**BROADCAST (multi-Blocks stRuctured high-Order Algorithmically Differentiated Compressible nAvier-stokes Stability Tools)**

C.Content ONERA DAAA-CLEF (21/01/2021)

*BROADCAST is a software suite that includes: a 2.5D compressible CFD solver and Stability Tools that include Jacobian extraction, Global stability analysis, Resolvent analysis.*

**Description of numerical methods in use in BROADCAST**

Solve compressible NS or RANS (SA-neg) 2.5D: 2D in multiblocks structured FV framework and a collocated spectral discretization (Fourier or Chebyshev) in the 3rd direction. Here, we describe the 2D Structured FV solver:

**System of equations under consideration:**

*w*: conservative state, *F*: Physical Fluxes (Convective and Viscous), *S*: Source term

The Cartesian formulation of this system can be found in any phD thesis, theorical manual or paper that deals with NS compressible. In axisymmetric formulation this system writes:

With  
; ;

; ; ;

; ; ;

**Numerical schemes:**

*Convective schemes:*

Cartesian equations and axisymmetric equations are discretized within several convective schemes:DNC (Direct Non-Compact, Lerat-Corre, VKI 2006, Cinnella-Content JCP 2016) order up to 9.

Shock capturing technique based on Jameson-Ducros sensor (**, see eq hereafter) is added on DNC. The way it is incorporated for DNC scheme is described in Sciacovelli et al. CF 2021.

Rossow like dissipation is incorporatedin DNC scheme to ensure a low-Mach number correction.

*Viscous scheme:*

When convective fluxes are made on a stencil larger than 3 points, viscous fluxes are discretized by means of compact 5 points scheme of 4th order (otherwise, 3 points- 2nd order scheme is used). We describe hereafter the computation of primitives’ variables gradients needed in viscous fluxes expressions:

**S**

**N**

E

**W**

i

j

**E**

*Figure 1: Notations for primitives’ gradients evaluations at interface*

Where

With

**Boundary conditions:**

Wall: Dirichlet on velocities, Neumann on pressure (as we deals with viscous fluids, we assume ). We can choose to use off-centered schemes near the boundary or continue to use central scheme with ghost cells (ill-posed).

Mirror: algebraic symmetry respectively to the boundary edge (not geometric symmetry).

Extrapolation: Usefull for supersonic outflow. Order 2, 3, 4, 5, 7 or 9.

Dirichlet: Usefull for supersonic Inflow. Impose several ghost cells.

No-Reflexion: fill all ghost cells with a target value at face center based on characteristic (as done in elsA).

SubandSupInflow: If M<1 at the center of the boundary’s Face use No-reflexion BC; else use Dirichlet

Join match: to deal with multiblock or periodic.

**Jacobian:**

Exact discrete linearization is made by the use of TAPENADE (Algorithmic Differentiation (AD) software) to obtain tangent product and adjoint product. Then the Jacobian could be obtained by red-black ordering (As in Mettot thesis 2013 or Beneddine thesis 2017).

**“Time” Solver:**

Explicit: Low-storage Runge-Kutta (Bogey Bailly JCP 2004)

Implicit:

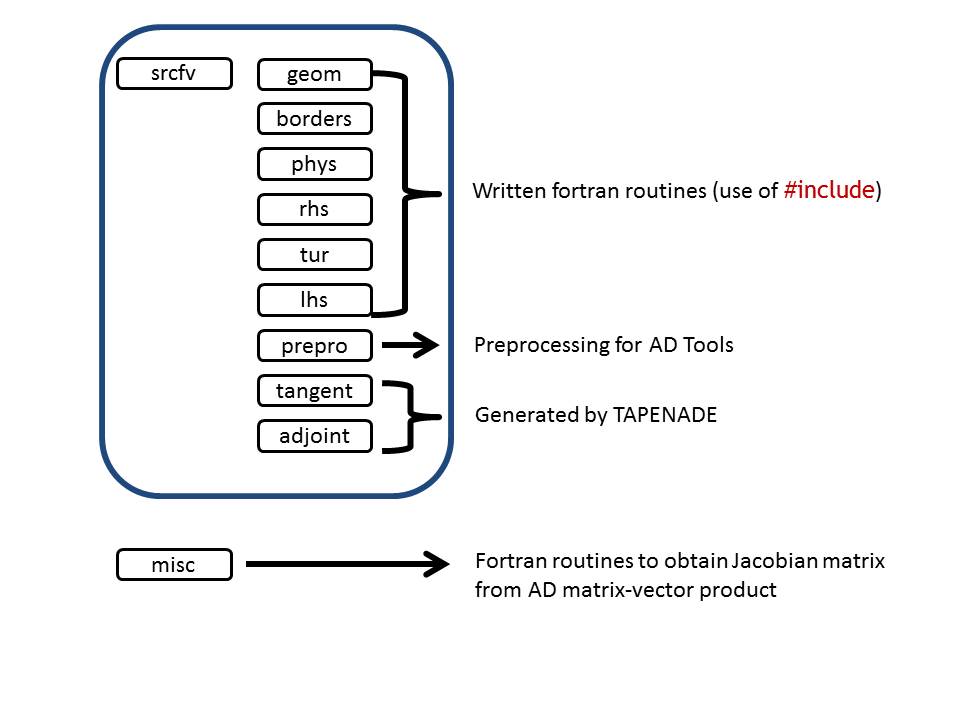
1. Backward Euler solved with Matrix Free scheme proposed by Corre (HDR 2004, similar to LU-SGS on approximated Jacobian)
2. Backward Euler in conjunction of adaptive CFL, solved by Petsc

Newton solved by Petsc

**Implementation**

BROADCAST uses a combination of a compiled and type-safe language (FORTRAN) with an interpreted and dynamically typed language (Python): this way, it benefits from the fast and safe execution of the first one, along with easy and fast development capabilities thanks to the second one. Actually, FORTRAN routines are wrapped in python via f2py.

The main part of the FV solveur is written in the srcfv package that is described hereafter. The misc package contains the formation of the Jacobian matrix, stability and resolvent functions.



**Stability and resolvent analysis functions**

*Figure 2: Main part of the code*

The code is also given with third party tools:

SIM: packages that provides compressible similarity solution for boundary layer flow (Arthur Poulain phD thesis)

meshBL.py: contains some function to stretch 1D mesh.

set\_bnd.f90: setting fields for Inflow BC and Noreflexion on the top of the domain (need to be compile with compilef90.py)

**Content of srcfv:**

**Directories:**

adjoint (AD adjoint FORTRAN files)

borders (boundary conditions FORTRAN files)

geom (metrics computation FORTRAN files)

lhs(Matrix-free Inversion based on poorly approximated Jacobian matrix-vector product)

phys (viscosity, primitives)

prepro (preprocessed FORTRAN files - needed for AD)

rhs (NS -spatial numerical schemes from geometry to residual computation FORTRAN files)

tangent (AD for linearised routines)

tur (RANS-spatial numerical schemes from geometry to residual computation FORTRAN files) : *under construction*

**Python script:** Useful to compile FORTRAN files

compile\_rhs.py

compile\_tur.py

compile\_borders.py

compile\_geom.py

tap\_adjoint.py

tap\_adjoint\_bc.py

tap\_adjoint\_jn.py

tap\_tangent.py

tap\_tangent\_bc.py

tap\_tangent\_jn.py

compile\_tangent.py

compile\_adjoint.py

compile\_norm.py

**Resulting python modules:**

f\_bnd.so

f\_geom.so

f\_sch.so

f\_lin.so

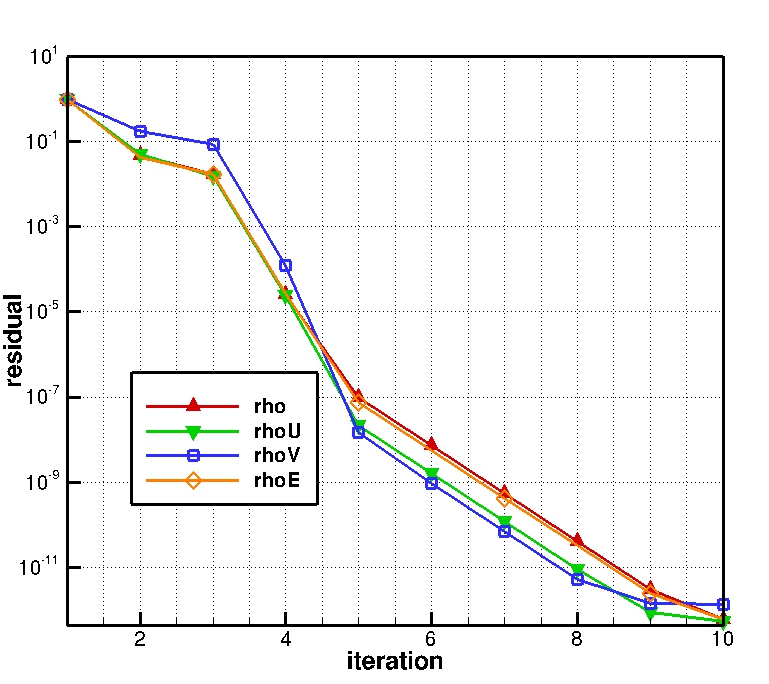
f\_adj.so

f\_norm.so

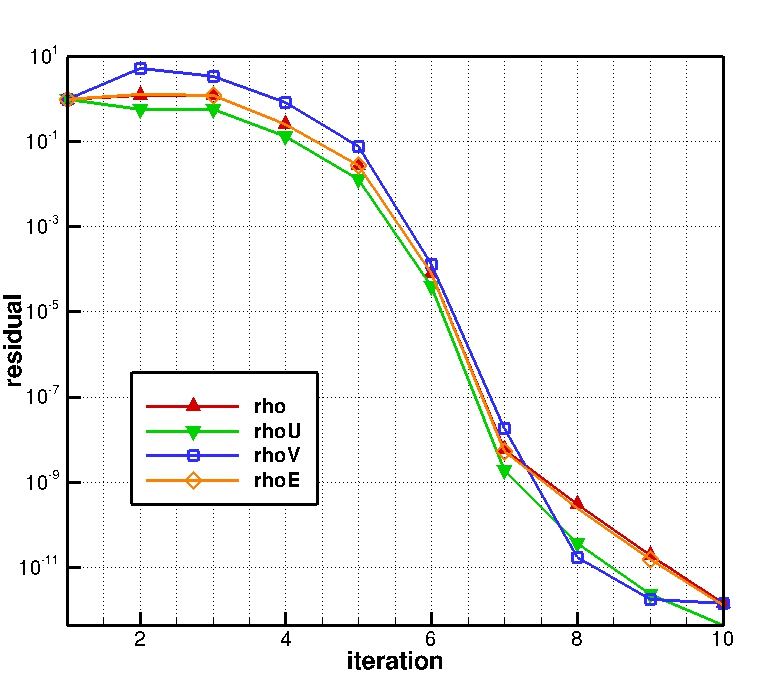
**Exemple:**

Simulation has been done by running: ***python card\_bl2d.py*** (that calls **BROADCAST.py**). (Python files are provided as exemple to use BROADCAST).

Results of Newton algorithm for Mach=4.5 , Runit= 106 , T0=65,15K, 80x60 monoblock Boundary Layer simulation solved with DNC+5 points MUSCL reconstruction (181 s) and DNC +11 points MUSCL reconstruction (470 s). (CPU time for a run from similitude solution on an Intel(R) Xeon(R) E5-2650 v4 processor)



*Fig 3: Newton - DNC 3*



*Fig 4: Newton - DNC 9*