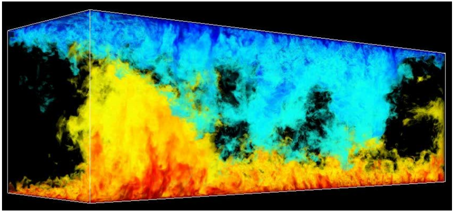
Centre Tecnològic de Transferència de Calor
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Tackling turbulence with (super)computers at CTTC

F.Xavier Trias, Joaquim Rigola, Assensi Oliva

Heat and Mass Transfer Technological Center, Technical University of Catalonia



Centre Tecnològic de Transferència de Calor
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Tackling turbulence with (super)computers at CTTC

F.Xavier Trias, Joaquim Rigola, Assensi Oliva

Heat and Mass Transfer Technological Center, Technical University of Catalonia

Centre Tecnològic de Transferència de Calor
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Tackling turbulence with (super)computers at CTTC

F.Xavier Trias, Joaquim Rigola, Assensi Oliva

Heat and Mass Transfer Technological Center, Technical University of Catalonia

Centre Tecnològic de Transferència de Calor
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Preserving operator symmetries? Physical, numerical and computational implications

F.Xavier Trias, Joaquim Rigola, Assensi Oliva

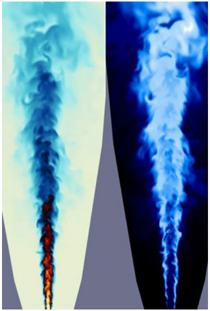
Heat and Mass Transfer Technological Center, Technical University of Catalonia

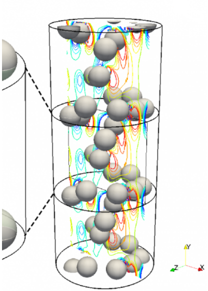
Contents

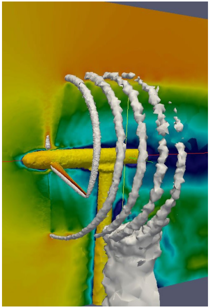
The CTTC research group

Heat and Mass Transfer Technological Center (Catalan: *Centre Tecnològic de Transferència de Calor*) has more than 20 years experience on CFD:

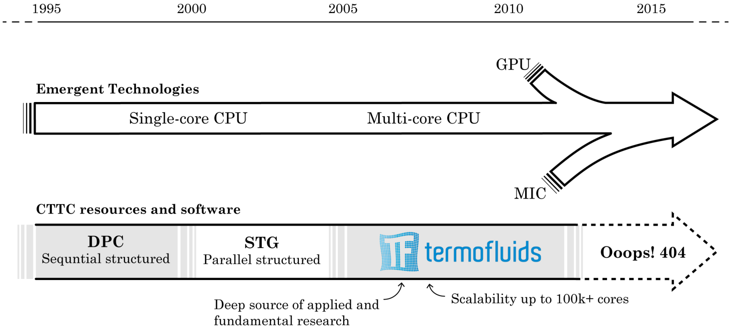
- **Fundamental research** on numerical methods, fluid dynamics and heat and mass transfer phenomena.
- **Applied research** on thermal and fluid dynamic optimization of thermal system and equipment.



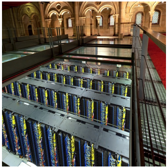




CTTC's historical background in HPC





Let's begin with some math...

$$\langle \nabla \cdot \vec{a} | \phi \rangle = -\langle \vec{a} | \nabla \phi \rangle$$

$$\langle \nabla^2 f | g \rangle = -\langle \nabla f | \nabla g \rangle = \langle f | \nabla^2 g \rangle$$

$$\langle C(\vec{u}, \phi_1) | \phi_2 \rangle = -\langle C(\vec{u}, \phi_2) | \phi_1 \rangle \quad \text{if} \quad \nabla \cdot \vec{u} = 0$$

$$\langle \nabla \times \vec{a} | \vec{b} \rangle = \langle \vec{a} | \nabla \times \vec{b} \rangle$$

Notation:

$$\langle a | b \rangle := \int_\Omega a b \, d\Omega \qquad C(\vec{u}, \phi) := (\vec{u} \cdot \nabla) \phi$$

REMEMBER: we always assume **no contribution from domain boundary,** $\partial \Omega$

Let's begin with some math...

$$\langle \nabla \cdot \vec{a} | \phi \rangle = -\langle \vec{a} | \nabla \phi \rangle$$

Proof:

$$\nabla \cdot (\phi \vec{a}) = \phi \nabla \cdot \vec{a} + \vec{a} \cdot \nabla \phi$$

$$\int_{\Omega} \nabla \cdot (\phi \vec{a}) = \langle \phi | \nabla \cdot \vec{a} \rangle + \langle \vec{a} | \nabla \phi \rangle$$

$$\int_{\partial \Omega} (\phi \vec{a}) \cdot \vec{n} \, dS = \langle \phi | \nabla \cdot \vec{a} \rangle + \langle \vec{a} | \nabla \phi \rangle = 0$$

REMAINDER!!!

$$\langle a | b \rangle := \int_{\Omega} ab \, d\Omega$$

REMEMBER: we always assume **no contribution from domain boundary,** $\partial \Omega$

Operator symmetries and conservation

$\langle \vec{u} | \vec{u} \rangle$ Kinetic energy (in 2D/3D)

$$\frac{1}{2}\frac{d\langle \vec{u}|\vec{u}\rangle}{dt} = \langle \frac{\partial \vec{u}}{\partial t}|\vec{u}\rangle = -\langle C(\vec{u},\vec{u})|\vec{u}\rangle + \nu\langle \nabla^2 \vec{u}|\vec{u}\rangle - \langle \nabla p|\vec{u}\rangle$$

$$= -\nu\langle \nabla \vec{u}|\nabla \vec{u}\rangle = -\nu\|\nabla \vec{u}\|^2 \leq 0$$

$$= -\nu\langle \nabla\times\nabla\times\vec{u}|\vec{u}\rangle = -\nu\|\omega\|^2 \leq 0$$

If $\nu=0$, then $\langle \vec{u}|\vec{u}\rangle$ remains constant!!!

Also, if the flow is irrotational, $\vec{\omega}=\vec{0}$. Remember Bernoulli!

ADDITIONAL REMAINDER!!!

$$\nabla^2\vec{u} = \nabla(\nabla\cdot\vec{u}) - \nabla\times\nabla\times\vec{u}$$

REMAINDER!!!

$$\langle \nabla\cdot\vec{a}|\phi\rangle = -\langle \vec{a}|\nabla\phi\rangle$$

$$\langle \nabla^2 f|g\rangle = -\langle \nabla f|\nabla g\rangle = \langle f|\nabla^2 g\rangle$$

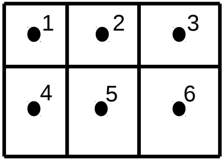
$$\langle C(\vec{u},\phi_1)|\phi_2\rangle = -\langle C(\vec{u},\phi_2)|\phi_1\rangle \quad \text{if } \nabla\cdot\vec{u}=0$$

$$\langle \nabla\times\vec{a}|\vec{b}\rangle = \langle \vec{a}|\nabla\times\vec{b}\rangle$$

From Calculus to Algebra (C2A)

$$\langle a|b\rangle := \int_\Omega ab\, d\Omega \in \mathbb{R}$$

$$\langle a_h|b_h\rangle := a_h^T \mathbf{\Omega}\, b_h \in \mathbb{R}$$

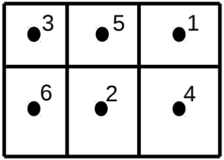


$$a_h = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{pmatrix} \qquad \mathbf{\Omega} = \begin{pmatrix} \Omega_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \Omega_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \Omega_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & \Omega_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & \Omega_5 & 0 \\ 0 & 0 & 0 & 0 & 0 & \Omega_6 \end{pmatrix} \qquad b_h = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \end{pmatrix}$$

From Calculus to Algebra (C2A)

$$\langle a|b\rangle := \int_{\Omega} ab \, d\Omega \quad \in \mathbb{R}$$

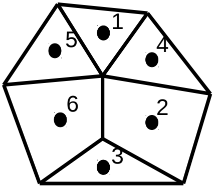
$$\langle a_h|b_h\rangle := a_h^T \mathbf{\Omega} \, b_h \quad \in \mathbb{R}$$



$$a_h = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{pmatrix} \qquad \mathbf{\Omega} = \begin{pmatrix} \Omega_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \Omega_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \Omega_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & \Omega_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & \Omega_5 & 0 \\ 0 & 0 & 0 & 0 & 0 & \Omega_6 \end{pmatrix} \qquad b_h = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \end{pmatrix}$$

From Calculus to Algebra (C2A)

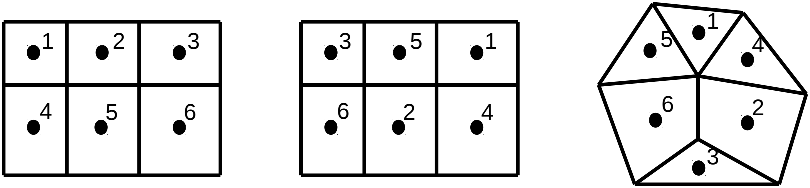
$$\langle a|b\rangle := \int_{\Omega} ab\, d\Omega \quad \in \mathbb{R}$$



$$\langle a_h|b_h\rangle := a_h^T \boldsymbol{\Omega}\, b_h \quad \in \mathbb{R}$$

$$a_h = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{pmatrix} \qquad \boldsymbol{\Omega} = \begin{pmatrix} \Omega_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \Omega_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \Omega_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & \Omega_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & \Omega_5 & 0 \\ 0 & 0 & 0 & 0 & 0 & \Omega_6 \end{pmatrix} \qquad b_h = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \end{pmatrix}$$

From Calculus to Algebra (C2A)



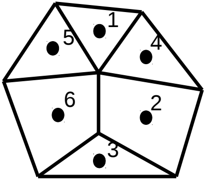
$$T=\begin{pmatrix} \times & \times & 0 & \times & 0 & 0 \\ \times & \times & \times & 0 & \times & 0 \\ 0 & \times & \times & 0 & 0 & \times \\ \times & 0 & 0 & \times & \times & 0 \\ 0 & \times & 0 & \times & \times & \times \\ 0 & 0 & \times & 0 & \times & \times \end{pmatrix} \quad T=\begin{pmatrix} \times & 0 & 0 & \times & \times & 0 \\ 0 & \times & 0 & \times & \times & \times \\ 0 & 0 & \times & 0 & \times & \times \\ \times & \times & 0 & \times & 0 & 0 \\ \times & \times & \times & 0 & \times & 0 \\ 0 & \times & \times & 0 & 0 & \times \end{pmatrix} \quad T=\begin{pmatrix} \times & 0 & 0 & \times & \times & 0 \\ 0 & \times & \times & \times & 0 & \times \\ 0 & \times & \times & 0 & 0 & \times \\ \times & \times & 0 & \times & 0 & 0 \\ \times & 0 & 0 & 0 & \times & \times \\ 0 & \times & \times & 0 & \times & \times \end{pmatrix}$$

From Calculus to Algebra (C2A)

$$\frac{\partial \vec{u}}{\partial t} + (\vec{u} \cdot \nabla)\vec{u} = \nu \nabla^2 \vec{u} - \nabla p \qquad \nabla \cdot \vec{u} = 0$$

$$\boldsymbol{\Omega} \frac{d u_h}{d t} + \boldsymbol{C}(u_h) u_h = \boldsymbol{D} u_h - \boldsymbol{\Omega} \boldsymbol{G} p_h \qquad \boldsymbol{M} u_h = 0_h$$

$$p_h(t) = \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{pmatrix} \qquad u_h(t) = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{pmatrix}$$



$$\boldsymbol{\Omega} = \begin{pmatrix} \boldsymbol{\Omega}_u & \\ & \boldsymbol{\Omega}_v \end{pmatrix}$$

$$\boldsymbol{T} = \begin{pmatrix} \times & 0 & 0 & \times & \times & 0 \\ 0 & \times & \times & \times & 0 & \times \\ 0 & \times & \times & 0 & 0 & \times \\ \times & \times & 0 & \times & 0 & 0 \\ \times & 0 & 0 & 0 & \times & \times \\ 0 & \times & \times & 0 & \times & \times \end{pmatrix}$$

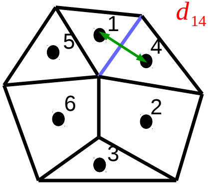
$$\boldsymbol{\Omega}_u = \boldsymbol{\Omega}_v = \begin{pmatrix} \Omega_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \Omega_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \Omega_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & \Omega_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & \Omega_5 & 0 \\ 0 & 0 & 0 & 0 & 0 & \Omega_6 \end{pmatrix}$$

From Calculus to Algebra (C2A)

$$\frac{\partial \vec{u}}{\partial t} + (\vec{u}\cdot\nabla)\vec{u} = \nu \nabla^2 \vec{u} - \nabla p \qquad \nabla\cdot\vec{u} = 0$$

$$\boldsymbol{\Omega}\frac{d u_h}{dt} + \boldsymbol{C}(u_h)u_h = \boldsymbol{D} u_h - \boldsymbol{\Omega}\boldsymbol{G} p_h \qquad \boldsymbol{M} u_h = 0_h$$

$$p_h(t) = \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{pmatrix} \qquad u_h(t) = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{pmatrix}$$



$$d_{14} = \nu A_{14}/\delta_{14}$$

$$\boldsymbol{D} = \begin{pmatrix} \boldsymbol{D}_u & \\ & \boldsymbol{D}_v \end{pmatrix}$$

$$\boldsymbol{T} = \begin{pmatrix} \times & 0 & 0 & \times & \times & 0 \\ 0 & \times & \times & \times & 0 & \times \\ 0 & \times & \times & 0 & 0 & \times \\ \times & \times & 0 & \times & 0 & 0 \\ \times & 0 & 0 & 0 & \times & \times \\ 0 & \times & \times & 0 & \times & \times \end{pmatrix}$$

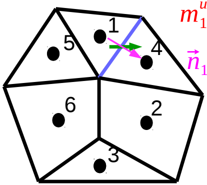
$$\boldsymbol{D}_u = \boldsymbol{D}_v = \begin{pmatrix} d_{11} & 0 & 0 & d_{14} & d_{15} & 0 \\ 0 & d_{22} & d_{23} & d_{24} & 0 & d_{26} \\ 0 & d_{23} & d_{33} & 0 & 0 & d_{36} \\ d_{14} & d_{24} & 0 & d_{44} & 0 & 0 \\ d_{15} & 0 & 0 & 0 & d_{55} & d_{56} \\ 0 & d_{26} & d_{36} & 0 & d_{56} & d_{66} \end{pmatrix}$$

From Calculus to Algebra (C2A)

$$\frac{\partial \vec{u}}{\partial t} + (\vec{u}\cdot\nabla)\vec{u} = \nu \nabla^2 \vec{u} - \nabla p \qquad \nabla\cdot\vec{u} = 0$$

$$\boldsymbol{\Omega}\frac{d u_h}{d t} + \boldsymbol{C}(u_h)u_h = \boldsymbol{D} u_h - \boldsymbol{\Omega}\boldsymbol{G} p_h \qquad \boldsymbol{M} u_h = 0_h \qquad p_h(t) = \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{pmatrix}$$

$$u_h(t) = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{pmatrix}$$



$$m^u_{14} = A_{14} n^x_{14}/2$$

$$\vec{n}_{14} = \begin{pmatrix} n^x_{14} & n^y_{14} \end{pmatrix}$$

$$\boldsymbol{M} = \begin{pmatrix} \boldsymbol{M}^u & \boldsymbol{M}^v \end{pmatrix}$$

$$\boldsymbol{T} = \begin{pmatrix} \times & 0 & 0 & \times & \times & 0 \\ 0 & \times & \times & \times & 0 & \times \\ 0 & \times & \times & 0 & 0 & \times \\ \times & \times & 0 & \times & 0 & 0 \\ \times & 0 & 0 & 0 & \times & \times \\ 0 & \times & \times & 0 & \times & \times \end{pmatrix}$$

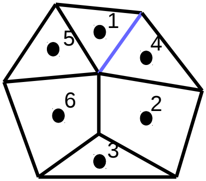
$$\boldsymbol{M}^u = \begin{pmatrix} 0 & 0 & 0 & m^u_{14} & m^u_{15} & 0 \\ 0 & 0 & m^u_{23} & m^u_{24} & 0 & m^u_{26} \\ 0 & -m^u_{23} & 0 & 0 & 0 & m^u_{36} \\ -m^u_{14} & -m^u_{24} & 0 & 0 & 0 & 0 \\ -m^u_{15} & 0 & 0 & 0 & 0 & m^u_{56} \\ 0 & -m^u_{26} & -m^u_{36} & 0 & -m^u_{56} & 0 \end{pmatrix}$$

From Calculus to Algebra (C2A)

$$\frac{\partial \vec{u}}{\partial t} + (\vec{u}\cdot\nabla)\vec{u} = \nu\nabla^2\vec{u} - \nabla p \qquad \nabla\cdot\vec{u} = 0$$

$$\boldsymbol{\Omega}\frac{d u_h}{dt} + \boldsymbol{C}(u_h)u_h = \boldsymbol{D}u_h - \boldsymbol{\Omega}\boldsymbol{G}p_h \qquad \boldsymbol{M}u_h = 0_h$$

$$p_h(t) = \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{pmatrix} \qquad u_h(t) = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{pmatrix}$$



$$\boldsymbol{G} = \begin{pmatrix} \boldsymbol{G}^x \\ \boldsymbol{G}^y \end{pmatrix}$$

$$\boldsymbol{T} = \begin{pmatrix} \times & 0 & 0 & \times & \times & 0 \\ 0 & \times & \times & \times & 0 & \times \\ 0 & \times & \times & 0 & 0 & \times \\ \times & \times & 0 & \times & 0 & 0 \\ \times & 0 & 0 & 0 & \times & \times \\ 0 & \times & \times & 0 & \times & \times \end{pmatrix}$$

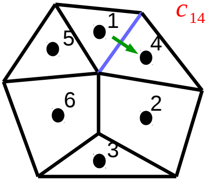
$$\boldsymbol{G}^x = \begin{pmatrix} 0 & 0 & 0 & g^x_{14} & g^x_{15} & 0 \\ 0 & 0 & g^x_{23} & g^x_{24} & 0 & g^x_{26} \\ 0 & -g^x_{23} & 0 & 0 & 0 & g^x_{36} \\ -g^x_{14} & -g^x_{24} & 0 & 0 & 0 & 0 \\ -g^x_{15} & 0 & 0 & 0 & 0 & g^x_{56} \\ 0 & -g^x_{26} & -g^x_{36} & 0 & -g^x_{56} & 0 \end{pmatrix}$$

From Calculus to Algebra (C2A)

$$\frac{\partial \vec{u}}{\partial t} + (\vec{u}\cdot\nabla)\vec{u} = \nu \nabla^2 \vec{u} - \nabla p \qquad \nabla \cdot \vec{u} = 0$$

$$\boldsymbol{\Omega}\frac{d u_h}{d t} + \mathbf{C}(u_h)u_h = \mathbf{D}u_h - \boldsymbol{\Omega}\mathbf{G}p_h \qquad \mathbf{M}u_h = 0_h$$

$$p_h(t) = \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{pmatrix} \qquad u_h(t) = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{pmatrix}$$



$$c_{14} = A_{14} U_{14}$$

$$\mathbf{C} = \begin{pmatrix} \mathbf{C}_u & \\ & \mathbf{C}_v \end{pmatrix}$$

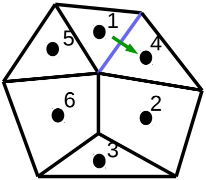
$$\mathbf{T} = \begin{pmatrix} \times & 0 & 0 & \times & \times & 0 \\ 0 & \times & \times & \times & 0 & \times \\ 0 & \times & \times & 0 & 0 & \times \\ \times & \times & 0 & \times & 0 & 0 \\ \times & 0 & 0 & 0 & \times & \times \\ 0 & \times & \times & 0 & \times & \times \end{pmatrix}$$

$$\mathbf{C}_u = \mathbf{C}_v = \begin{pmatrix} c_{11} & 0 & 0 & c_{14} & c_{15} & 0 \\ 0 & c_{22} & c_{23} & c_{24} & 0 & c_{26} \\ 0 & c_{32} & c_{33} & 0 & 0 & c_{36} \\ c_{41} & c_{42} & 0 & c_{44} & 0 & 0 \\ c_{51} & 0 & 0 & 0 & c_{55} & c_{56} \\ 0 & c_{62} & c_{63} & 0 & c_{65} & c_{66} \end{pmatrix}$$

Algebraic operators

$$\frac{\partial \vec{u}}{\partial t} + (\vec{u} \cdot \nabla)\vec{u} = \nu \nabla^2 \vec{u} - \nabla p \qquad \nabla \cdot \vec{u} = 0$$

$$\boldsymbol{\Omega} \frac{d u_h}{d t} + \boldsymbol{C}(u_h) u_h = \boldsymbol{D} u_h - \boldsymbol{\Omega} \boldsymbol{G} p_h \qquad \boldsymbol{M} u_h = 0_h$$



$$\boldsymbol{\Omega} = \begin{pmatrix} \boldsymbol{\Omega}_u & \\ & \boldsymbol{\Omega}_v \end{pmatrix}$$

$$\boldsymbol{M} = \begin{pmatrix} \boldsymbol{M}^u & \boldsymbol{M}^v \end{pmatrix}$$

$$\boldsymbol{G} = \begin{pmatrix} \boldsymbol{G}^x \\ \boldsymbol{G}^y \end{pmatrix}$$

$$\boldsymbol{T} = \begin{pmatrix} \times & 0 & 0 & \times & \times & 0 \\ 0 & \times & \times & \times & 0 & \times \\ 0 & \times & \times & 0 & 0 & \times \\ \times & \times & 0 & \times & 0 & 0 \\ \times & 0 & 0 & 0 & \times & \times \\ 0 & \times & \times & 0 & \times & \times \end{pmatrix}$$

$$\boldsymbol{C} = \begin{pmatrix} \boldsymbol{C}_u & \\ & \boldsymbol{C}_v \end{pmatrix}$$

$$\boldsymbol{D} = \begin{pmatrix} \boldsymbol{D}_u & \\ & \boldsymbol{D}_v \end{pmatrix}$$

Algebraic operators: basic properties

Let us consider square matrices, $\boldsymbol{A} \in \mathbb{R}^{n \times n}$:

- Eigenvalues&eigenvectors: $\boldsymbol{A}\vec{v}_i = \lambda_i \vec{v}_i, \quad i = 1, \ldots, n$

  ...or equivalently $\left(\boldsymbol{A} - \lambda \boldsymbol{I}\right)\vec{v} = \vec{0}$

  $|\boldsymbol{A} - \lambda \boldsymbol{I}| = 0$ characteristic equation of $\boldsymbol{A}$

- $$\boldsymbol{A} = \underbrace{\frac{1}{2}\left(\boldsymbol{A} + \boldsymbol{A}^T\right)}_{symmetric} + \underbrace{\frac{1}{2}\left(\boldsymbol{A} - \boldsymbol{A}^T\right)}_{skew-symmetric}$$

Algebraic operators: basic properties

Symmetric matrices, $\boldsymbol{A} = \boldsymbol{A}^T$:

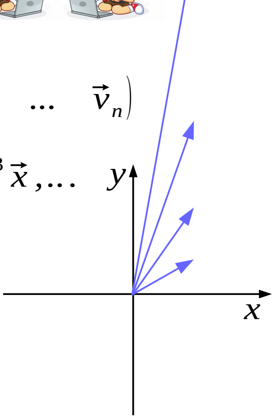
$$\boldsymbol{A}\vec{v}_i = \lambda_i \vec{v}_i, \quad \lambda_i \in \mathbb{R} \quad \vec{v}_i \in \mathbb{R}^n$$

$$\boldsymbol{\Lambda} = \boldsymbol{P}^{-1}\boldsymbol{A}\boldsymbol{P} \quad \text{where} \quad \boldsymbol{P} = \begin{pmatrix} \vec{v}_1 & \vec{v}_2 & \ldots & \vec{v}_n \end{pmatrix}$$

Example:

$$\vec{x}, \boldsymbol{\Lambda}\vec{x}, \boldsymbol{\Lambda}^2\vec{x}, \boldsymbol{\Lambda}^3\vec{x}, \ldots$$

$$\boldsymbol{\Lambda} = \begin{pmatrix} 1 & 0 \\ 0 & 1/2 \end{pmatrix} \quad \vec{x} = \begin{pmatrix} 3 \\ 16 \end{pmatrix}$$



It resembles a diffusive process!

Algebraic operators: basic properties

Skew-symmetric matrices, $\boldsymbol{A} = -\boldsymbol{A}^T$:

$$\boldsymbol{A}\vec{v}_i = \lambda_i \vec{v}_i, \quad \lambda_i \in I \quad \vec{v}_i \in I^n$$

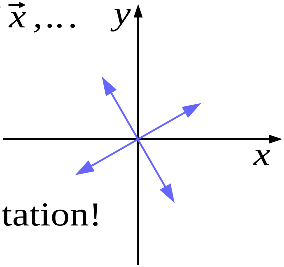
Example:

$$\vec{x}, \boldsymbol{A}\vec{x}, \boldsymbol{A}^2\vec{x}, \boldsymbol{A}^3\vec{x}, \dots$$

$$\boldsymbol{A} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \quad \vec{x} = \begin{pmatrix} 3 \\ 2 \end{pmatrix}$$

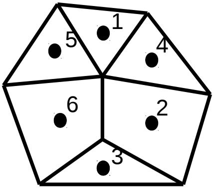
$$\boldsymbol{R} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

It is a 90º rotation!



Algebraic operators: basic properties

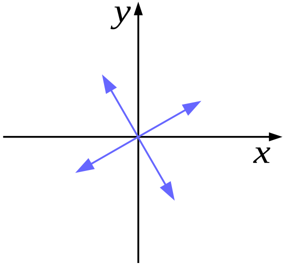
Skew-symmetric matrices, $\boldsymbol{A} = -\boldsymbol{A}^T$:



$$\boldsymbol{A}\vec{v}_i = \lambda_i \vec{v}_i, \quad \lambda_i \in I \quad \vec{v}_i \in I^n$$

And it is always a 90º rotation!!!

$$\vec{x}^T \boldsymbol{A} \vec{x} = 0, \quad \forall \vec{x} \in \mathbb{R}^n$$



Reminder: symmetry and conservation of kinetic energy

$\langle \vec{u} | \vec{u} \rangle$ Kinetic energy (in 2D/3D)

$$\frac{1}{2}\frac{d\langle \vec{u}|\vec{u}\rangle}{dt} = \langle \frac{\partial \vec{u}}{\partial t} | \vec{u} \rangle = -\langle C(\vec{u},\vec{u})|\vec{u}\rangle + \nu \langle \nabla^2 \vec{u} | \vec{u} \rangle - \langle \nabla p | \vec{u} \rangle$$

$$= -\nu \langle \nabla \vec{u} | \nabla \vec{u} \rangle = -\nu ||\nabla \vec{u}||^2 \leq 0$$

$$= -\nu \langle \nabla \times \nabla \times \vec{u} | \vec{u} \rangle = -\nu ||\omega||^2 \leq 0$$

If $\nu = 0$, then $\langle \vec{u} | \vec{u} \rangle$ remains constant!!!

Also, if the flow is irrotational, $\vec{\omega} = \vec{0}$. Remember Bernoulli!

ADDITIONAL REMAINDER!!!

$$\nabla^2 \vec{u} = \nabla(\nabla \cdot \vec{u}) - \nabla \times \nabla \times \vec{u}$$

REMAINDER!!!

$$\langle \nabla \cdot \vec{a} | \phi \rangle = -\langle \vec{a} | \nabla \phi \rangle$$

$$\langle \nabla^2 f | g \rangle = -\langle \nabla f | \nabla g \rangle = \langle f | \nabla^2 g \rangle$$

$$\langle C(\vec{u}, \phi_1) | \phi_2 \rangle = -\langle C(\vec{u}, \phi_2) | \phi_1 \rangle \quad \text{if } \nabla \cdot \vec{u} = 0$$

$$\langle \nabla \times \vec{a} | \vec{b} \rangle = \langle \vec{a} | \nabla \times \vec{b} \rangle$$

Algebraic operators

$$\frac{\partial \vec{u}}{\partial t} + (\vec{u} \cdot \nabla)\vec{u} = \nu \nabla^2 \vec{u} - \nabla p \qquad \nabla \cdot \vec{u} = 0 \qquad \langle a | b \rangle := \int_\Omega a b \, d\Omega$$

$$\boldsymbol{\Omega} \frac{d u_h}{d t} + \boldsymbol{C}(u_h) u_h = \boldsymbol{D} u_h - \boldsymbol{\Omega} \boldsymbol{G} p_h \qquad \boldsymbol{M} u_h = 0_h \qquad \langle a_h | b_h \rangle := a_h^T \boldsymbol{\Omega} b_h$$

Let's consider the time evolution of $1/2 \langle u_h | u_h \rangle$...

$$\frac{1}{2} \frac{d \langle u_h | u_h \rangle}{dt} = u_h^T \boldsymbol{\Omega} \frac{d u_h}{dt} = -u_h^T \boldsymbol{C}(u_h) u_h + u_h^T \boldsymbol{D} u_h - u_h^T \boldsymbol{\Omega} \boldsymbol{G} p_h$$

$$= u_h^T \boldsymbol{D} u_h \leq 0$$

...mimicking the properties of continuous NS eqs leads to

REMAINDER!!!

$$\frac{1}{2} \frac{d \langle \vec{u} | \vec{u} \rangle}{dt} = \langle \frac{\partial \vec{u}}{\partial t} | \vec{u} \rangle = -\langle C(\vec{u}, \vec{u}) | \vec{u} \rangle + \nu \langle \nabla^2 \vec{u} | \vec{u} \rangle - \langle \nabla p | \vec{u} \rangle$$
$$= -\nu \langle \nabla \vec{u} | \nabla \vec{u} \rangle = -\nu \| \nabla \vec{u} \|^2 \leq 0$$
$$= -\nu \langle \nabla \times \nabla \times \vec{u} | \vec{u} \rangle = -\nu \| \omega \|^2 \leq 0$$

Numerical stability !!!

Algebraic operators

$$\frac{1}{2}\frac{d\langle u_h|u_h\rangle}{dt} = u_h^T \boldsymbol{\Omega} \frac{d\,u_h}{dt} = -u_h^T \boldsymbol{C}(u_h) u_h + u_h^T \boldsymbol{D} u_h - u_h^T \boldsymbol{\Omega}\boldsymbol{G} p_h$$

$$= u_h^T \boldsymbol{D} u_h \leq 0 \;, \quad \text{if } \boldsymbol{M} u_h = 0_h, \; \forall u_h, p_h$$

$$u_h^T \boldsymbol{C}(u_h) u_h = 0 \longrightarrow \boldsymbol{C}(u_h) = -\boldsymbol{C}^T(u_h)$$

$$u_h^T \boldsymbol{\Omega}\boldsymbol{G} p_h = 0 \longrightarrow \boldsymbol{\Omega}\boldsymbol{G} = -\boldsymbol{M}^T$$

$$u_h^T \boldsymbol{D} u_h \leq 0 \longrightarrow \boldsymbol{D} = \boldsymbol{D}^T \;\; \text{def-}$$

REMAINDER!!!

$$\frac{1}{2}\frac{d\langle \vec{u}|\vec{u}\rangle}{dt} = \langle \frac{\partial \vec{u}}{\partial t}|\vec{u}\rangle = -\langle C(\vec{u},\vec{u})|\vec{u}\rangle + \nu\langle \nabla^2 \vec{u}|\vec{u}\rangle - \langle \nabla p|\vec{u}\rangle$$

$$= -\nu\langle \nabla\vec{u}|\nabla\vec{u}\rangle = -\nu||\nabla\vec{u}||^2 \leq 0$$

$$= -\nu\langle \nabla\times\nabla\times\vec{u}|\vec{u}\rangle = -\nu||\omega||^2 \leq 0$$

REMAINDER!!!

$$\langle \nabla\cdot\vec{a}|\phi\rangle = -\langle \vec{a}|\nabla\phi\rangle$$

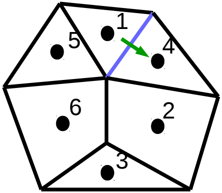
$$\langle \nabla^2 f|g\rangle = -\langle \nabla f|\nabla g\rangle = \langle f|\nabla^2 g\rangle$$

$$\langle C(\vec{u},\phi_1)|\phi_2\rangle = -\langle C(\vec{u},\phi_2)|\phi_1\rangle \quad \text{if } \nabla\cdot\vec{u} = 0$$

$$\langle \nabla\times\vec{a}|\vec{b}\rangle = \langle \vec{a}|\nabla\times\vec{b}\rangle$$

Algebraic operators

$$\Omega \frac{d u_h}{dt} + C(u_h) u_h = D u_h - \Omega G p_h \qquad M u_h = 0_h$$



$$T = \begin{pmatrix} \times & 0 & 0 & \times & \times & 0 \\ 0 & \times & \times & \times & 0 & \times \\ 0 & \times & \times & 0 & 0 & \times \\ \times & \times & 0 & \times & 0 & 0 \\ \times & 0 & 0 & 0 & \times & \times \\ 0 & \times & \times & 0 & \times & \times \end{pmatrix}$$

$$\Omega = \begin{pmatrix} \Omega_u & \\ & \Omega_v \end{pmatrix}$$

$$M = \begin{pmatrix} M^u & M^v \end{pmatrix}$$

$$\Omega G = -M^T$$

$$G = \begin{pmatrix} G^x \\ G^y \end{pmatrix}$$

$$D_u = D_v = M G = -M \Omega^{-1} M^T$$

$$C = \begin{pmatrix} C_u & \\ & C_v \end{pmatrix}$$

$$D = \begin{pmatrix} D_u & \\ & D_v \end{pmatrix}$$

From Calculus to Algebra (C2A)

C2A

$$\frac{\partial \vec{u}}{\partial t} + (\vec{u}\cdot\nabla)\vec{u} = \nu \nabla^2 \vec{u} - \nabla p; \quad \nabla\cdot\vec{u} = 0 \qquad \boldsymbol{\Omega}\frac{d u_h}{dt} + \boldsymbol{C}(u_h)u_h = \boldsymbol{D}u_h - \boldsymbol{\Omega}\boldsymbol{G}p_h; \quad \boldsymbol{M}u_h = 0_h$$

$$\langle a|b\rangle := \int_\Omega ab\, d\Omega \longrightarrow \langle a_h|b_h\rangle := a_h^T \boldsymbol{\Omega} b_h$$

$$\langle C(\vec{u},\phi_1)|\phi_2\rangle = -\langle C(\vec{u},\phi_2)|\phi_1\rangle \longrightarrow \boldsymbol{C}(u_h) = -\boldsymbol{C}^T(u_h)$$

$$\langle \nabla\cdot\vec{a}|\phi\rangle = -\langle \vec{a}|\nabla\phi\rangle \longrightarrow \boldsymbol{\Omega}\boldsymbol{G} = -\boldsymbol{M}^T$$

$$\langle \nabla^2 f|g\rangle = \langle f|\nabla^2 g\rangle \longrightarrow \boldsymbol{D} = \boldsymbol{D}^T \ \text{def-}$$

REMAINDER!!!

$\langle \nabla\cdot\vec{a}|\phi\rangle = -\langle \vec{a}|\nabla\phi\rangle$

$\langle \nabla^2 f|g\rangle = -\langle \nabla f|\nabla g\rangle = \langle f|\nabla^2 g\rangle$

$\langle C(\vec{u},\phi_1)|\phi_2\rangle = -\langle C(\vec{u},\phi_2)|\phi_1\rangle$ if $\nabla\cdot\vec{u} = 0$

$\langle \nabla\times\vec{a}|\vec{b}\rangle = \langle \vec{a}|\nabla\times\vec{b}\rangle$

$u_h^T \boldsymbol{C}(u_h) u_h = 0 \longrightarrow \boldsymbol{C}(u_h) = -\boldsymbol{C}^T(u_h)$

$u_h^T \boldsymbol{\Omega}\boldsymbol{G} p_h = 0 \longrightarrow \boldsymbol{\Omega}\boldsymbol{G} = -\boldsymbol{M}^T$

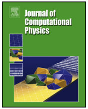
$u_h^T \boldsymbol{D} u_h \leq 0 \longrightarrow \boldsymbol{D} = \boldsymbol{D}^T$ def-

Journal of Computational Physics 258 (2014) 246–267

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp



Symmetry-preserving discretization of Navier–Stokes equations on collocated unstructured grids

F.X. Trias [a,b], O. Lehmkuhl [a,c], A. Oliva [a,*,1], C.D. Pérez-Segarra [a], R.W.C.P. Verstappen [b]

[a] *Heat and Mass Transfer Technological Center, Technical University of Catalonia, ETSEIAT, c/ Colom 11, 08222 Terrassa, Spain*
[b] *Johann Bernoulli Institute for Mathematics and Computing Science, University of Groningen, P.O. Box 407, 9700 AK Groningen, The Netherlands*
[c] *Termo Fluids, S.L., c/ Magí Colet 8, 08204 Sabadell, Spain*

ARTICLE INFO

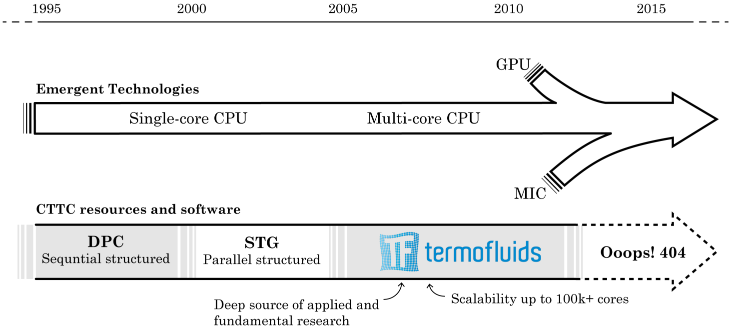
Article history:
Received 19 November 2012
Received in revised form 16 September 2013
Accepted 17 October 2013
Available online 24 October 2013

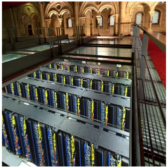
Keywords:
Symmetry-preserving discretization
Collocated formulation
Unstructured grid
Checkerboard
Regularization
Differentially heated cavity

ABSTRACT

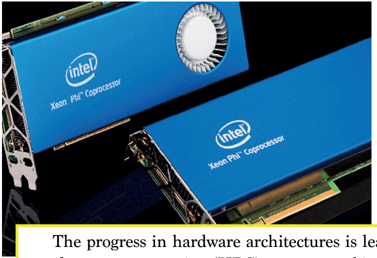
A fully-conservative discretization is presented in this paper. The same principles followed by Verstappen and Veldman (2003) [3] are generalized for unstructured meshes. Here, a collocated-mesh scheme is preferred over a staggered one due to its simpler form for such meshes. The basic idea behind this approach remains the same: mimicking the crucial symmetry properties of the underlying differential operators, *i.e.*, the convective operator is approximated by a skew-symmetric matrix and the diffusive operator by a symmetric, positive-definite matrix. A novel approach to eliminate the checkerboard spurious modes without introducing any non-physical dissipation is proposed. To do so, a fully-conservative regularization of the convective term is used. The supraconvergence of the method is numerically showed and the treatment of boundary conditions is discussed. Finally, the new discretization method is successfully tested for a buoyancy-driven turbulent flow in a differentially heated cavity.

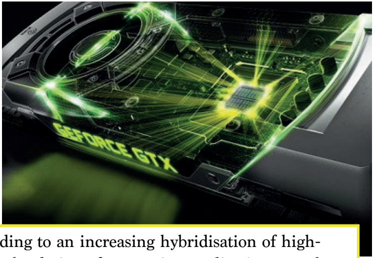
CTTC's historical background in HPC



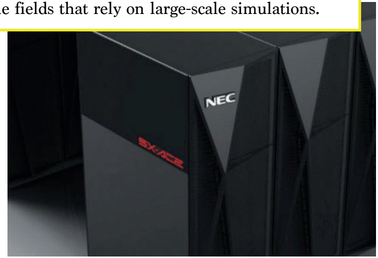


Divergence of HPC systems





The progress in hardware architectures is leading to an increasing hybridisation of high-performance computing (HPC) systems, making the design of computing applications a rather complex problem, and is affecting most of the fields that rely on large-scale simulations.



Divergence of HPC systems

Fully-portable implementation models

Is it **necessary** to use the new hardware architectures?

- In our opinion, **yes**. New hardware is designed to overcome the power constraint in the context of the **exascale** challenge.

Fully-portable implementation models

Is it **necessary** to use the new hardware architectures?

- In our opinion, **yes**. New hardware is designed to overcome the power constraint in the context of the **exascale** challenge.

Do the **traditional** implementation models facilitate code portability?

- In our opinion, **no**. **Legacy codes** were **not designed for** providing **portability** simply because it was not necessary.

Fully-portable implementation models

Is it **necessary** to use the new hardware architectures?

- In our opinion, **yes**. New hardware is designed to overcome the power constraint in the context of the **exascale** challenge.

Do the **traditional** implementation models facilitate code portability?

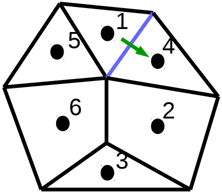
- In our opinion, **no**. **Legacy codes** were **not designed for** providing **portability** simply because it was not necessary.

Do we need to **change** the way we look at scientific computing in general?

- In our opinion, **yes**. Making an effort to design modular applications composed of a **reduced number of independent and well-defined code blocks** helps to reduce the generation of errors and facilitates debugging and **portability**.

My basic blocks: matrices and vectors!

$$\Omega \frac{d u_h}{d t} + C(u_h) u_h = D u_h - \Omega G p_h \qquad M u_h = 0_h$$



$$T = \begin{pmatrix} \times & 0 & 0 & \times & \times & 0 \\ 0 & \times & \times & \times & 0 & \times \\ 0 & \times & \times & 0 & 0 & \times \\ \times & \times & 0 & \times & 0 & 0 \\ \times & 0 & 0 & 0 & \times & \times \\ 0 & \times & \times & 0 & \times & \times \end{pmatrix}$$

$$\Omega = \begin{pmatrix} \Omega_u & \\ & \Omega_v \end{pmatrix}$$

$$M = \begin{pmatrix} M^u & M^v \end{pmatrix}$$

$$\Omega G = -M^T$$

$$G = \begin{pmatrix} G^x \\ G^y \end{pmatrix}$$

$$D_u = D_v = M G = -M \Omega^{-1} M^T$$

$$C = \begin{pmatrix} C_u & \\ & C_v \end{pmatrix}$$

$$D = \begin{pmatrix} D_u & \\ & D_v \end{pmatrix}$$

Fully-portable implementation models

Stencil-based

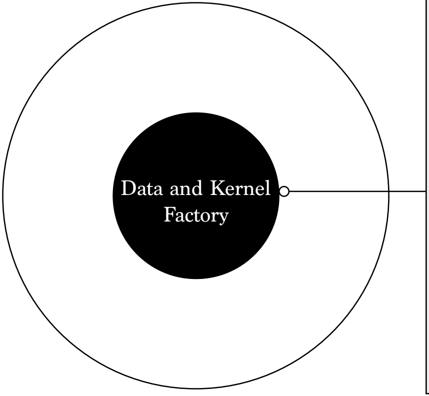
Traditionally, the stencil-based implementations are used by the scientific computing community. These implementations arise straightforward from the formulation of the numerical method. However, they require **specific stencil sweeps and data structures** for each numerical method.

Algebra-based

Algebra-based implementations only rely on a reduced number of **universal algebraic kernels and data structures**, allowing the use of standard optimised libraries and, therefore, providing portability. As a counterpart, the formulation of the numerical method becomes more complex and could even lead to an increase in the number of operations.

The HPC^2 fully-portable, algebra-based framework

The HPC^2 (Heterogeneous Portable Code for HPC) is a fully-portable, algebra-based framework with a multilevel MPI+OpenMP+OpenCL+CUDA parallelisation. Naturally provides **modularity and portability**.



Data and Kernel Factory

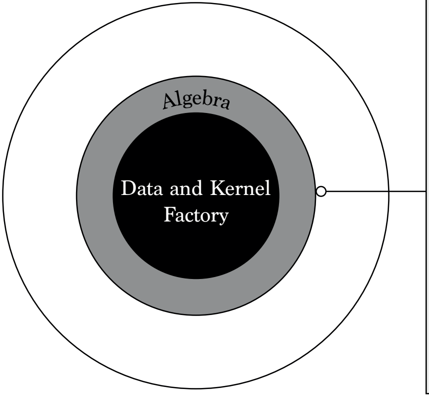
The Data and Kernel Factory layer holds the data containers and the computing kernels.

```
SpMV(A, u, y)
axpy(u, v, y)
ddot(u, v)
```

Its layout, inspired in the **abstract factory** design pattern, employs virtual objects for minimising the dependencies. Thus, it allows adding a new implementation of data containers and computing kernels easily.

The HPC^2 fully-portable, algebra-based framework

The HPC^2 (Heterogeneous Portable Code for HPC) is a fully-portable, algebra-based framework with a multilevel MPI+OpenMP+OpenCL+CUDA parallelisation. Naturally provides **modularity and portability**.



Algebra

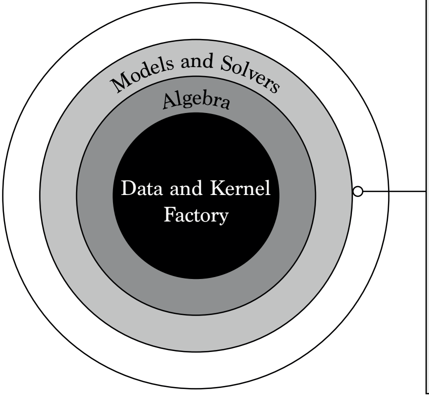
The Algebra layer contains classes and objects that mimic algebraic structures. They wrap data containers and computing kernels from the abstract factories, which makes them entirely independent of the implementation.

```
Set
VectorSpace
DifferentialManifold
```

Differential operators (i.e. DIV, GRAD) can be generated by means of whatever numerical method such as FEM, FVM. They're just matrices.

The HPC^2 fully-portable, algebra-based framework

The HPC^2 (Heterogeneous Portable Code for HPC) is a fully-portable, algebra-based framework with a multilevel MPI+OpenMP+OpenCL+CUDA parallelisation. Naturally provides **modularity and portability**.



Models and Solvers

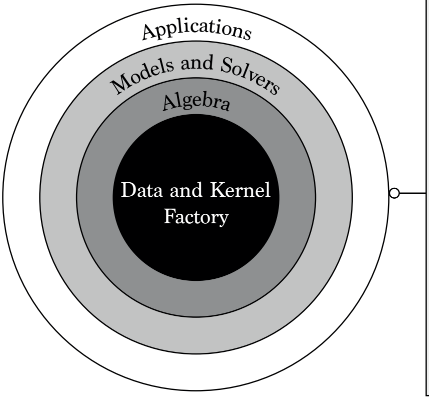
The Models and Solvers layer engages the algebraic objects for generating both physical models and solver methods.

```
NavierStokes
RungeKutta
ConjugateGradient
```

Recall that the numerical method (i.e. FEM, FVM, FDM) is used only at the preprocessing stage for building the matrices.

The HPC^2 fully-portable, algebra-based framework

The HPC^2 (Heterogeneous Portable Code for HPC) is a fully-portable, algebra-based framework with a multilevel MPI+OpenMP+OpenCL+CUDA parallelisation. Naturally provides **modularity and portability**.



Applications

The Application layer is where the user can combine models and solvers to set-up the case for running a simulation.

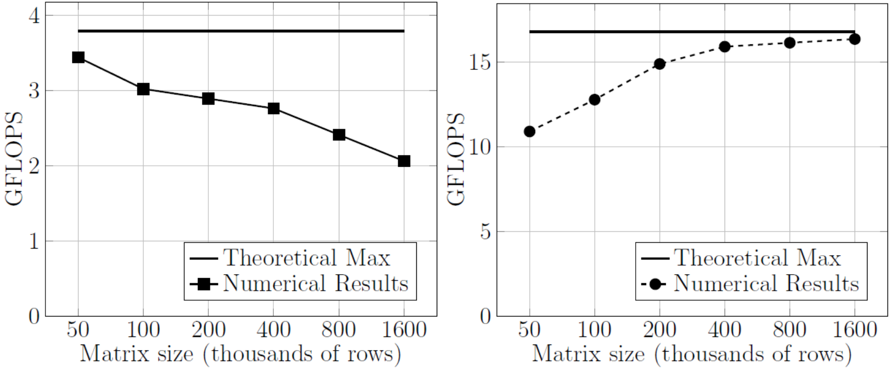
```
DrivenCavity
RisingBubble
FallingFilm
```

In summary, everything that can be expressed by means of algebraic structures such as Matrix and Vector, it can be run within this framework.

A performance overview of the HPC^2

Study case 1

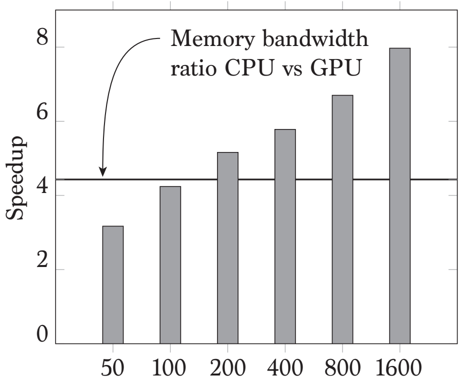
Single-device performance of the SpMV kernel *vs* the matrix size on an Intel Xeon E5649 (left) and Nvidia M2090 (right) for a matrix derived from a symmetry-preserving discretisation[1] on an unstructured hex-dominant mesh.



A performance overview of the HPC[2]

Study case 1

Single-device performance of the SpMV kernel *vs* the matrix size on an Intel Xeon E5649 (left) and Nvidia M2090 (right) for a matrix derived from a symmetry-preserving discretisation[1] on an unstructured hex-dominant mesh.



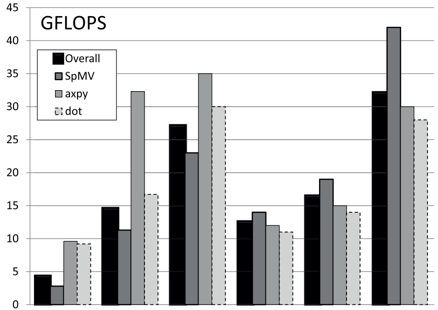
CPU vs GPU

In memory-bounded applications, the GPU performance improves with the size of the matrix, in contrast with that of the CPU. Hence, the speedup depends on both the matrix size and the memory bandwidth.

A performance overview of the HPC^2

Study case 2

Single-device performance comparison of the algebraic DNS algorithm using the symmetry-preserving discretisation[2] on an unstructured hex-dominant mesh of 1M cells.

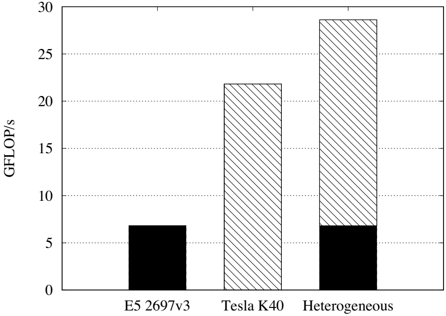


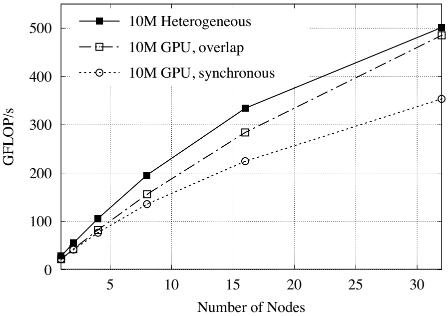
Intel Xeon **E5-2660**	Intel Xeon **8160**	Intel Xeon **Phi 7290**	NVIDIA **2090**	NVIDIA **K40**	AMD **Radeon R9** Nano
8 cores	24 cores	72 cores	178 GB/s	288GB/s	512 GB/s
51GB/s	120 GB/s	400GB/s	0.67TF	1.5TF	0.5 TF
140GF	1.6TF	3.4TF			

A performance overview of the HPC^2

Study case 3

Heterogeneous performance study of the SpMV kernel on a hybrid node equipped with an Intel E5 2697v3 and an Nvidia Tesla K40 for a matrix derived from a symmetry-preserving discretisation[3] on an unstructured hex-dominant mesh of 10M cells. On the left, the single-node performance study. On the right, the strong-scaling study.



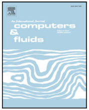


Computers and Fluids 173 (2018) 285–292

Contents lists available at ScienceDirect

Computers and Fluids

journal homepage: www.elsevier.com/locate/compfluid



HPC²—A fully-portable, algebra-based framework for heterogeneous computing. Application to CFD

X. Álvarez [a,*], A. Gorobets [a,b], F.X. Trias [a], R. Borrell [a,c], G. Oyarzun [a]

[a] *Heat and Mass Transfer Technological Center, Technical University of Catalonia, C/ Colom 11, Terrassa (Barcelona) 08222, Spain*
[b] *Keldysh Institute of Applied Mathematics RAS, Miusskaya Sq. 4, Moscow 125047, Russia*
[c] *Termo Fluids, S.L., C/ Magí Colet 8, Sabadell (Barcelona), 08024, Spain*

A R T I C L E   I N F O

Article history:
Received 2 November 2017
Accepted 23 January 2018
Available online 10 February 2018

Keywords:
Heterogeneous computing
MPI+OpenMP+OpenCL
Hybrid CPU+GPU systems
CFD
Symmetry-preserving discretization

A B S T R A C T

The variety of computing architectures competing in the exascale race makes the portability of codes of major importance. In this work, the HPC² code is presented as a fully-portable, algebra-based framework suitable for heterogeneous computing. In its application to CFD, the algorithm of the time-integration phase relies on a reduced set of only three algebraic operations: the sparse matrix-vector product, the linear combination of vectors and the dot product. This algebraic approach combined with a multilevel MPI+OpenMP+OpenCL parallelization naturally provides portability. The performance has been studied on different architectures including multicore CPUs, Intel Xeon Phi accelerators and GPUs of AMD and NVIDIA. The multi-GPU scalability is demonstrated up to 256 devices. The heterogeneous execution is tested on a CPU+GPU hybrid cluster. Finally, results of the direct numerical simulation of a turbulent flow in a 3D air-filled differentially heated cavity are presented to show the capabilities of the HPC² dealing with large-scale CFD simulations.

Concluding remarks

- Symmetry-preserving discretization is based on a **very reduced set of operators (matrices)**. The rest follow straightforwardly by preserving fundamental symmetries.
- We consider that it forms a **solid basis for** testing **sub-grid scale models** (details in next talk).
- Preserving operators symmetries leads to **numerical stability** (in the L2-norm sense).
- An algebra-based approach naturally provides with **modularity and portability**.

Concluding remarks

- Symmetry-preserving discretization is based on a **very reduced set of operators (matrices)**. The rest follow straightforwardly by preserving fundamental symmetries.
- We consider that it forms a **solid basis for** testing **sub-grid scale models** (details in next talk).
- Preserving operators symmetries leads to **numerical stability** (in the L2-norm sense).
- An algebra-based approach naturally provides with **modularity and portability**.

Takeaway message:

- Differential **calculus** and **linear algebra** are intimately connected.

Thank you for your attention