FLUBIO-PETSC
Yet another CFD solver
– In turbulent times –

Joel Guerrero
University of Genoa + Wolf Dynamics

Edoardo Alinovi
University of Genoa

FLUBIO-PETSC

- **FLUBIO-PETSC** or **FLUBIO** for short, is an unstructured, parallel, finite-volume based Navier–Stokes and convection–diffusion like equations solver for teaching and research purposes.

What about the name of the solver? Why FLUBIO-PETSC?

- Homage to our old research group **FLUBIO** at the University of Genoa.
- The word **FLUBIO** is a portmanteau of the word FLUVIO (river in Italian) and the prefix BIO (as in **bio**mimetics, **bio**logical, **bio**inspiration).
- And it is **FLUBIO-PETSC**, because we leverage the high-performance library **PETSC**.

What is FLUBIO? – General description

- **FLUBIO** is an unstructured, parallel, finite-volume method based solver.
- **FLUBIO** can deal with 1D, 2D, and 3D domains.
- **FLUBIO** is targeted to solve the Navier-Stokes equations and convection-diffusion like equations.
- **FLUBIO** is written using modern Fortran (2003+ standard).
- **FLUBIO** is object-oriented and allows for code re-use and code extensions.
- **FLUBIO** comes with a collection of modules containing the numerical operations commonly found during the discretization of the governing equations using the Finite Volume Method (FVM).
- **FLUBIO** uses a coding standard to promote clarity and understandability. It makes wide use of comments.
- **FLUBIO** is targeted at students, academics, and practitioners willing to understand the general theory behind modern CFD solution methods and discretization techniques.
- At the same time, **FLUBIO** is general and capable enough to deal with complex problems.
- And most important, it is open-source.

What is NOT FLUBIO? – Disclaimer

- First of all, it is under development by a very small group of two people (as of March 2021); therefore, do **not** expect rapid development times.
- Many things are **not** present at the moment, but there is always a way around the problems. To name a few missing features:
    - Meshing capabilities.
    - Extended sampling capabilities.
    - Many monitors and on-the-fly post-processing.
    - Support to hanging nodes.
    - Support to post-process polyhedral cells.
- **FLUBIO** is **not** bug-free.
- **FLUBIO** is **not** a silver bullet and it may **not** be the right tool for you.

What is NOT FLUBIO? – Disclaimer

- It is **not** a GUI-based solver; therefore, it has a steep learning curve.
  - But we have plans for a GUI and a cloud-based interface.
- It is **not** a solver for the absolute beginner.
  - To start using **FLUBIO**, you need to be a little bit familiar with the FVM method.
  - You also need to have a minimal exposure to the Linux OS and JSON file format.
- **FLUBIO** is **not** a high-level programming environment, at least for the moment.
- Do **not** expect to start to do your own developments right ahead.
  - To start developing in **FLUBIO**, you need to have some FORTRAN working knowledge.
  - You will also need to get familiar with the API.
- As a research-academic software, it is very experimental, it is **not** very stable.
  - But on the positive side, we have had the opportunity to experiment with many crazy concepts.

FLUBIO under the hood

- Collocated (cell-centered) finite volume method solver.
- It uses unstructured meshes.
- It supports the following cell types:
    - Triangle and quad cells in 2D.
    - Tetrahedral, hexahedral, prismatic, and pyramid cells in 3D.
- The governing equations can be discretized on a term-by-term basis.
- Many discretization schemes available for the time derivative, gradient terms, diffusive terms and convective terms appearing in the governing equations.
- It comes with many high-resolution methods.
- Overall second-order accuracy in space and time.
- To enforce the monotonicity principle, slope limiters are implemented.

FLUBIO under the hood

- Pressure-velocity coupling is achieved via segregated solvers using the following methods,
  - **SIMPLE**, **SIMPLEC**, **PISO**, fractional-step.
- To avoid the checkerboard instability, the Rhie–Chow interpolation is used.
- It is possible to save in ASCII format the matrices assembled during the discretization process.
- Non-uniform boundary conditions, non-uniform initial conditions, and linear and non-linear source terms can be defined directly in the input files. No need for programming.
- Parallelization via domain decomposition.
- It leverages **PETSC** for the solution of the linear systems arising from the discretization of the equations.
- The solution can be visualized using paraview or any visualization tool that supports unstructured VTK data.

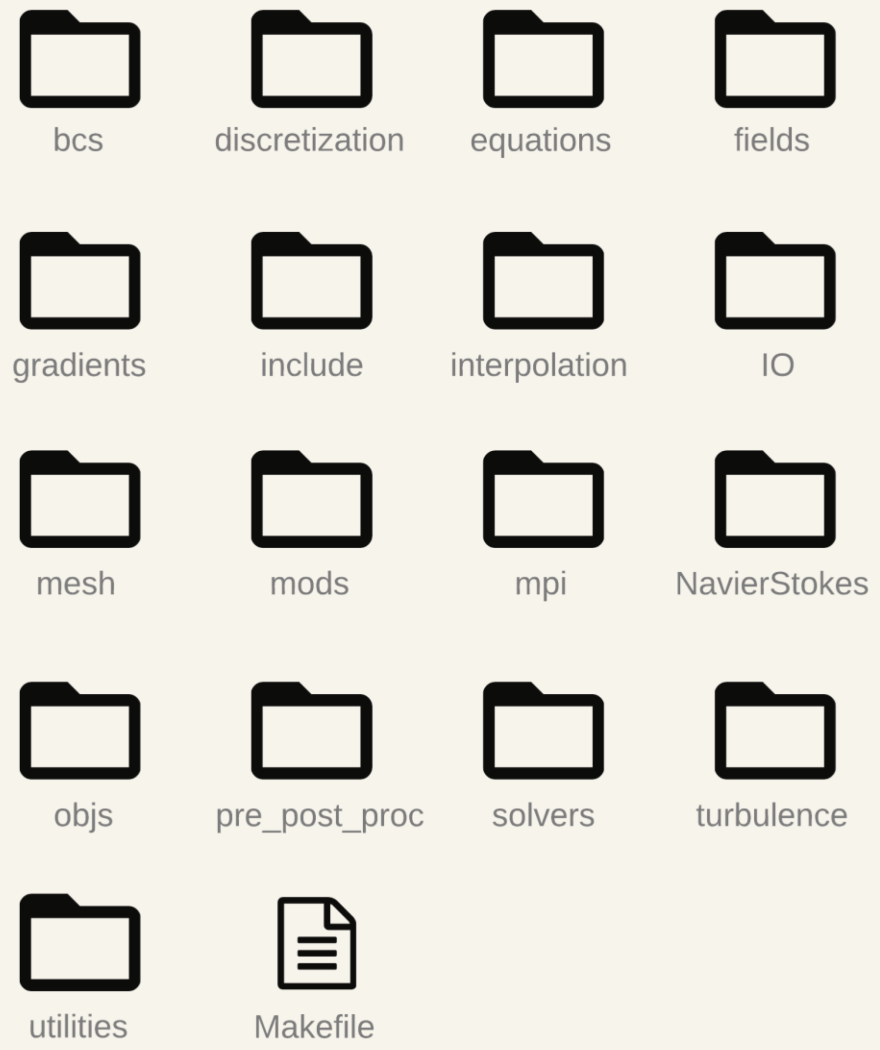
Motivation behind FLUBIO

- There is a lot of literature addressing CFD and the FVM that is distributed with basic solvers for teaching purposes [1-7].
- These solvers are getting old, do not make use of modern developments in CFD, are not supported anymore, do not support unstructured meshes, do not run in parallel, are not well documented, among many issues.
- With **FLUBIO**, we aim to address the need of many students and researchers to have a code easy to understand and to modify, but at the same time uses modern CFD developments.
- A solver that can be used to test hypotheses but still able to deal with complex geometries and industrial problems.
- We want to address the frustrations and difficulties found when using legacy solvers and solvers lacking documentation.
- We do not want to compete with well-established open-source CFD solvers. To name a few, OpenFOAM, SU2, code-saturne, MFIX, FEniCS, fluidity.
- As an open-source project, everybody is welcome to join the development. There is a lot of room for new ideas, we are open minded and open to suggestions.

[1] The Finite Volume Method in Computational Fluid Dynamics: An Advanced Introduction With OpenFOAM and Matlab. F. Moukalled, L. Mangani, M. Darwish. 2015, Springer-Verlag
[2] Finite Volume Methods for Hyperbolic Problems. R. Leveque. 2002, Cambridge University Press.
[3] Computational Gasdynamics. C. Laney. 1998, Cambridge University Press.
[4] Computational Methods for Fluid Dynamics. J. H. Ferziger, M. Peric, R. Street. 2020, Springer.
[5] Computation of Conduction and Duct Flow Heat Transfer. S. Patankar. 1991, Innovative Research, Inc.
[6] Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction. E. Toro. 2009, Springer.
[7] Computational Fluid Dynamics. Vol. 1-2. K. Hoffmann, S. Chiang. 2000, EESbooks.

Source code organization

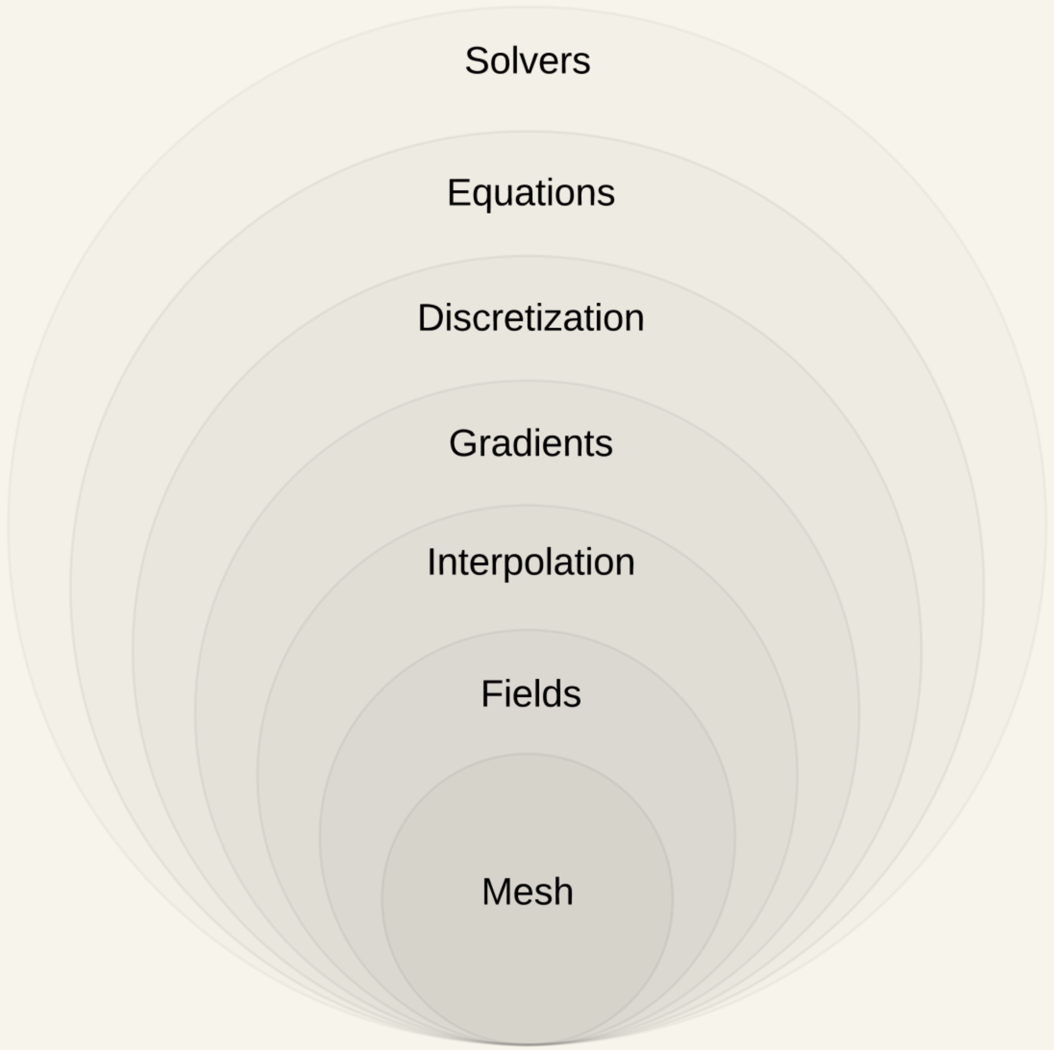
- The source code is distributed into many directories.
- Each directory containing a building block of the whole library.
- The name of each directory is representative of a step in the FVM, the physics involved, or the data structure.
- Inside each directory, you will find the source code, which has been commented to the best of our capability.
- Additionally, you will find the documentation in FORD format.
- The code is very compact, and it compiles fast.



bcs discretization equations fields
gradients include interpolation IO
mesh mods mpi NavierStokes
objs pre_post_proc solvers turbulence
utilities Makefile

Building blocks

- **FLUBIO** structure follows the steps used when discretizing the governing equations using the FVM.
- **FLUBIO** objects are wrapped into different drivers to be used when implementing solvers.
- The method and procedures used are compact and reusable.
- **FLUBIO** is extensively commented and documented to the best of our capability.
- Additions are straightforward with a good understanding of the API and FORTRAN.



Discretization options

- The governing equations can be discretized on a term-by-term basis.
- Different methods with different accuracy can be used.
- Tailored discretization methods can be added by simply following the existing templates.

Term	Brief description
Gradient	Cell-based Green–Gauss and least-squares. To enforce monotonicity and improve solution stability, gradient limiters can be used. Face-limited and cell-limited versions are available.
Transient term	Methods for iterative and time marching. Methods implemented: Steady-state, Euler, Backward differencing, Crank–Nicolson.
Diffusion term	Weighted central differences with non-orthogonal corrections. Three methods available, namely, over-relaxed, orthogonal correction, and minimum correction.
Convective term	State of the art convective schemes applicable to unstructured meshes. Many methods implemented, to name a few, upwind, central differences, second-order upwind, QUICK, minmod, vanleer, superbee.

Solvers currently implemented

Solver name	Target equation	Brief description
`flubio_poisson`	$\partial_t \phi = \nabla \cdot \Gamma \nabla \phi + f$	Transient and steady solver for the Poisson equation
`flubio_potential`	$\nabla^2 \phi = 0$	Steady state potential flow solver
`flubio_burgers`	$\partial_t \mathbf{u} + \mathbf{u} \nabla \cdot \mathbf{u} = \nabla \cdot \nu \nabla \mathbf{u}$	Transient solver for the viscous Burgers equation.
`flubio_transport`	$\partial_t \phi + \nabla \cdot \mathbf{u} \phi = \nabla \cdot \Gamma \nabla \phi + f$	Transient and steady state solver for the convection–diffusion equation
`flubio_simple`	$\nabla \cdot \mathbf{u} = 0, \quad \nabla \cdot \rho \mathbf{u}\mathbf{u} = -\nabla p + \nabla \cdot \mu \nabla \mathbf{u} + f$	Steady state, incompressible, Navier–Stokes solver using the SIMPLE algorithm
`flubio_piso`	$\nabla \cdot \mathbf{u} = 0, \quad \partial_t \rho \mathbf{u} + \nabla \cdot \rho \mathbf{u}\mathbf{u} = -\nabla p + \nabla \cdot \mu \nabla \mathbf{u} + f$	Transient, incompressible, Navier–Stokes solver using the PISO algorithm
`flubio_fs`	$\nabla \cdot \mathbf{u} = 0, \quad \partial_t \rho \mathbf{u} + \nabla \cdot \rho \mathbf{u}\mathbf{u} = -\nabla p + \nabla \cdot \mu \nabla \mathbf{u} + f$	Transient, incompressible, Navier–Stokes solver using the fractional step method by Teman & Chorin

Case organization

- In **FLUBIO**, cases are organized in stand-alone folders.
- The user must respect the directory organization illustrated in the figure.
- The input files are formatted using JSON format.

Case directory

- **BCs:** input files containing boundary and initial conditions
- **grid:** mesh in FLUBIO format
  - **processor*:** decomposed mesh (for parallel simulations)
  - **serial:** serial mesh
  - **bin:** mesh saved in binary format
- **physical:** physical constants required by the solver
- **settings:** numerical settings and simulation controls
- **postProc:** post-processed fields and monitors
  - **fields:** fields saved in binary format
  - **monitors:** monitors and sampled output
  - **VTKfields:** fields in binary XML format (*.vtu)

Input files – JSON format

- The format of the input files is very flexible.
- Input files are parsed and stored as JSON dictionaries.
- There is no need to respect the order of the entries.
- Only compulsory entries are required.
- If you forget one compulsory entry or if you misspelled an entry, **FLUBIO** will let you know what and where the problem is, and in most situations, it will give you a list of options.
- Unused keywords are allowed. They are ignored at parsing time.
- Comments can be added by inserting the exclamation mark (!) at the beginning of the line.

Sample input files

Boundary conditions:

```
“sides”:
{
   “type”: “Dirichlet”,
   “value”: [0]
),

“top”:
{
   “type”: “userDefinedDirichlet”,
   “value:: “sin(3.14159*x)”
},

“fb”:
{
   “type”: “void”,
   “value”: [0]
}

“internalField”: “0.0”
```

Controls:

```
{

! Time step controls

“TimeStep”: 1.0,

“MaximumTime”: 1000,

! I/O controls

“FieldsToKeep”: 5,

“saveEvery”: 100,

“SaveGradients”: [“T”],

! Restart from

“RestartFrom”: -1

}
```

Turbulence modeling capabilities

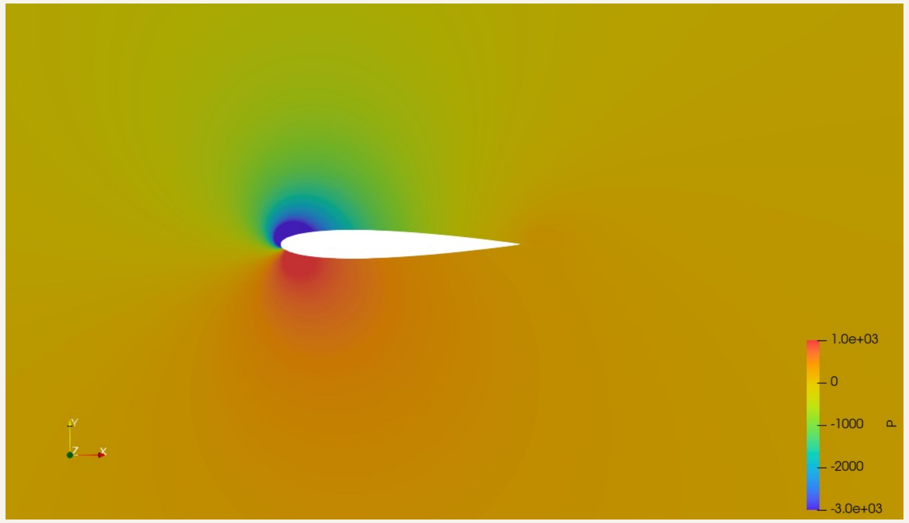
- The following **RANS/URANS** models are currently implemented:
    - Spallart-Almaras.
    - k-Omega Wilcox, k-Omega BSL, k-Omega SST.
    - y^+ insensitive wall functions are implemented for k-Omega models.
- The following **LES** models are currently implemented:
    - Smagorinsky.
    - WALE.
- And, of course, the exact Navier-Stokes equations (**DNS**).
- We are currently working hard on extending **FLUBIO** turbulence modeling capabilities.
- We are also conducting an extensive validation campaign addressing the cases presented in the Nasa Turbulence Modeling site:
    - https://turbmodels.larc.nasa.gov/

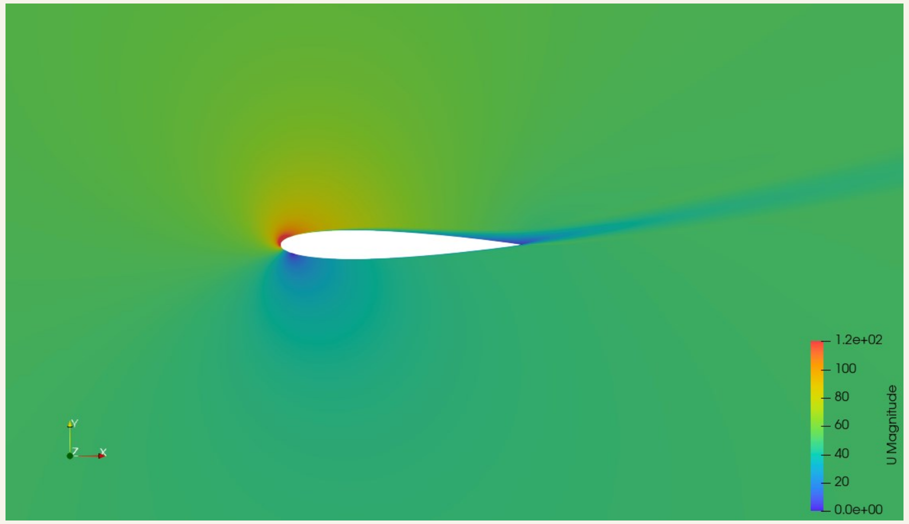
Some FLUBIO sugar

- **Turbulent flow past a NACA 0012 airfoil**
  - Re = 6 000 000
  - Angle of attack = 0°, 10°, 15°,
  - RANS simulation using Spalart-Allmaras turbulence model.
  - Wall resolving.

Term	Discretization method
Gradient terms	Cell-based Green-Gauss with slope limiters
Convective terms	Second-order upwind
Diffusive terms	Centered differences – Over-relaxed corrections

Pressure contours (top) and velocity contours (bottom) at 15°



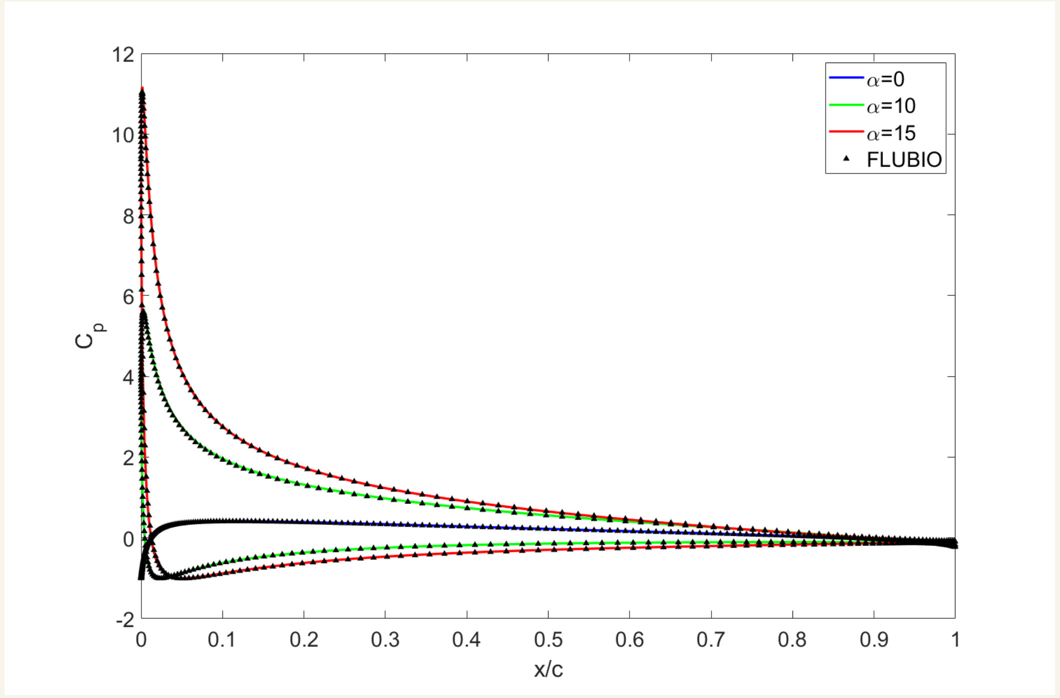


Some FLUBIO sugar

- **Turbulent flow past a NACA 0012 airfoil**
  - Re = 6 000 000
  - Angle of attack = 0°, 10°, 15°,
  - RANS simulation using Spalart-Allmaras turbulence model.
  - Wall resolving.

	0°		15°	
Solver	**C_L**	**C_D**	**C_L**	**C_D**
FLUBIO	**≈ 0**	**0.00858**	**1.5527**	**0.02190**
CFL3D	**≈ 0**	**0.00819**	**1.5461**	**0.02124**
FUN3D	**≈ 0**	**0.00812**	**1.5547**	**0.02159**
SUMB	**≈ 0**	**0.00813**	**1.5446**	**0.02141**

Term	Discretization method
Gradient terms	**Cell-based Green-Gauss with slope limiters**
Convective terms	**Second-order upwind**
Diffusive terms	**Centered differences – Over-relaxed corrections**

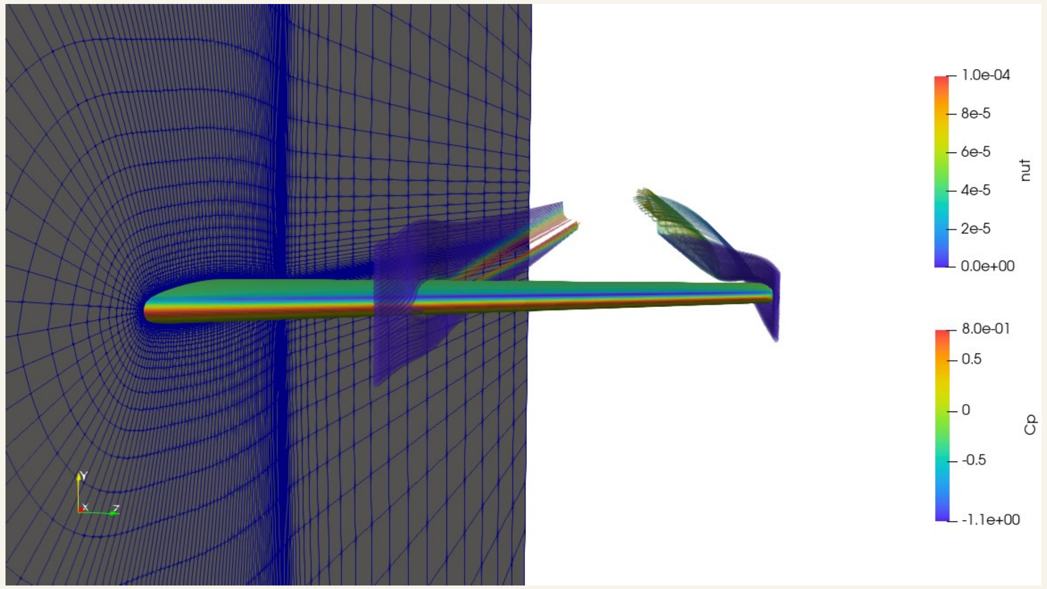


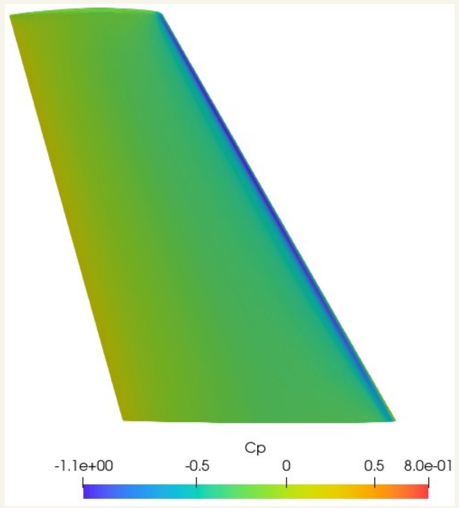
Pressure coefficient (CP) distribution – FLUBIO against CFL3D

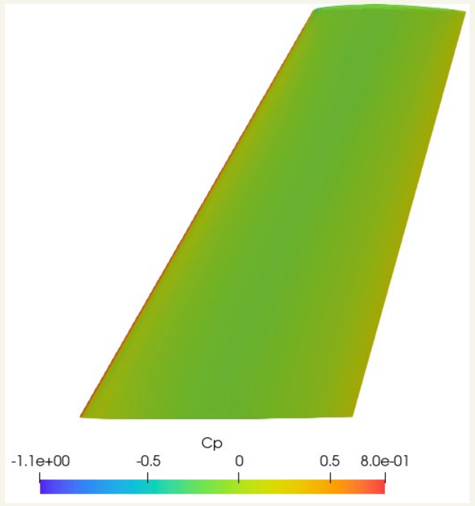
Some FLUBIO sugar

- **Turbulent flow past ONERA-M6 wing**
  - Re = 1 000 000
  - Angle of attack = 3.06°
  - RANS simulation using Spalart-Allmaras turbulence model.
  - Wall resolving.

Term	Discretization method
Gradient terms	Cell-based Green-Gauss with slope limiters
Convective terms	Second-order upwind
Diffusive terms	Centered differences – Over-relaxed corrections





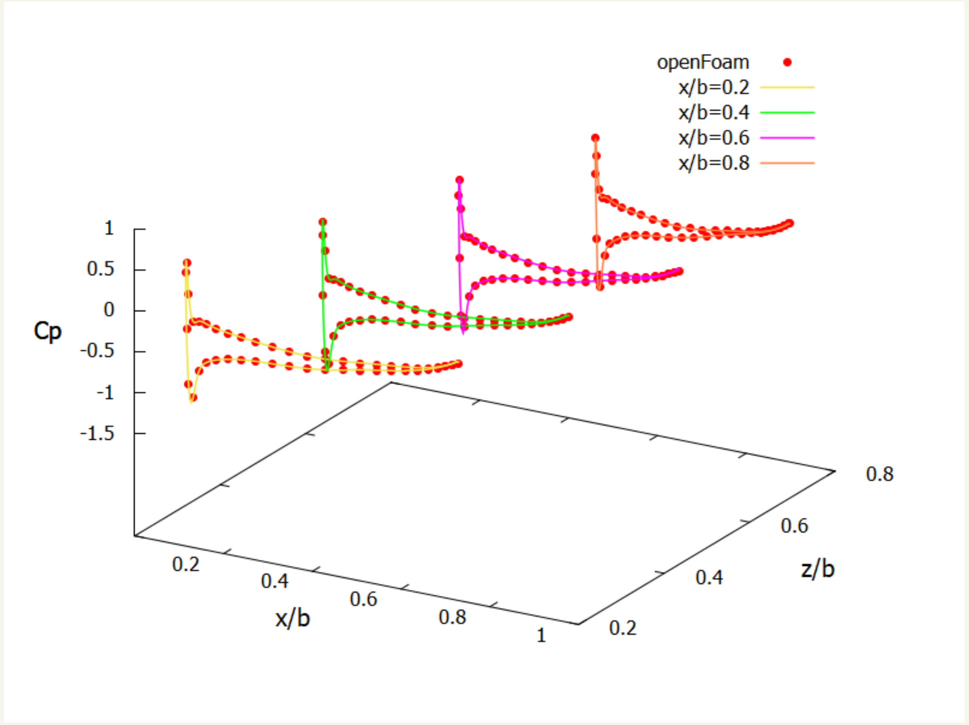


Some FLUBIO sugar

- **Turbulent flow past ONERA-M6 wing**
  - Re = 1 000 000
  - Angle of attack = 3.06°
  - RANS simulation using Spalart-Allmaras turbulence model.
  - Wall resolving.

Solver	C_D	C_L
FLUBIO	0.0067	0.0865
OpenFOAM	0.0071	0.0847
Ansys Fluent	0.0066	0.0866

Term	Discretization method
Gradient terms	Cell-based Green-Gauss with slope limiters
Convective terms	Second-order upwind
Diffusive terms	Centered differences – Over-relaxed corrections

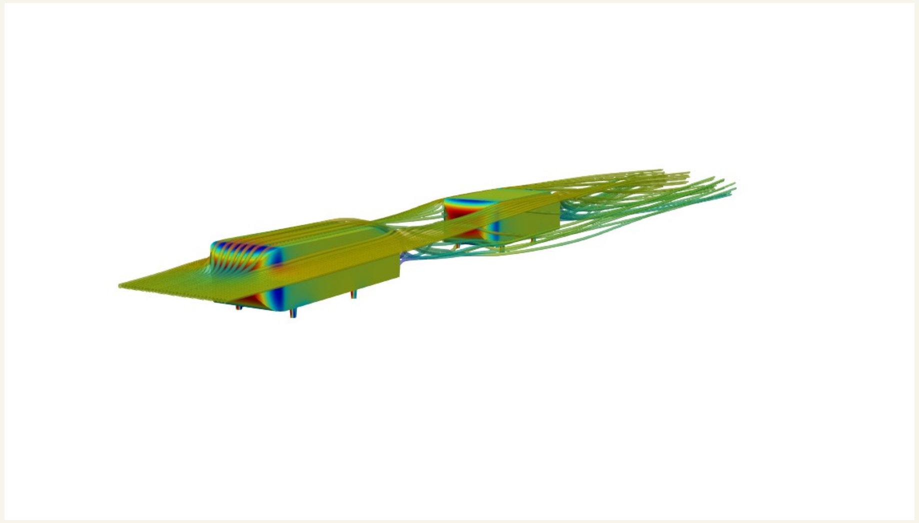


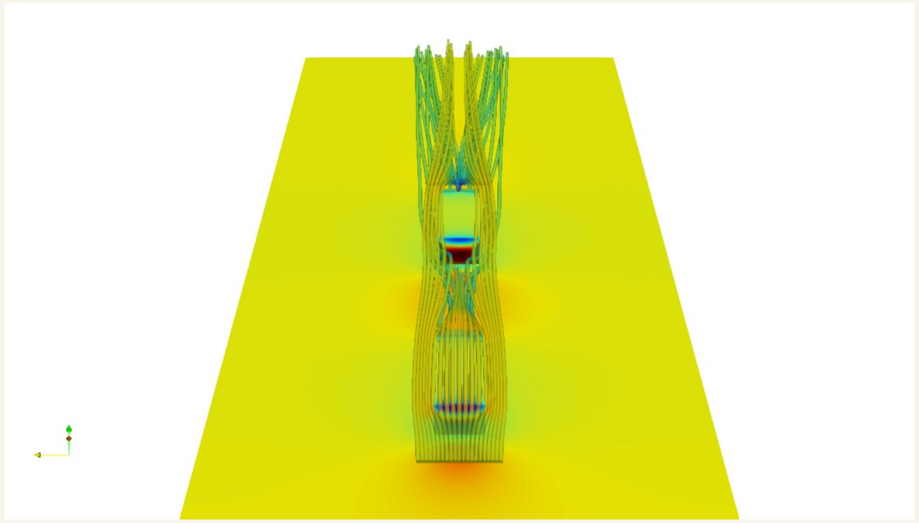
Pressure coefficient (CP) distribution at different locations

Some FLUBIO sugar

- **Two ahmed bodies in platoon formation**
  - Inlet velocity = 10 m/s
  - Slant angle = 25°
  - RANS simulation using Spalart-Allmaras turbulence model.
  - Wall resolving.

Term	Discretization method
Gradient terms	Cell-based Green-Gauss with slope limiters
Convective terms	Second-order upwind
Diffusive terms	Centered differences – Over-relaxed corrections

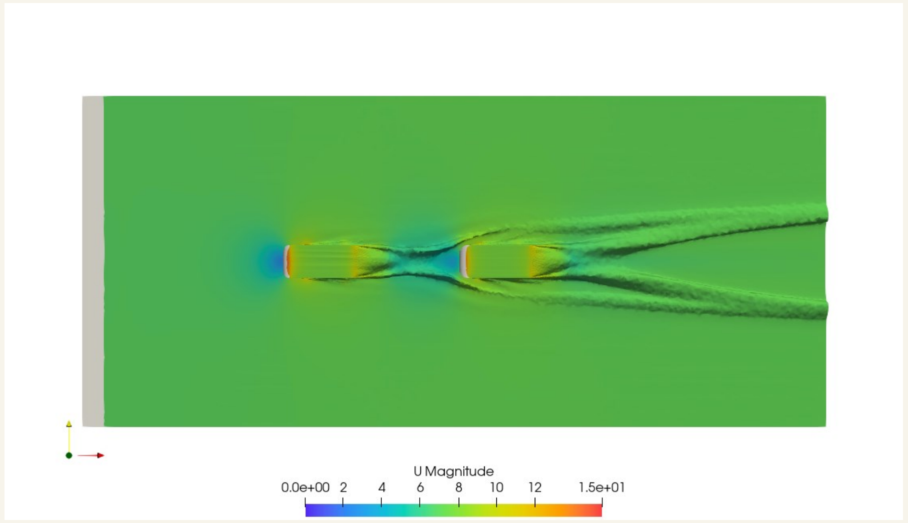


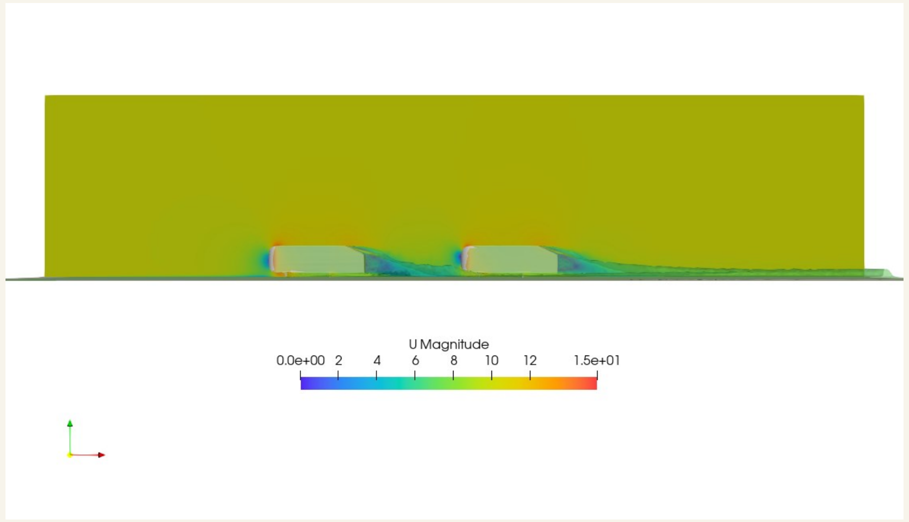


Some FLUBIO sugar

- **Two ahmed bodies in platoon formation**
  - Inlet velocity = 10 m/s
  - Slant angle = 25°
  - RANS simulation using Spalart-Allmaras turbulence model.
  - Wall resolving.

Term	Discretization method
Gradient terms	Cell-based Green-Gauss with slope limiters
Convective terms	Second-order upwind
Diffusive terms	Centered differences – Over-relaxed corrections





What is next?

- Short term goals,
    - Detailed validation campaign of the solvers implemented so far.
    - More on turbulence modeling.
    - Code profiling and code performance assessment.
    - Improvement of post-processing capabilities.
    - Removing some dependencies with OpenFOAM.
        - For the moment only meshing.
        - We just removed the mesh decomposition dependency.
    - More on documentation.
    - More video tutorials.

What is next?

- Long term plans (very ambitious),

    - Pressure-based solvers for compressible flows.
    - Coupled pressure-based solvers.
    - VOF solvers (volume of fluid).
    - Immersed boundary methods (direct forcing) for moving and elastic bodies.
    - Conjugate heat transfer capabilities.
    - Support for CGNS format.
    - Exascale computing leveraging **PETSC-HYPRE**.

Help is needed and much appreciated

- **FLUBIO** is under constant development by a very small but dedicated group.
- To reach the previous goals, we need your support.
- Everybody is welcome to contribute, and we are open to any collaboration.
- Also, we highly encourage you to break the solver. Do not forget to report the issue.

- If you want to help us in improving **FLUBIO** or you are interested in joining the development team, please send an email to,
    - flubiopetsc@gmail.com

- You can also follow us on twitter,
    - https://twitter.com/FlubioP

Where can I get FLUBIO?

- You can get the source code at the following link,
    - https://gitlab.com/alie89/flubio-code-fvm

- If you break the solver or find bugs, please report the issue using the issue tracker on the project's GitLab repository.

Where can I get more information?

- To find more information, visit **FLUBIO** website,
    - https://flubiopetsc.github.io/flubiopetsc/

- To get started with **FLUBIO** and to get further information about how to use the solver; we invite you to read the tutorial manual, which comes with the source code and is located in the examples directory.

- You can also follow the link **Getting started with FLUBIO**. At this link, you will find many video tutorials.
    - https://flubiopetsc.github.io/flubiopetsc/getting_started.html

How to cite FLUBIO?

- E. Alinovi, J. Guerrero, “FLUBIO – An unstructured, parallel, finite-volume based Navier-Stokes and convection-diffusion like equations solver for teaching and research purposes,” SoftwareX, Volume 13, January 2021, 100655. https://doi.org/10.1016/j.softx.2020.100655

- Permanent link to the publication (open access)
  - https://www.sciencedirect.com/science/article/pii/S235271102030368X

Live demonstration