AIRBUS

AIRBUS QUANTUM COMPUTING CHALLENGE

Problem Statement n°2

Computational Fluid Dynamics (CFD) on Quantum Computers

1. Problem statement

A fundamental tool in aircraft aerodynamic design are Computational Fluid Dynamics (CFD) simulations, from which the flow behavior around the aircraft and the aerodynamic forces acting its surfaces are retrieved. These calculations are computationally expensive, making the quest of highly efficient CFD algorithms a permanently open question for researchers.

This problem aims at exploring and providing some ideas about how Quantum Computing (QC) algorithms should be applied to solving CFD problems. In particular, it is requested to implement a quantum-hybrid solution in which one or some parts of an existing CFD solver (in this case SU2 [7]) are replaced by QC-based algorithms.

The case study considered in the transonic flow around a NACA 0012 airfoil. This problem is fully representative of those usually addressed in aerodynamic design and features complex flow phenomena that must be correctly predicted by the CFD solver. For the shake of simplicity, the flow will be assumed to be 2-D, inviscid and steady.

2. Problem Formulation

Under the assumptions of 2-D, inviscid and steady flow, the Navier-Stokes equation describing the motion of the fluid reduce to the so-called Euler equations. These equations are a system of partial differential equations (PDE) for which the unknowns are the density ρ, the momentum $\rho V = (\rho u, \rho v)$ and the total entalphy h_t. The equations then read:

$$\frac{\partial \rho u}{\partial x} + \frac{\partial \rho v}{\partial y} = 0 \tag{1}$$

$$\rho u \frac{\partial u}{\partial x} + \rho v \frac{\partial u}{\partial y} = -\frac{\partial P}{\partial x} \tag{2}$$

$$\rho u \frac{\partial v}{\partial x} + \rho v \frac{\partial v}{\partial y} = -\frac{\partial P}{\partial y} \tag{3}$$

$$\rho u \frac{\partial h_t}{\partial x} + \rho v \frac{\partial h_t}{\partial y} = 0 \tag{4}$$

These equations are complemented by the definition of the stagnation enthalpy and the equation of state (where $c_p = 1005 \frac{J}{Kg \cdot K}$ and $R = 287 \frac{J}{Kg \cdot K}$ are constants for the air and T stands for the temperature):

$$h_t = c_p T + \frac{1}{2}|V|^2 \tag{5}$$

$$P = \rho R T \tag{6}$$

The definition of the Mach number M is also reminded, which is a measure of the compressibility of the flow (with $\gamma = 1.4$ for the air):

$$M = \frac{|V|}{\sqrt{\gamma RT}} \tag{7}$$

This system of PDE is has to be solved along with its boundary conditions. The two fundamental types of boundary conditions that might be considered for the use case proposed here are the following:

- **Far field conditions.** Sufficiently far from the object, all flow variables should be uniform.
- **Solid wall conditions.** For the inviscid case treated here, in the walls delimiting the object the velocity component parallel to the local normal vector to the surface has to be zero. This condition means that the flow cannot penetrate the obstacle.

3. Reference CFD approach

This section provides an introduction to the standard algorithms that are implemented in CFD solvers used in industry and research.

3.1 Elements of CFD

Starting from a mathematical model, in this case the Euler equations 1-4, the following elements have to be defined [2]:

- **Numerical grid.** The grid or mesh is the discrete representation of the spatial continuous domain in which the problem is to be solved. Each spatial point considered is a node, and the volume delimited by a set of adjacent nodes defines a cell. In the case of structured grids, in which each node and cell is uniquely identified by a set of indices, (i, j) in the 2-D case. Unstructured grids are generally composed of tetrahedral elements and require a list of the connectivity which specifies the way a given set of nodes make up a cell.

- **Discretization method.** The standard approach to solve the fluid dynamic equations shown above is the Finite Volume Method (FVM). This method considers the local volume associated to each cell of the mesh and applies and integral conservation law. Such law basically states that the variation of any quantity ϕ inside a volume Ω only depends on the

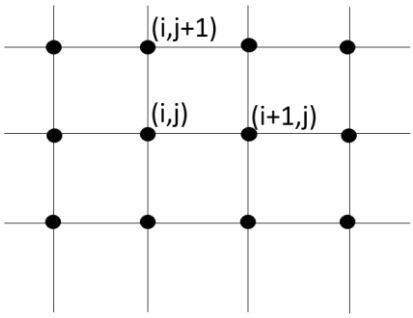


Figure 1: 2-D structured mesh with indexed nodes

fluxes F across its surfaces S. FVM allows automatically satisfying the conservative nature of the fluid mechanics equation. This equation of conservation can be writen as follows:

$$\frac{\partial}{\partial t}\int_{\Omega} \phi d\Omega + \oint_{S} F dS = 0 \tag{8}$$

- **Numerical schemes**. Following the discretization of the domain, the mathematical operators of the equations to be solved have to be discretized as well. This approximation of the equations is achieved by the numerical schemes. Two types of schemes are considered: the time integration schemes and the spatial schemes. The time integration schemes allow modeling the time evolution of the variables to be solved, while the spatial schemes represent the spatial gradients of those variables.
- **Solution method**. Once the discrete problem and the numerical schemes are set up, a system of algebraic equations has to be solved. Depending on the nature of the schemes, an iterative method might be needed. In such case, methods for solving linear systems of the form $\mathcal{A}x = b$ are used.

3.2 FVM and numerical schemes

The conservation law in equation 8 has to be first writen in the discrete domain. The time dependent term is transformed by computing a cell-averaged value of the variable, $\bar{\phi}$, that is subsequently attributed to the cell geometric center. The fluxes across the cell boundaries are approximated as the sum over the cell faces of the numerical flux, that is itself the discrete representation of the physical flux. The discrete version of the conservation law reads [3]:

$$\frac{d}{dt}\left[\bar{\phi}_{i,j}\Omega_{i,j}\right] = -\sum_{faces} F^{*}\Delta S = -R_{i,j} \tag{9}$$

The numerical flux F^{*} is determined by means of a spatial numerical scheme presented in section 3.2.1. The balance of fluxes over all the domain cells allow calculating the residual R that will subsequently be used to perform the time marching. For this purpose, the left hand side term is approximated by means of a time integration scheme, presented in section 3.2.2. In both cases a structured mesh is assumed for simplicity.

3.2.1 Spatial numerical scheme

The role of the spatial numerical scheme is to determine the numerical flux $\boldsymbol{F}^{*}$ of equation 9 at a given time instant. For this purpose, a cell-centered approach is presented here as shown in figure 2. From a set of nodes (A,B,C,D,...) forming the grid with known coordinates, the coordinates of the (i,j)-cell center, its face normals and its volume can be defined for the 2-D case as [3]:

$$x_{i,j} = \frac{1}{4}(x_A + x_B + x_C + x_D) \tag{10}$$

$$\Delta S_{i+1/2,j} = (y_B - y_A)1_x - (x_B - x_A)1_y \tag{11}$$

$$\Omega_{i,j} = \frac{1}{2}[(x_C - x_A)(y_D - y_B) - (x_D - x_B)(y_C - y_A)] \quad (12)$$

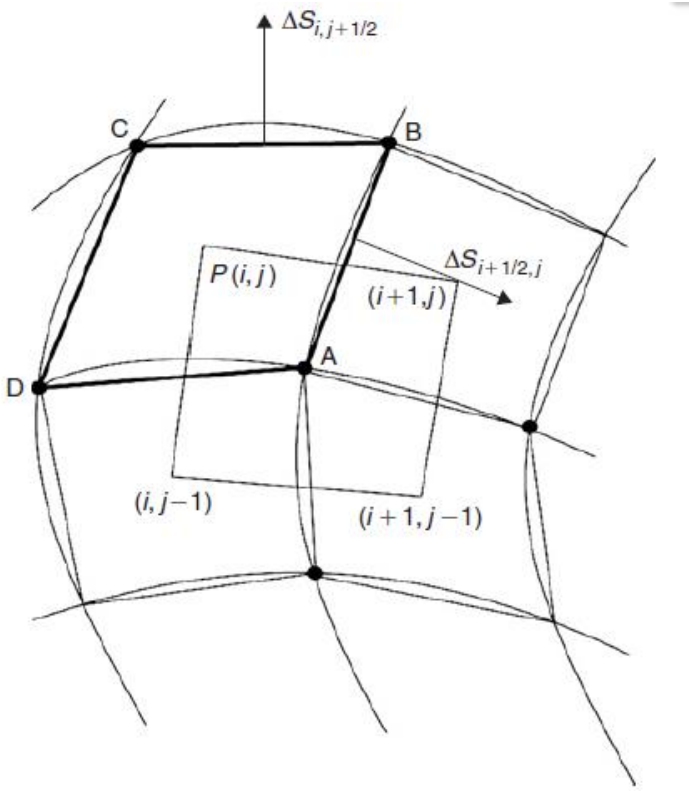


Figure 2: Cell-centered approach from [3]

Once this geometric information is available, the numerical flux can be computed. Among the wide range of numerical schemes used in CFD, one of the most frequently used is the centered second-order Jameson scheme [4]. The scheme reads:

$$\boldsymbol{F}^*_{i+1/2,j} = \frac{1}{2}(F_{i,j} + F_{i+1,j}) + \beta_{i+1/2,j}(\phi_{i+1,j} - \phi_{i,j}) - \gamma_{i+1/2,j}(\phi_{i+2,j} - 3\phi_{i+1,j} + 3\phi_{i,j} - \phi_{i-1,j}) \quad (13)$$

The first term of 13 is the centered approximation of the flux. In order to ensure the stability of the scheme, the second and third right hand side terms in 13 are added representing an artificial viscosity. These term are calculated as follows:

$$\beta_{i+1/2,j} = -\frac{1}{2}\kappa^{(2)}[v\Delta S + c\Delta|S|]_{i+1/2,j} \quad (14)$$

$$\gamma_{i+1/2,j} = \frac{1}{2}\kappa^{(4)}[v\Delta S + c\Delta|S|]_{i+1/2,j} \quad (15)$$

Where $\kappa^{(2)}$ and $\kappa^{(4)}$ stand for coefficients of dissipation that can be tuned by the user. Typical value considered are $\kappa^{(2)} = 0.5$ and $\kappa^{(4)} = 1/64$.

With these elements, for each cell the flux shall be computed for each of its faces. Then, a balance of flux can be performed for each cell so as to compute the residual appearing in the right hand side of 9. Special attention must be paid to the cells in the boundaries of the domain, that have to fulfill the prescribed boundary conditions.

3.2.2 Time integration

Even for steady problems, the steady state solution is usually searched by solving the time-dependent version of the equations so that their hyperbolic nature is kept (*pseudo-time marching*). It is therefore necessary to introduce a time integration scheme. Schemes can be classified into $implicit$ and $explicit$, depending whether the right hand side of equation 9 is evaluated on the current known time instant or the next one. To illustrate this, let us consider the first order Euler scheme in its two versions:

$$\frac{\bar{\phi}_{i,j}^{n+1} - \bar{\phi}_{i,j}^{n}}{\Delta t} = \frac{1}{\Omega_{i,j}} R_{i,j}^{n} \tag{16}$$

$$\frac{\bar{\phi}_{i,j}^{n+1} - \bar{\phi}_{i,j}^{n}}{\Delta t} = \frac{1}{\Omega_{i,j}} R_{i,j}^{n+1} \tag{17}$$

Where ϕ represents any flow variable, n is the pseudo-time step and R is the residual resulting from the flux balance. Equation 16 is the explicit formulation while equation 17 is the implicit one.

Explicit schemes (equation 16) have to respect a condition in order to be stable. This is the CFL condition, that states that $CFL \leq 1$ for the scheme to converge. On the other hand, explicit schemes are computationally cheap as the calculation of the next step solution is straightforward. Implicit schemes unconditionally stable but the computation of the next step requires to solve a linear system $\mathcal{A}\Phi = R$, where $\mathcal{A}$ is a sparse matrix resulting from the numerical schemes that has to be inverted. The inversion of this matrix is computationally expensive and requires an iterative method to be implemented.

In the particular case of steady simulation, the local time stepping technique can be used. It consists of adapting the time step Δt to the cell size and its local conditions so that the CFL number constraint is locally respected. As a consequence, the time consistency of the transients solutions is lost as each cell has its own time step, which is irrelevant since only the steady state solution is of interest. The local time is computed as follows [3]:

$$\Delta t_{i,j} \leq CFL \frac{\Omega_{i,j}}{|(v+c)_{i,j}\Delta S_i| + |(v+c)_{i,j}\Delta S_j|} \tag{18}$$

With

$$\Delta S_i = \frac{1}{2}(S_{i+1/2,j} + S_{i-1/2,j}) \tag{19}$$

$$\Delta S_j = \frac{1}{2}(S_{i,j+1/2} + S_{i,j-1/2}) \tag{20}$$

3.2.3 Initial and boundary conditions

Unless a better guess of the final solution is available (for instance the result of a previous calculation with similar boundary conditions), the initial solution is typically taken as an uniform flow over the complete domain, with the variables taking the value of those specified in the inlet boundary conditions.

The boundary conditions specify the values of the flow variables at the domain boundaries. Two of the most common boundary conditions were introduced in section 2.

3.2.4 Solution method: iterative methods

As previously stated, if the time integration method is chosen to be implicit, then an iterative method has to be implemented for time marching. Among the most widely used are the Jacobi method, LU method or GMRES method [3].

4. Quantum approach to CFD

The goal of this problem statement is to explore how quantum computing can be applied to a CFD solver by means of implementing a quantum-hybrid algorithm.
The quantum-hybrid approach consists in replacing one or several modules of a standard CFD solver (an overview of such modules was provided in section 3) by a QC-based algorithm. The CFD solver to be used is the open-source SU2 code, described in section 5.1.
The tasks to be carried out are:

1. Identify which modules of the standard CFD solvers might be replaced by a QC-based algorithm, taking into account the potential computational gains that the algorithm would bring with respect to the standard algorithm.
2. Develop such algorithm.
3. Provide a mapping of the algorithm to the number of qubits required to run it using QC hardware.
4. Provide an estimation of the evolution of the computational time required to run the algorithm when the problem size increases (i.e., the number of cells contained in the mesh).

The application case to eventually demonstrate the proposed quantum-hybrid solution is the transonic flow around the NACA 0012 airfoil, presented in section 5.2.

5. Quantum-hybrid CFD solver

5.1 SU2 solver for quantum-hybrid approach

The CFD solver for building a quantum-hybrid approach is the SU2 code [7]. SU2 is an open-source, free software written in C++ and Python for solving PDE such as the fluid dynamic equations in unstructured mesh topologies. The code structure follows a modular approach, allowing to replace parts of the code by external algorithms. The repository of the code and a detailed explanation of its structure can be found in $https://github.com/su2code/SU2$ and $https://su2code.github.io/$.

The repository also provides all the necessary input files to compute the flow around a NACA 0012 profile in transonic regime: $https://su2code.github.io/docs/Quick-Start/$. A description of this problem is presented in the following section.

5.2 Application: Transonic flow around the NACA 0012 airfoil

The NACA 0012 airfoil is a symmetric aerodynamic profile frequently used in the assessment of CFD algorithms and models. The hypothesis of 2-D, steady and inviscid flow are applied. A transonic upstream Mach number will be considered, leading to supersonic regions and the eventual presence of shock waves. The airfoil is placed in an uniform upstream flow with a non-zero angle of attack.

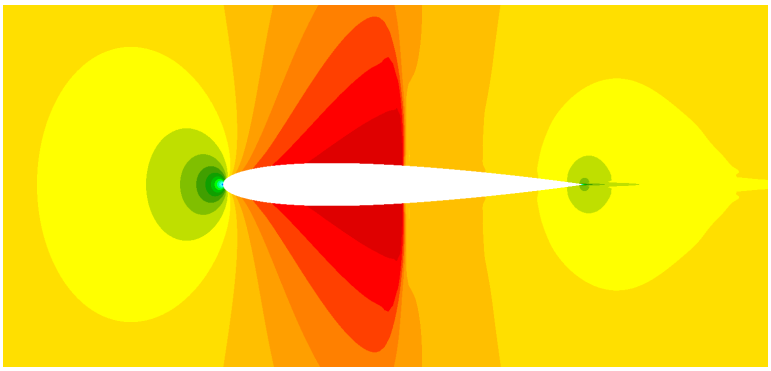
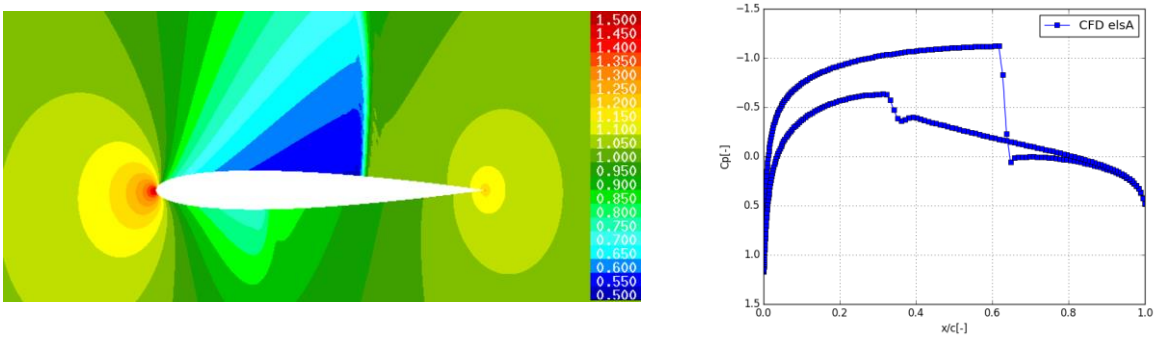


Figure 3: NACA 0012 airfoil in transonic regime

In this application we aim to solve the Euler equations considering an angle of attack of $\alpha = 1.25deg$, and freestream uniform conditions of $P_\infty = 101325Pa$, $T_\infty = 273.15K$ and $M_\infty = 0.8$ (suffix ∞ standing for freestream). The walls are modeled as inviscid.

In addition to the reference solution that can be obtained by running the SU2 code files for this case, results obtained using the CFD solver elsA [5] are provided hereafter in a mesh featuring 65000 cells. Fluxes were treated using the Jameson scheme with coefficients $\kappa^{(2)} = 0.5$ $\kappa^{(4)} = 1/64$. An implicit pseudo-time integration scheme was used with $CFL = 10$ and local time stepping, using the LU-SSOR algorithm. The

steady-stage solution was reached after 10000 iterations. Turnaround time with 4 processors in a High Performance Computing facility was around 430 seconds. Figure 6 shows the contours of non-dimensional static pressure P/P_{908} and the coefficient of pressure distribution over the airfoil surface defined as:



(b) Pressure contours, P/P_{∞} (a) Cp curve

Figure 4: Pressure distribution over the NACA 0012

$$C_p(x/c) = \frac{P(x/c) - P_{\infty}}{\frac{1}{2}\rho_{\infty}u_{\infty}^2} \quad (21)$$

With x being the chord-wise coordinate and c the chord of the airfoil.

Both the C_p curve and the pressure contours show the presence of a shock wave in the upper part of the airfoil, consisting in an abrupt change of the flow variables. In this context, CFD simulations are meant to provide the location and strength of this flow phenomenon with accuracy.

6. KPI

The assessment of the proposed algorithms will be based on the following criteria:

1. **Problem decomposition.** The choice of the SU2 solver modules that are to be replaced by QC-based algorithms.
2. **Algorithm mapping**. The QC hardware resources (number of qubits) required by the proposed QC-based algorithm for a given problem size (number of mesh cells).
3. **Algorithm scalability**. The evolution of the computational time required to run the QC-based algorithm with respect to the problem size (number of mesh cells).

The algorithm that provides the best performances based on these criteria will used to create a SU2-based quantum-hybrid solver and demonstrated in the solution of the transonic flow around the NACA 0012 profile using QC hardware.

References

[1] J.D. Anderson. *Fundamentals of aerodynamics*

[2] J.H. Ferziger & M. Peric. *Computational Methods for Fluid Dynamics*

[3] C. Hirsch . *Numerical Computation Of Internal & External Flows. Volume I: Fundamentals of Computational Fluid Dynamics*

[4] A. Jameson. *Numerical solution of the Euler equations by finite volume methods using Runge Kutta time stepping schemes*

[5] L. Cambier et al. *The Onera elsA CFD software: input from research and feedback from industry*

[6] NASA turbulence modeling resource $https://turbmodels.larc.nasa.gov/naca0012\_val.html$

[7] T.D. Economon et al. *SU2: An Open-Source Suite for Multiphysics Simulation and Design*

Read More: https://arc.aiaa.org/doi/10.2514/1.J053813