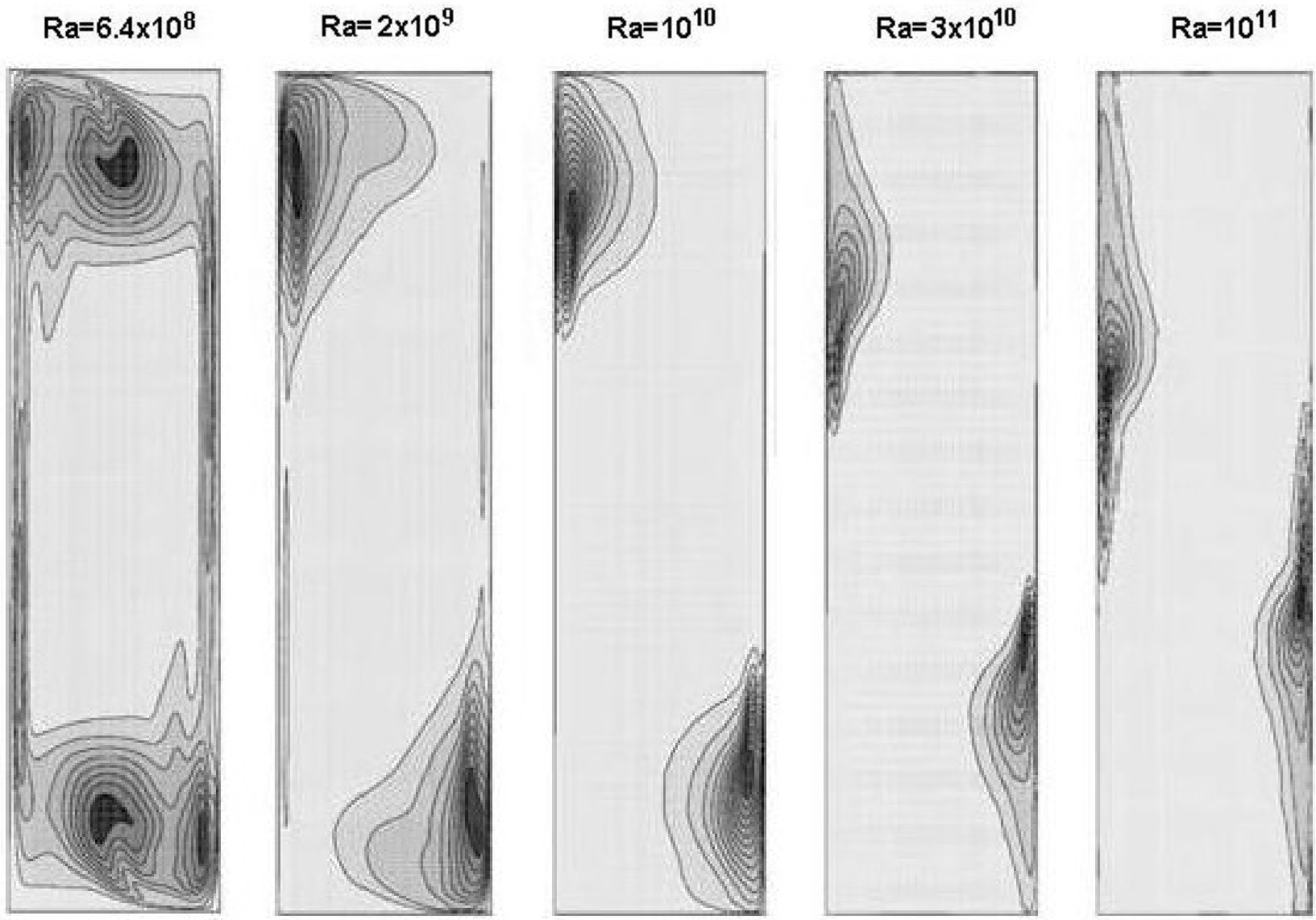
Zebra temperature snapshots



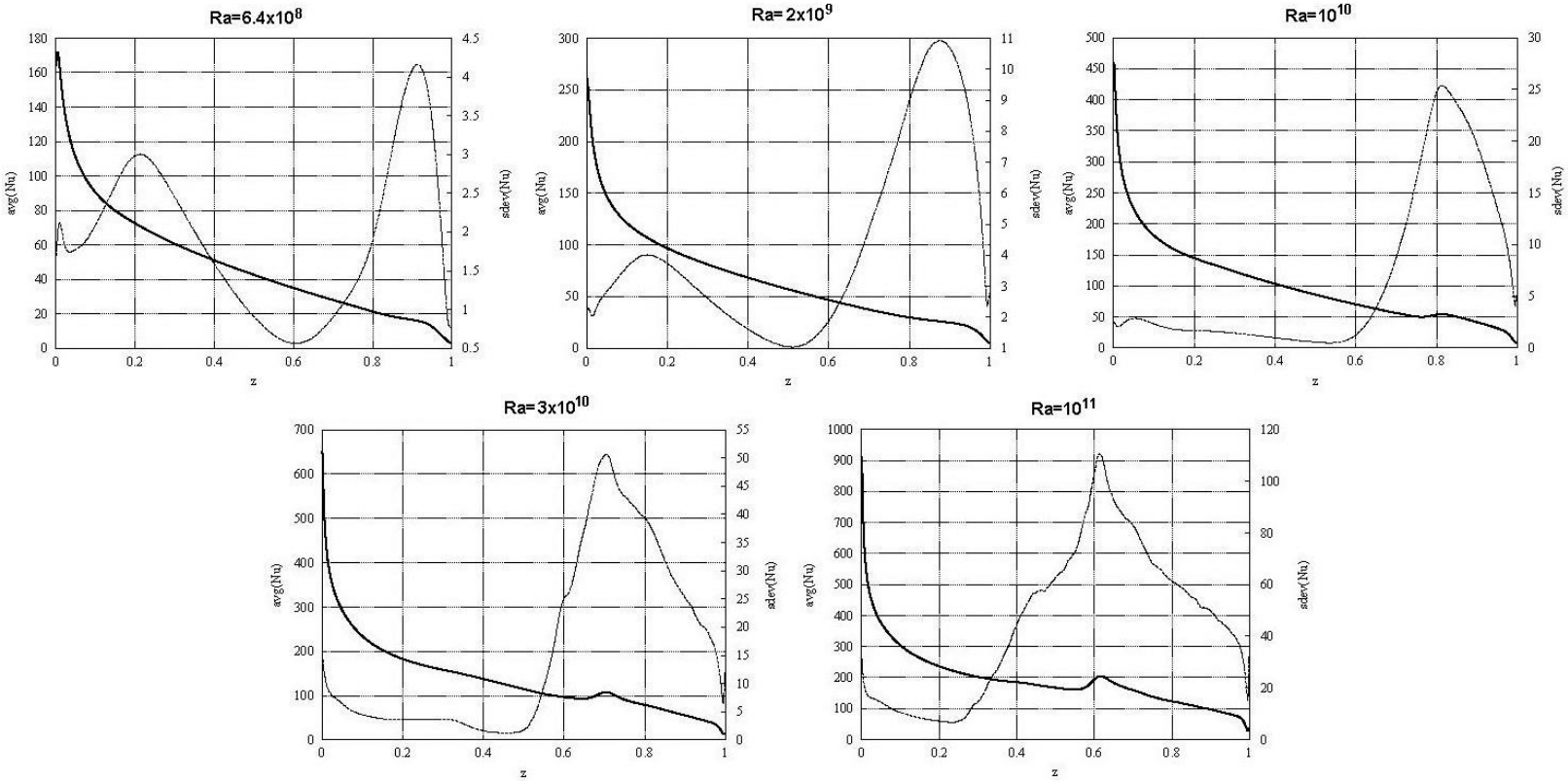
First order statistics: stream function isolines



Second order statistics: turbulent kinetic energy



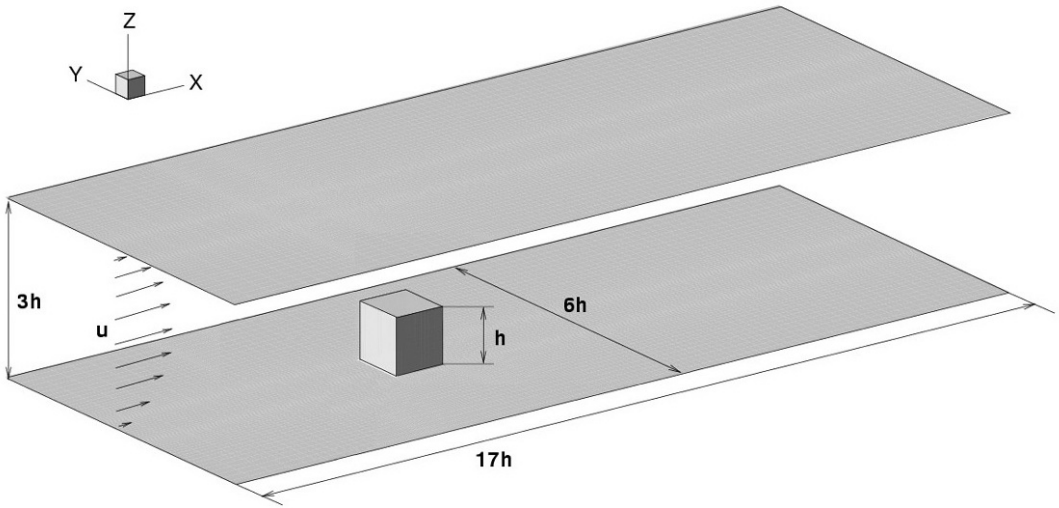
Local Nusselt number distribution

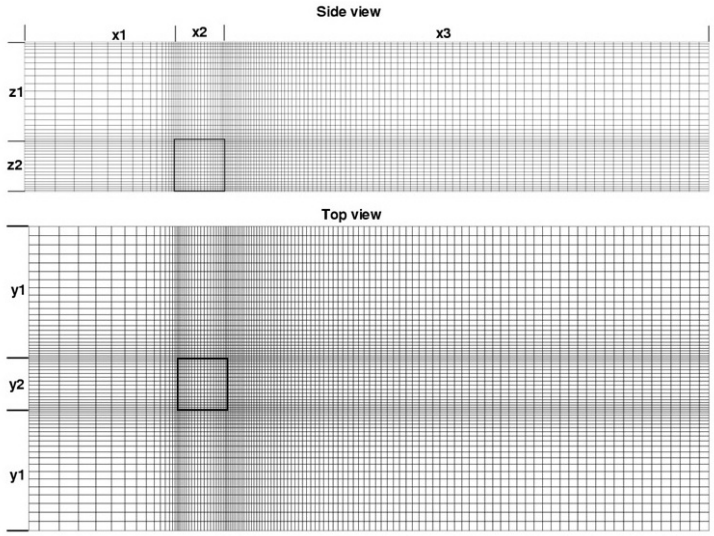


Future research: The Cube

DNS of surface mounted cube in a channel

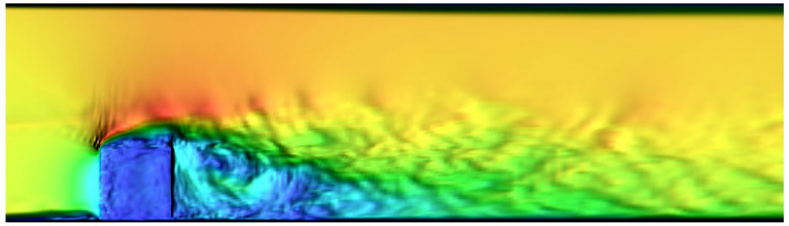
- Extended Poisson solver with multigrid overlay for fully 3D geometry
- Non uniform mesh in all 3 directions
- Estimated mesh size 30M nodes, $Re > 7000$ (based on cube height), 400-600CPU in use on MareNostrum

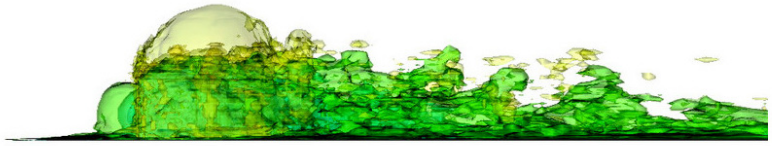


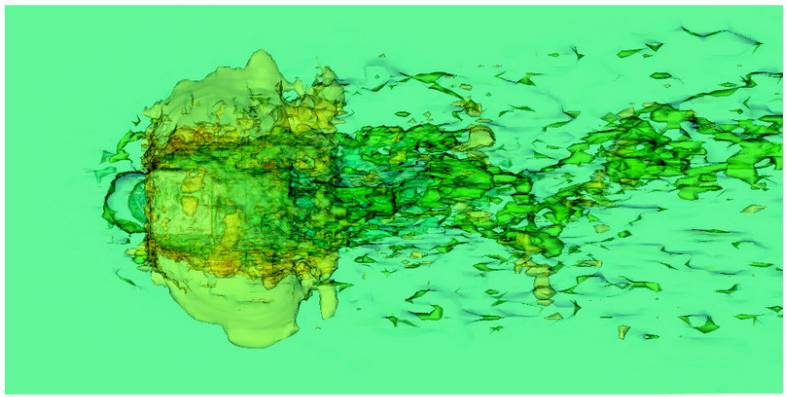


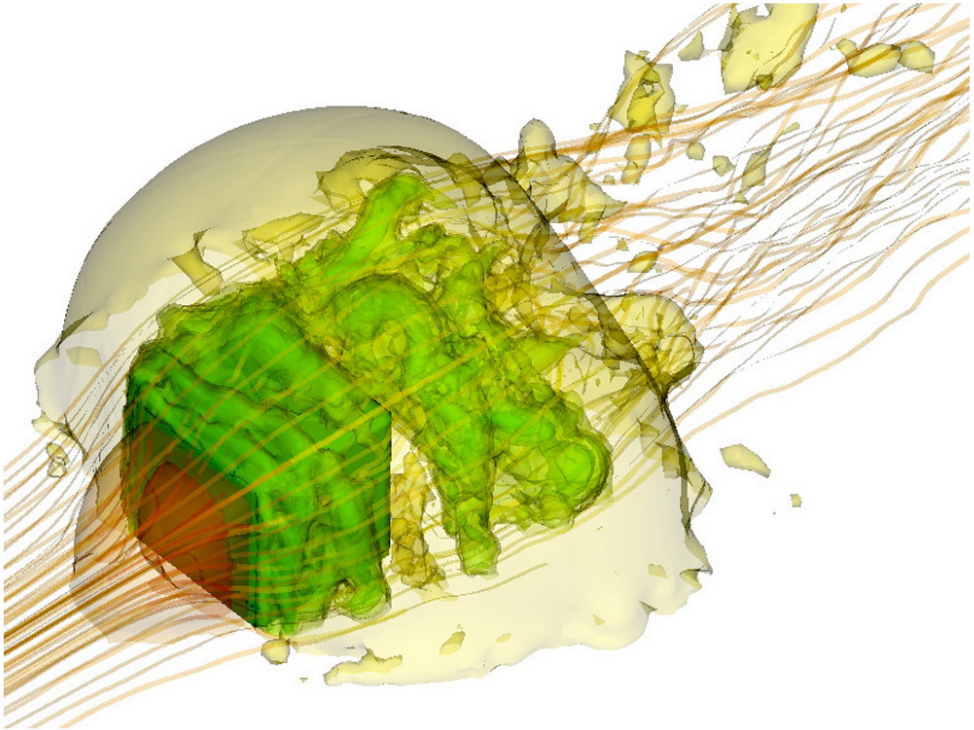
Future research: The Cube

Some snapshots from preliminary DNS









Thank you for attention