21st Summer School of PARALLEL COMPUTING

July 2 - 13, 2012 (Italian)
September 3 - 14, 2012 (English)

Parallel Algorithms for CFD

Ivan Spisso – i.spisso@cineca.it

SuperComputing Applications and Innovation Department

CINECA

Outline of the presentation

1. Why Computational Fluid Dynamics?
2. Example of massively parallel CFD code
3. Finite Differences (FD) vs Finite Volume (FV)
4. Example of FD scheme: High-order Compact scheme for CAA
5. Example of FV Scheme: OpenFOAM and his parallel aspects
6. Overview of Artificial Boundary Conditions (ABC) for compressible flows

Outline of the presentation

Why Computational Fluid Dynamics?

- To obtain an approximate solution of the Compressible Navier-Stokes (NS) Equations. Set of Partial Differential Equations (PDE)

$$\frac{\partial}{\partial t}\mathbf{U} + \nabla \cdot (\mathbf{F}_c + \mathbf{F}_v) = 0 \tag{1}$$

$$\mathbf{U} = \begin{pmatrix} \rho \\ \rho\mathbf{u} \\ \rho e \end{pmatrix} \quad \mathbf{F}_c = \begin{pmatrix} \rho\mathbf{u} \\ \rho\mathbf{u} \otimes \mathbf{u} + p\mathbf{I} \\ \rho\mathbf{u}\left(e + p/\rho\right) \end{pmatrix} \quad \mathbf{F}_v = \begin{pmatrix} 0 \\ \tau \\ \tau \cdot \mathbf{u} - k_T \nabla \cdot T \end{pmatrix}$$

$$e = \frac{1}{\gamma - 1} RT + \frac{1}{2}\mathbf{u} \cdot \mathbf{u} \qquad \tau = \mu\left(\nabla\mathbf{u} + \mathbf{u} \otimes \nabla - 2/3\mathbf{I}\nabla \cdot \mathbf{u}\right)$$

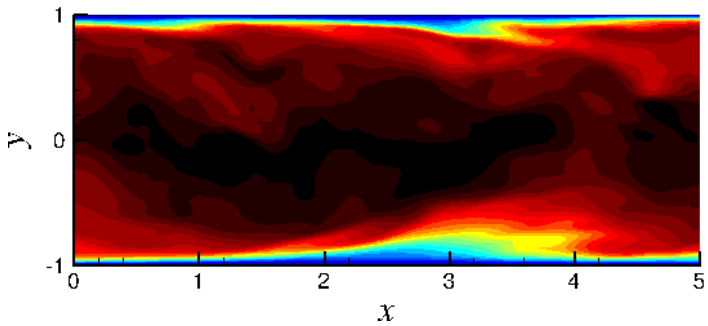
- No exact solution, except simple cases.
- The Clay Mathematics Institute has a 1 million $prize waiting for the first person to solve this problem

www.claymath.org/millennium/Navier-Stokes_Equations/

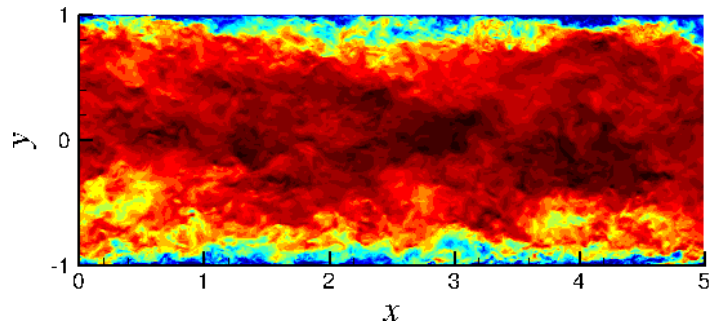
Outline of the presentation

DNS of NS Equations

- DNS (Direct Numerical Simulation) of turbulent flows
  - Research group of Pirozzoli, Orlandi, Bernardini, Grasso. Dipartimento di Ingegneria Meccanica ed Aerospaziale, Universita' di Roma "La Sapienza"
- Extremely **challenging** and **expensive**
  - Large-scales dictated by the characteristic size of the problem
  - Small-scales dictated by the **Reynolds number** ($Re = U\,L/\nu$)



$$Re = \frac{U_c\, h}{\nu} = 4300$$



$$Re = \frac{U_c\, h}{\nu} = 65200$$

Pirozzoli, S., 2011, Numerical methods for high-speed flows, Annu. Rev. Fluid Mech., 43, 163-194

DNS of NS Equations

Different approach to model N-S equations
(DNS,LES,DES,RANS,URANS)

- Direct Numerical Simulation (DNS)
  - Solving the N-S equations without turbulence models
  - All scales of motion are properly resolved on the computational mesh
  - Extremely **high computational cost** (limited to low and moderate Reynolds numbers)
  - Code requirements
    - High-order accurate, low-dissipative, but stable algorithms required
    - Computational efficiency **mandatory** for large-scale computations
    - High scalability performance for parallel computations

Flow solver description

- Programme Language
  - Flow solver written in Fortran 90
- Numerical strategy
  - NS Equations are solved using the Finite Differences (FD) schemes
  - Structured Cartesian Domain of dimension $L_x \times L_y \times L_z$
  - Equations discretized on a Cartesian grid with $N_x \times N_y \times N_z$ points
- Method of the lines approach
  - Space and Time integration handled separately in the set of PDE
  - Discretization of spatial derivatives
  - Set of Ordinary Differential Equations (ODE)
  - Runke-Kutta (RK) scheme for time integration

Flow solver description

Spatial Derivatives

- Convective Terms
  - Convective Terms F_c in eq.(1) are **non linear**
  - Tendency to form steep gradients in compressible flows (shock waves)
  - Non-linear instability
  - **Hybrid Approach** for shock capturing
    - Central Differences in the smooth regions
    - WENO reconstructions in shocked regions (adaptive stencils)
    - Switch controlled by a shock sensor $0 \leq \Theta \leq 1$ to detect discontinuities
- Viscous Terms
  - Viscous Terms F_v in eq.(1) are linear, no particular problems
  - Direct applications of central differences formulas

Flow solver description

Time integration-Runge-Kutta scheme

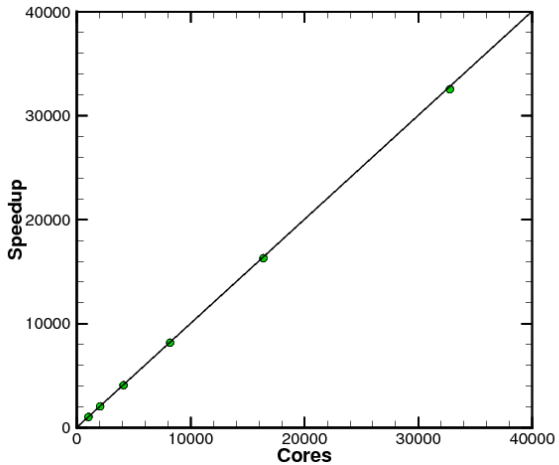
- Low storage Runge-Kutta scheme
- Two memory spaces allocated (10 three-dimensional arrays)

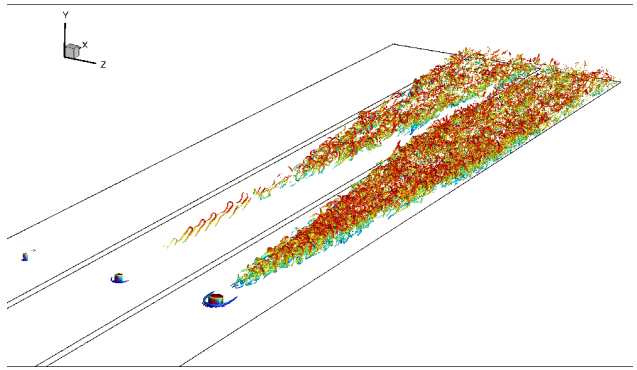
Code Parallelization

  - Pure **MPI** parallelization
  - Block decomposition of the computational domain
  - Each core is assigned to a specific core
  - **Explicit schemes** allow to exchange only planes. No global transposition of data
- Virtual MPI Topology
  - Cartesian 3D topology
  - Cartesian 1D communicator in the x-direction
  - Advantanges of Cartesian topologies: Easier and faster implementation of communications, reduction of the number of data to exchange

Flow solver description

- Weak Scaling
  - Variation of the solution time with the number of processors for a fixed problem size
  - **High-scalability**, Speed-up $= npT_s/T_{np}$ @ BlueGene-Q, Julich.
  - Linear scalability up to 32.768 cores



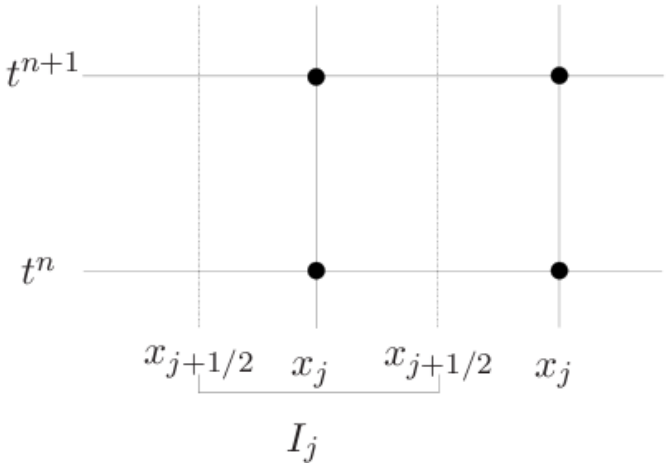


Roughness-induced boundary layer transition

Outline of the presentation

Finite Differences (FD) vs Finite Volume (FV)

- Uniformly spaced mesh with h grid spacing in x, k time step constant.
- Nodes $x_j = jh$, intermediate nodes $x_{j+1/2} = x_j + \frac{h}{2}$, interval $I_j = \left[x_{j-1/2}; x_{j+1/2}\right]$, $j = \ldots, -1, 0, 1, 2, \ldots$
  $t^n = nk$, $n = 0, 1, 2, \ldots$



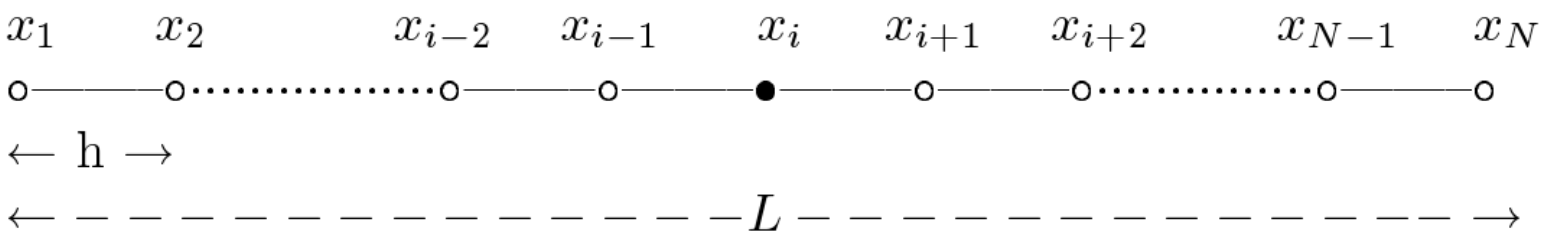
- Finite difference $U_j^n = v\left(x_j, t^n\right)$
- Finite Volume
  $U_j^n = \frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} v\left(x_j, t^n\right) dx$

$$v\left(x_j, t^n\right) = \frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} v\left(x_j, t^n\right) dx + O(h^2)$$

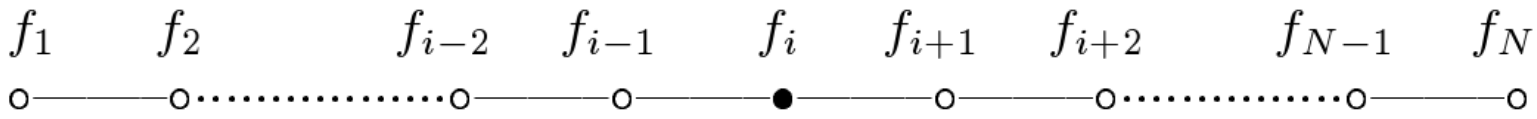
Outline of the presentation

Finite Difference Approximation (FD)

- Uniformly spaced mesh where the nodes are indexed as i, $h = L/(N-1)$ grid spacing, for $1 \leq i \leq N$.
- Independent variable at the nodes is $x_i = (i-1)h$



- Given the values of a function on a set of nodes, the finite difference approximation to the derivative of the function is expressed as a linear combination of the given function values.



Finite Difference Approximation (FD)

- The finite difference approximation f_i' to the first derivative $(df/dx)(x_i)$ at the node i, using a $(q+r+1)$ point stencil, depends on the function values at the nodes near i:

$$\sum_{L=-Q}^{R} \alpha_L \, f'_{i+L} = \frac{1}{h} \sum_{\ell=-q}^{r} a_\ell \, f_{i+\ell} + O(h^n) \qquad (2)$$

$$(q+r) \geq 1;\ Q \leq q \text{ and } R \leq r$$

- $q = r$ the $(q+r+1)$ stencil is symmetric $\Rightarrow$ **Central** FD
- $q < r$ the $(q+r+1)$ stencil is asymmetric $\Rightarrow$ **Forward** FD
- $q > r$ the $(q+r+1)$ stencil is asymmetric $\Rightarrow$ **Backward** FD
- $Q = R = 0$ **explicit** schemes.
- $Q = R \neq 0$ **implicit**, compact or Pade' schemes.
- the non-optimized classical FD will be denoted as $CQRqr$

FD Approximation in matrix form

- the system of eqs. (2) can be rewritten in matrix form as:

$$
\begin{pmatrix} \dots \\ \alpha_{-Q} \\ \dots \\ \alpha_{-1} \\ \alpha_0 \\ \alpha_1 \\ \dots \\ \alpha_R \\ \dots \end{pmatrix}
\begin{pmatrix}
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\dots & 0 & f'_{i-Q} & \dots & f'_{i-1} & f'_i & f'_{i+1} & \dots & f'_{i+R} & 0 & \dots & \\
 & \dots & 0 & f'_{i-Q} & \dots & f'_{i-1} & f'_i & f'_{i+1} & \dots & f'_{i+R} & 0 & \dots \\
 & & \dots & 0 & f'_{i-Q} & \dots & f'_{i-1} & f'_i & f'_{i+1} & \dots & f'_{i+R} & 0 \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots
\end{pmatrix}
\cong
$$

$$
\frac{1}{h}
\begin{pmatrix} \dots \\ a_{-q} \\ \dots \\ a_{-1} \\ a_0 \\ a_1 \\ \dots \\ a_r \\ \dots \end{pmatrix}
\begin{pmatrix}
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\dots & 0 & f_{i-q} & \dots & f_{i-1} & f_i & f_{i+1} & \dots & f_{i+r} & 0 & \dots & \\
 & \dots & 0 & f_{i-q} & \dots & f_{i-1} & f_i & f_{i+1} & \dots & f_{i+r} & 0 & \dots \\
 & & \dots & 0 & f_{i-q} & \dots & f_{i-1} & f_i & f_{i+1} & \dots & f_{i+r} & 0 \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots
\end{pmatrix}
$$

An intuitive introduction to FD...

- Starting from eq. (2)
- Explicit, second-order central FD, 3 point stencil $Q = R = 0$, $q = r = 1$, $C0011$

$$f_i' \cong \frac{1}{h}\left[a_{-1} f_{i-1} + a_1 f_{i+1}\right] \tag{3}$$

- The function f_{i-1} and f_{i+1} are Taylor series expanded with a truncation error of the order of $\sim O(h^2)$:

$$f_{i+1} \cong f_i + h f_i' + O(h^2)$$
$$f_{i-1} \cong f_i - h f_i' + O(h^2)$$

....and the big O notation

- By substituting the Taylor series expansion in eq. (3):

$$f_i' = \frac{1}{h}\left[(a_{-1} + a_1)f_i + h(a_{-1} + a_1)f_i'\right]$$

- And equating member by member:

$$\begin{cases} a_{-1} + a_1 = 0 \\ -a_{-1} + a_1 = 1 \end{cases}$$

- $\Rightarrow a_1 = -a_{-1} = 1/2$. Note the the relation $a_{-p} = -a_p$ is valid as long as $p = q$.
- And so the eq. (3) can be rewritten in

$$f_i' = \frac{(f_{i+1} - f_{i-1})}{2h} + O(h^2)$$

Fourier analysis of FD schemes

The finite difference equation (2) is a special case of the following equation with respect to the continuous variable x:

$$\sum_{j=-P}^{Q} \alpha_j f'(x + j\,h) = \frac{1}{h} \sum_{j=-R}^{S} a_j f(x + j\,h) + O(h^n), \qquad (4)$$

which discretizes into eq. (2) by setting $x = x_i$. The Fourier transform of the function $f(x)$ is:

$$\tilde{f}(k) = \int_{-\infty}^{\infty} f(x)\, e^{-\mathrm{i}\,k\,x} dx, \qquad (5)$$

where $\mathrm{i} = \sqrt{-1}$, k is the wavenumber, and the tilde "~" represents the transformed function.

Fourier analysis of FD schemes

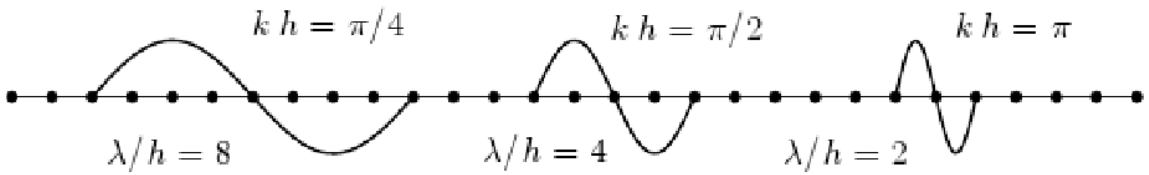
Given a monochromatic wave of wavelength λ, resolved with N_λ number of points per wavelength, the equation

$$N_\lambda = \frac{\lambda}{h} = \frac{2\pi}{\kappa}, \tag{6}$$

is used to relate the scaled wavenumber

$$\kappa = k\,h \tag{7}$$

to the wavelength λ. The Figure shows an example of monochromatic wave, of wavelength λ, resolved respectively with $N_\lambda = 8, 6$ ans 4



Fourier analysis of FD schemes

Taking the Fourier transform of both sides of eq. (4) gives:

$$\bar{\kappa}(\kappa) = \bar{k}(k)\,h = \frac{1}{\mathrm{i}} \frac{\sum_{j=-R}^{S} a_j e^{\mathrm{i}\,j\,\kappa}}{\sum_{j=-P}^{Q} \alpha_j e^{\mathrm{i}\,j\,\kappa}}, = \frac{\sum_{j=-K}^{S} a_j \sin(j\kappa)}{1 + \sum_{j=-Q}^{R} a_j \cos(j\kappa)} \qquad (8)$$

where $\bar{\kappa} = \bar{k}\,h$ is the scaled pseudo-wavenumber. The scaled wavenumber and the scaled pseudo-wavenumber are both non-dimensional values, $\kappa \in \mathbb{R}$, $0 < |\kappa| \leq \pi$, and generally $\bar{\kappa} \in \mathbb{C}$, with real and imaginary part $\Re[\bar{\kappa}]$ and $\Im[\bar{\kappa}]$. It is desirable to make $\bar{\kappa}$ equal to κ. It is **impossible** to build up a perfect match between $\bar{\kappa}$ and κ over the entire wavenumber range due to the limitation of numerical discretization. The coefficients aj, α_j of eq. (8) may be chosen: to minimize the truncation error (the big O), by truncation of the Taylor series or to optimize the behaviour in wavenumber space so as to have $\bar{\kappa}(\kappa) \simeq \kappa$ at the expense of the formal order of accuracy or to reduce the computational cost for a given level of error.

Fourier analysis of FD schemes

In practice, the scaled pseudo-wavenumber $\bar{\kappa}$ implies a certain deviation from the true scaled wavenumber κ, which increases as $\kappa \to \pi$ (for $\kappa = \pi$, $\bar{\kappa} = 0$, see Fig. **??**). This deviation results in spatial numerical error:

$$e_0(\kappa) = \left| \frac{\bar{\kappa}(\kappa) - \kappa}{\kappa} \right|, \tag{9}$$

where the real part represents the dispersive error

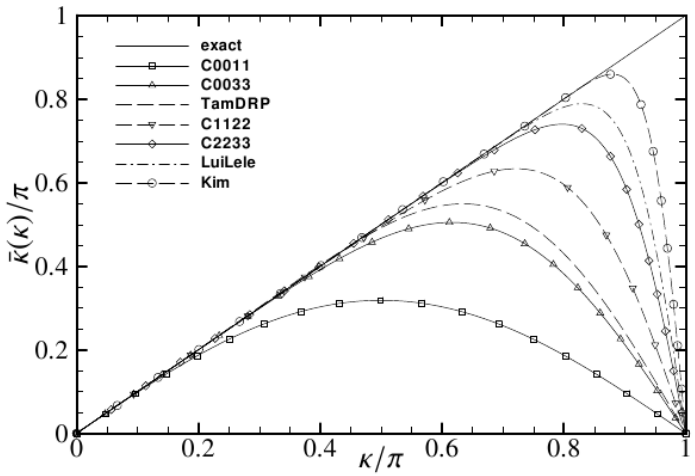
$$\varepsilon_R(\kappa) = \left| \frac{\Re\left[\bar{\kappa}(\kappa)\right] - \kappa}{\kappa} \right|, \tag{10}$$

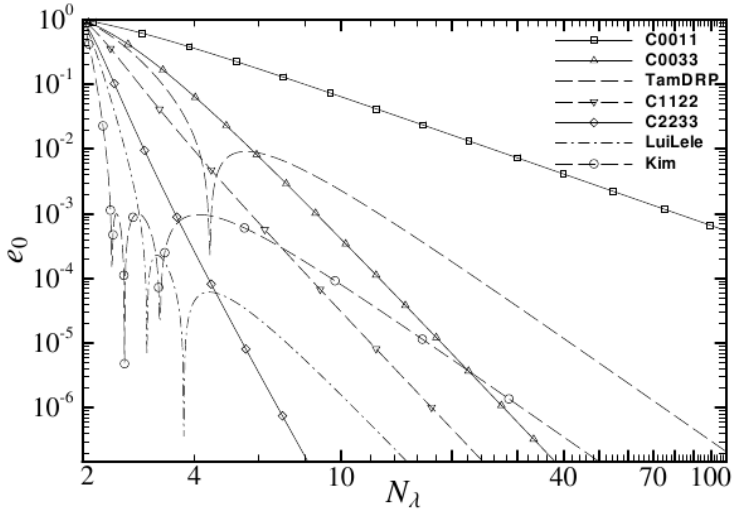
and the imaginary part the dissipative error

$$\varepsilon_I(\kappa) = \left| \frac{\Im\left[\bar{\kappa}(\kappa)\right]}{\kappa} \right|. \tag{11}$$

Dispersive characteristics of the schemes

- schemes are centered, that is $\varepsilon_I(\kappa) = \left| \dfrac{\Im\left[\bar{\kappa}(\kappa)\right]}{\kappa} \right| = 0.$





Explicit vs Compact Schemes

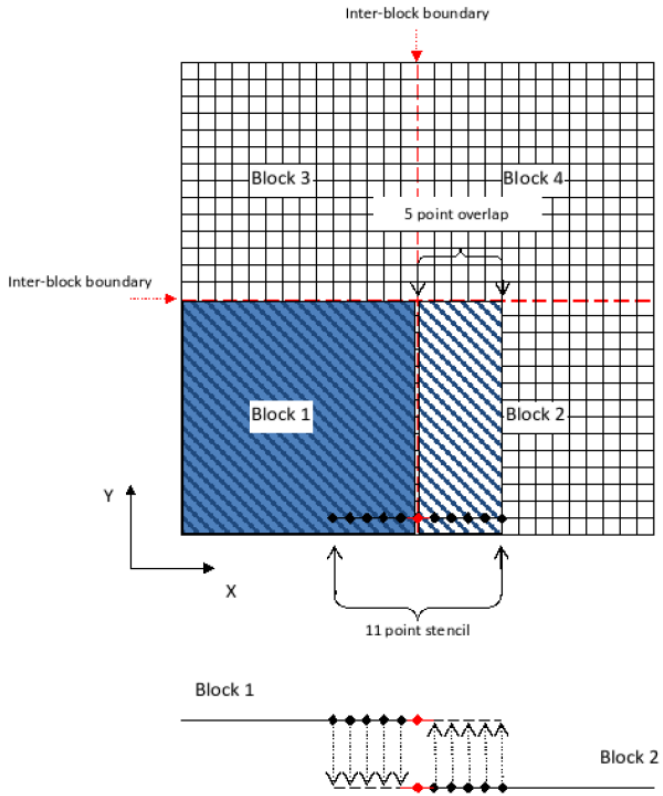
- Explicit schemes employ large computational stencil for a given level of accuracy
- Compact schemes use smaller stencil by solving a linear system at each grid point
- Compact schemes are more accurate than the equivalent formal order of accuracy explicit scheme
- A matrix has to be inverted to obtain the spatial derivative at each grid point
- Boundary stencil has a large effect on the stability and accuracy of the scheme
- Correct implementation and non-trivial parallelization

Parallelization

$$\frac{\partial \mathbf{U}}{\partial t} + A_0 \frac{\partial \mathbf{U}}{\partial x} + B_0 \frac{\partial \mathbf{U}}{\partial y} = 0 \qquad (12)$$

- LEE (Linearized Euler Equations)
- Domain decomposition Each block use 1 proc
- Communication method by finite-sized overlaps
- At every time-step solution computed independently in each block with interior/boundary closure
- Up-date at every RK sub-iteration
- Interior scheme sixth-order compact
- 11 point stencil inter-block boundary
- Method of communication suitable for explicit scheme
- Compact scheme, introduce an error given by the different dispersion characteristics.

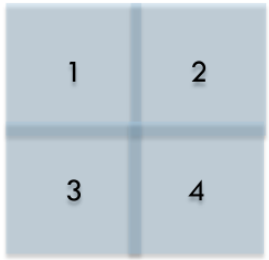
Parallelization



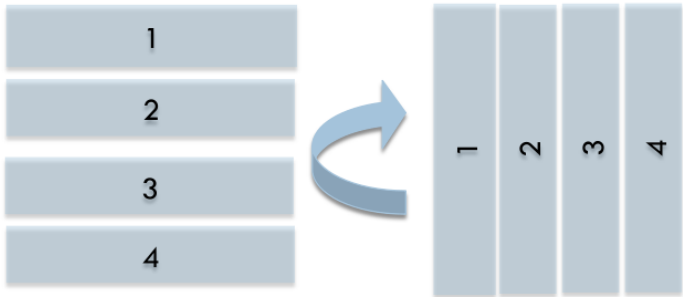
I. Spisso, P. Ghillani, A Rona: Development of multi-block interface for a high-order compact scheme applied to sound scattering problems in aeronautics: Parallelization strategy and efficiency, Science and SuperComputing in Europe, Report 2009.

Parallelization

- Explicit FD, share data where stencil overlap



- Compact FD, data transpose. Slice division for given direction of parallelization (slub decomposition). Use *MPI_Alltoall* to transpose data.



Outline of the presentation

OpenFOAM

- OpenFOAM The Open Source Computational Fluid Dynamics Toolbox
  - Version 2.1.x by OpenCFD Ltd
  - Version 1.6.ext by Wikki
  - The OpenFOAM (Open Field Operation and Manipulation) CFD Toolbox is a free, open source CFD software package.
  - OpenFOAM has an extensive range of features to solve anything from complex fluid flows involving chemical reactions, turbulence and heat transfer, to solid dynamics and electromagnetics.
  - It includes tools for meshing, notably snappyHexMesh, and for pre- and post-processing
  - Almost everything (including meshing, and pre- and post-processing) runs in parallel.
  - OpenFOAM includes over 80 solver applications that simulate specific problems in engineering mechanics and over 170 utility applications that perform pre- and post-processing tasks, e.g. meshing, data visualisation, etc.
  - Free-to-use means using the software without paying for license and support, including massively parallel computers: free 1000-CPU CFD license!

OpenFOAM

- Why OpenSource CFD?
  - Academic rationale for open source code is clear: open collaboration and sharing
  - Industrial users rely on commercial software with strict quality control and dedicated support teams
  - ...but its flexibitily is not sufficient, development is too slow, support quality varies and parallel CFD licences has to paid
- Open Source CFD Solution
  - Reminder: Open Source and GPL does not imply zero price
  - Computer time is still expensive but cost is unavoidable
  - Software support, help with running and customisation is still required
  - Engineers running the code are the most costly part: better!
- Mode of operation
  - When a CFD works well in a design process, it will be used in large volume
  - Development and validation may need to be funded by user but further cost drops significantly: no annual license fee to pay
  - Parts of acceptance and validation effort become responsibility of the user

OpenFOAM parallel aspects

PRACE 1IP - Task 7.2

. . . will provide petascaling expertise to ensure that key application codes can effectively exploit Tier-0 systems. Will identify opportunities to enable applications through engagement with selected scientific communities, industrial users and specific application projects . . .

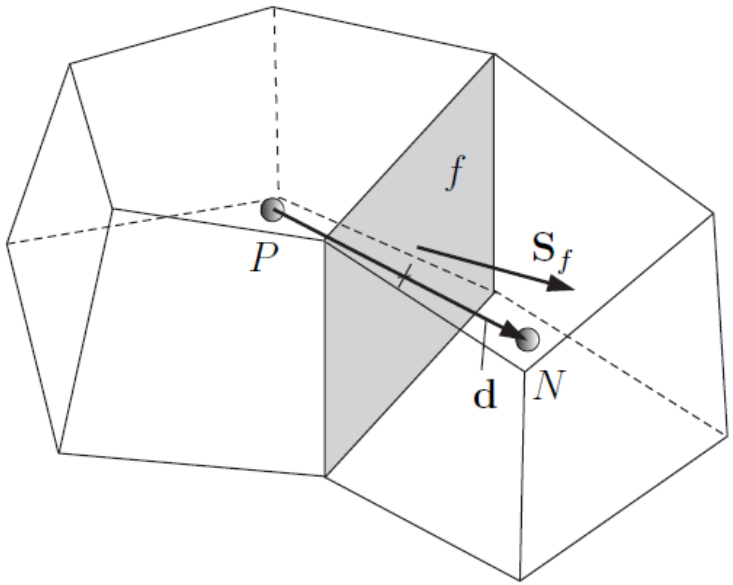
PRACE 1IP - Task 7.5

. . . will work with users to implement new programming techniques, paradigms and algorithms for Tier-0 systems, which have the potential to facilitate significant improvements in the performance of their applications. This task will work in close collaboration with tasks 7.1 and 7.2 . . .

Finite Volume Discretization

- Direct discretization of the integral form of the conservation laws.
  $\int_{V_i} \frac{\partial \mathbf{U}}{\partial t} dV + \int_{V_i} \nabla \cdot \mathbf{F}_c dV = 0$
- Finite-volume flux vector discretization. Applying the Gauss divergence theorem $\frac{\partial}{\partial t} \int_{V_i} \mathbf{U} dV + \oint_{S_i} \mathbf{F}_c \cdot \mathbf{n} dS = 0$
  $\mathbf{U}_i = \frac{1}{V_i} \int_{V_i} \mathbf{U} dV$, $\oint_{S_i} \mathbf{F}_c \cdot \mathbf{n} dS = \sum_{k=1}^{N_{faces}} \mathbf{F}_{c,k} \cdot \mathbf{n}_{i,k} \mathbf{S}_{i,k}$
  $V_i \frac{\partial \mathbf{U}_i}{\partial t} + \mathbf{R}_i = 0$
- To solve the Euler or NS equations, the residual operator $\mathbf{R}_i$ needs to linearize the flux vector $\mathbf{F}_c$. The Godunov method, or Flux Difference Splitting, is used. Interface fluxes normal to the finite-volume unit cell boundaries are tipically estimated by the approximate Riemann solver based on Roe (1981). To reduce the excessive artificial dissipation MUSCL or TVD.

Finite Volume Discretization



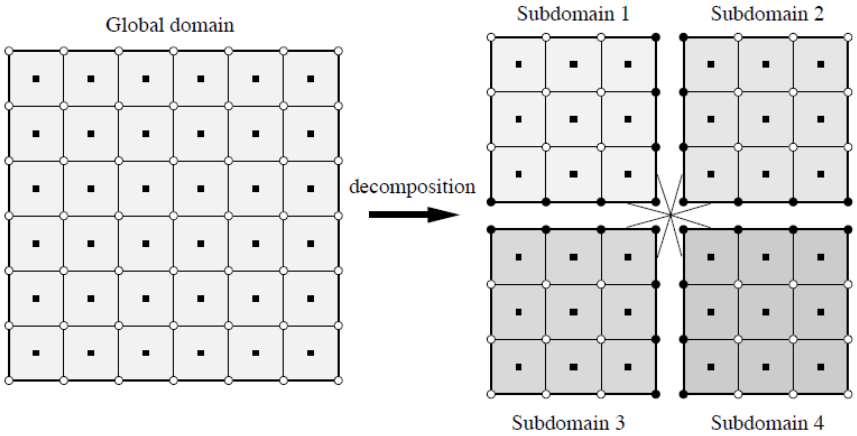
Finite Volume Primer

$$\int_{Cell} \mathrm{div}(A\nabla p)\ dV := \sum_{Faces} \int_f (A\nabla p) \cdot \mathbf{S}_f\ dS$$

Data Representation

- Variables may be associated with:
  1. cells
  2. faces
  3. points
- Time discretized independently

Finite Volume Discretization



Solution Process: Overview

1. Different solvers, same kernel
   - sparse linear systems
   - iterative Krylov methods
   - SpMV multiplication

Task Decomposition

- Zero-Halo layer approach
- Internal Edges
  1. treated as BCs
- Usual pattern
  1. Interface initialization
  2. Local computation
  3. Interface update
- Local: Diagonal Block
- Interface: Off-Diagonal Blocks

Preconditioned Conjugate Gradient

Base Algorithm

1: $\mathbf{r}_0 \leftarrow \mathbf{b} - \mathbf{A}\mathbf{x}_0$
2: $\mathbf{z}_0 \leftarrow \mathbf{M}^{-1}\mathbf{r}_0$
3: $\mathbf{p}_0 \leftarrow \mathbf{z}_0$
4: $k \leftarrow 0$
5: **repeat**
6: $\quad \alpha_k \leftarrow \dfrac{\mathbf{r}_k^{\mathrm{T}}\mathbf{z}_k}{\mathbf{p}_k^{\mathrm{T}}\mathbf{A}\mathbf{p}_k}$
7: $\quad \mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + \alpha_k \mathbf{p}_k$
8: $\quad \mathbf{r}_{k+1} \leftarrow \mathbf{r}_k - \alpha_k \mathbf{A}\mathbf{p}_k$
9: $\quad \mathbf{z}_{k+1} \leftarrow \mathbf{M}^{-1}\mathbf{r}_{k+1}$
10: $\quad \beta_k \leftarrow \dfrac{\mathbf{z}_{k+1}^{\mathrm{T}}\mathbf{r}_{k+1}}{\mathbf{z}_k^{\mathrm{T}}\mathbf{r}_k}$
11: $\quad \mathbf{p}_{k+1} \leftarrow \mathbf{z}_{k+1} + \beta_k \mathbf{p}_k$
12: $\quad k \leftarrow k + 1$
13: **until** $(\|\mathbf{r}_{k+1}\| < tol)$

Key Operations at Each Step

1. Sparse Matrix-Vector multiplication
1. Preconditioning
3. Scalar products

Zero-Halo Layer Scalar Product

```
scalar SumProd     = 0;
scalar partialSum  = 0;
// Local part of the product
for(label ii=0; ii<max; ii++)
   partialSum += f1p[ii] * f2p[ii];
// Gather other tasks contribution
MPI_Allreduce(&SumProd,&partialSum,1
MPI_SCALAR,MPI_SUM,MPI_COMM_WORLD);
```

Lessons learned so far...

1. The scalar products in the PCG algorithm act as barriers
2. The whole bunch of MPI_Allreduce
   - stems from an algorithmic constraint
   - is unavoidable, unless we venture on an algorithmic rewrite
3. How can we reduce communication and preserve the algorithm?
   - add multi-threading capabilities to sparse linear-algebra
4. To do that we have to dig into sparse matrix representation!

Outline of the presentation

Artificial Boundary Conditions

- **Why you need Artificial Boundary Conditions?**
- **Artificial Boundary Conditions**
  - **Characteristic Based Methods**
  - **Buffer Zone Method**
  - **Asymptotic Expansion Method**
  - **Perfectly Matched Layer**

Why ABC?

Why ABC?

God gave you the interior scheme, and Evil put the Boundary Conditions!

Colonius, T., "Modeling artificial boundary conditions for compressible flow", Annual Review of Fluid Mechanics, vol. 40, pp. 315–345, 2004.
Literature cited more than 100 papers!

Characteristic Based Methods

- **1D Characteristic Based Boundary Conditions**
- Thompson's Characteristic Based Boundary Conditions
  Thompson, K. W., Time Dependent Boundary Conditios for Hyperbolic System, J. Comp. Phys. 68 (1987) 1-24

$$\frac{\partial U}{\partial t} + A\frac{\partial U}{\partial x} = 0$$

$$\mathbf{U} = \begin{bmatrix} \rho \\ p \\ u \end{bmatrix}, \quad \mathbf{A_0} = \begin{bmatrix} u & 0 & \rho \\ 0 & u & \gamma p \\ 0 & 1/\rho & u \end{bmatrix}$$

By using eigenvectors and diagonalization, we get the primitive equations

$$l_i^T \frac{\partial U}{\partial t} + L_i = 0, \ , \quad L_i = \lambda_i l_i^T \frac{\partial U}{\partial x}$$

Determinations of L_i:

When the characteristic velocity λ_i points out of the solution volume, compute the corresponding L_i from the definition given above, using one-sided derivative approximations. When λ_i points into the domain, specify the value from the boundary conditions.

- **Multi-Dimensional Characteristic Based Boundary Conditions**
- *Giles, M. B.: 'Non-reflecting boundary conditions for Euler Equation calculation', AIAA J. 18 (1990) 2050-2058*
- *Rowley, C. W., Colonius, T.: 'Numerically Non-reflecting boundary conditions for multidimesional Aeroacoustic Coomputations', AIAA paper 4^{th} AIAA/CEAS Aeroacoustics Conference, 1998*
- *Poinsot & Lele: 'Boundary Conditions for direct simulations of compressible viscous flows' JCP, vol. 101, no. 1, 1992*
  LODI Approximation: Local One Dimensional Inviscid

Buffer Zone Method

- **Buffer Zone Method**

Several absorbing layers have been suggested to

- enhance the efficacy of an ABC or
- to obviate the need for any ABC.

Absorbing layer treatments typically provide for damping of disturbances prior to interact with an Artificial Boundary Condition.
Some obvious ways to do this are the to introduce artificial dissipation (by upwinding), to increase the value of physical viscosity (or add hyperviscosity), and perhaps most simply, to add a linear friction coefficient to the governing equations.

$$\tilde{\mathbf{Q}} = \mathbf{Q} - \sigma\left(\mathbf{Q} - \mathbf{Q}_{target}\right) \ , \quad \sigma(l) = \alpha_2 \left(\frac{L-l}{L}\right)^{\beta_1}$$

where $\tilde{\mathbf{Q}}$ is the damped solution vector and $\mathbf{Q}_{target}$ is a given reference.

Buffer Zone Method

- *Chen, X. X. and Zhang, X. and Morfey, C. L. and Nelson, P. A., "A numerical method for computation of sound radiation from an unflanged duct", Journal of Sound and Vibration, vol. 270, no. 3, pp. 573–586, 2003.*
- *Sandberg R. D., Sandham N. D. "Nonreflecting Zonal Characteristic Boundary Condition for Direct Numerical Simulation of Aerodynamic Sound", AIAA Journal, TN, vol. 44, no. 2, February 2006.*

Asymptotic Expansion Method

Assume that far-field equations reduce to simple model equations
Construct asymptotic solutions to model equations
Derive differential equations for asymptotic solutions
More accuracy as boundary moves away from source region

- Radiation Boundary Conditions
- Decompose the linearized Euler equations into their exact solutions - for acoustic waves,

$$\left(\frac{1}{V(\theta)}\frac{\partial}{\partial t}+\frac{\partial}{\partial r}+\frac{1}{2r}\right)\begin{bmatrix}\rho_a\\ u_a\\ v_a\\ p_a\end{bmatrix}=0+O(r^{-5/2}),$$

$$V(\theta)=c_o\left[M\cos\theta+\left(1-M^2 sin^2\theta\right)^{1/2}\right]$$

Similar expression for the outflow boundary conditions. When a flow exists through a boundary, acoustic, entropy, and vorticity waves must exit.

Asymptotic Expansion Method

- *Tam, C. K. W. and Web, J. C., "Dispersion-Relation-Preserving finite difference schemes for Computational Acoustics", Journal of Computational Physics, vol. 107, pp. 262–281, 1993.*
- *Tam, C., and Dong, Z., Radiation and Outflow Boundary Conditions for Direct Computation of Acoustic and Flow Disturbances in a Nonuni-form Mean Flow, Journal of Computational Acoustics, Vol. 4, No. 2, 1996, pp. 175201.*

Perfectly Matched Layer PML

- **Perfectly Matched Layer (PML)**

Originally developed for Maxwells equations electromagnetics
Extended to Linearized Euler equations by Hu
Surround computational domain with absorbing that damps outgoing waves, but allows waves to enter smoothly with no reflection = perfectly matched
Formulated in split and unsplit forms