Verification of transport equations in a general purpose commercial CFD code.

Matthieu Melot[1], Bernd Nennemann[2] and Claire Deschênes[3]

[1],[3] LAMH, Mechanical Engineering Department, 1065 avenue de la médecine, Université Laval, Québec G1V 0A6, Canada

[2] Andritz Hydro Canada Inc., 6100 Transcanada Highway, Pointe-Claire, Québec H9R 1B9, Canada

E-mail: `matthieu.melot.1@ulaval.ca`

Abstract. In this paper, the Verification and Validation methodology is presented. This method aims to increase the reliability and the trust that can be placed into complex CFD simulations. The first step of this methodology, the code verification is presented in greater details. The CFD transport equations in steady state, transient and Arbitrary Eulerian Lagrangian (ALE, used for transient moving mesh) formulations in Ansys CFX are verified. It is shown that the expected spatial and temporal order of convergence are achieved for the steady state and the transient formulations. Unfortunately this is not completely the case for the ALE formulation. As for a lot of other commercial and in-house CFD codes, the temporal convergence of the velocity is limited to a first order where a second order would have been expected.

1. Introduction

Within hydraulic turbine manufacturers, CFD is taking more and more importance in the decision making during the turbine design process. For a long time, those decisions, related to efficiency for instance, have been confirmed with laboratory testing of the turbine. Unfortunately, for other types of decisions, like the ones related to the transient behaviour of the machine for example, day to day laboratory testing is rarely available. During the design phase, the answers can only come from CFD. That is why trustworthy and reliable CFD is required, and the uncertainty associated with the results should be quantified as much as possible to make informed business decisions.

In this paper an overview of the Verification and Validation (V&V) methodology will be presented. The goal of this methodology is to quantify the uncertainty associated with the CFD results. It is a three step process: the code verification, the solution verification, and finally the solution validation.

The ultimate goal of this work is to apply the V&V methodology to simulations of hydraulic turbine start-ups. This paper will concentrate on the first step of the V&V methodology which is the code verification and the various techniques associated.

2. Verification and Validation Methodology

Verification and Validation is a rigorous approach to control the quality and the uncertainties encountered during any type of physical simulation and in CFD simulations in particular. This

Content from this work may be used under the terms of the Creative Commons Attribution 3.0 licence. Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.
Published under licence by IOP Publishing Ltd

methodology has been standardized by ASME [1, 2] or AIAA [3] and two recent textbooks exist [4, 5].

The V&V methodology prescribes three principal activities shown in figure 1 that have to be followed: code verification, solution verification and solution validation. These activities, are applied to one or several "realities of interest". In the case of hydraulic turbine start-ups this reality would correspond to the highest stress encountered in such operation. For the three V&V steps, the boxes represent the elements used or obtained. The listed elements are the main decisions or findings that a CFD practitioner has to make.

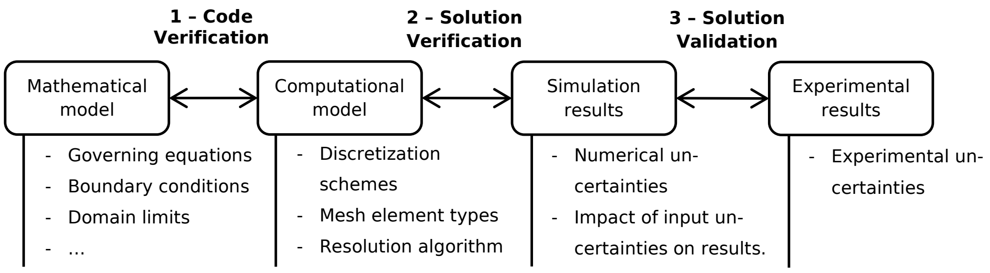

Figure 1. Verification and validation process

The first step, the code verification is presented in greater details in the next section. The two other steps, the solution verification and solution validation, which involve comparisons with experimental measurements will be covered in future work. The main foreseen test case will be the BulbT project [6] for which a start-up measurement campaign is taking place. Preliminary work for the solution verification of the best efficiency point BultT operation has already been presented by Magnal et al. [7].

3. Code verification

By performing a simulation, a mathematical model, is necessarily defined. The physical processes at play (example: turbulent flow, cavitation, vibration, etc.) and their governing equations are identified and chosen. The domain limits and the boundary conditions participate also in the definition of this mathematical model. This step requires a lot of engineering judgement as it supposes to understand which models may be pertinent and which are probably not (or less important) for obtaining a response with a sufficient precision.

In the majority of the cases, the mathematical model is solved numerically in a simulation code. A series of decisions must be made at this stage, for example, the governing equations, which are typically Partial Differential Equations (PDE) must be discretised. For each term of those equations, the simulation practitioner has to select the discretisation scheme to use: 1st order backward difference scheme, second order centered difference, to name a few. Other decisions can be related to the mesh elements types (hexahedral, tertrahedral), the resolution algorithms (Simple, Simplec, coupled, etc. in the case of CFD codes), the boundary condition interpolations etc.

With this discretised mathematical model clearly defined, the first step of the V&V methodology, the code verification assessment, can be conducted. It consists in verifying that the numerical solutions of the equations are actually solving the analytical mathematical equations properly. In other words: are the mathematical equations solved right (free of bugs) by the code? More precisely, with the refinement of the mesh, the discretisation error must converge towards zero with a certain order of accuracy. This order of accuracy must be checked, as it might reveal implementation mistakes, and compared with the order which is documented. For example, for a two dimensional case, when a second order differentiating scheme is used, the error must be divided by four when the mesh size is divided by two. If this is verified, then there is a strong indication that the code works as intended.

Obviously, to compute the discretisation error, the numerical solution must be compared with an exact analytical solution that should be chosen such that all terms in the equations are "exercised". For the Navier-Stokes equations, or for any other complicated non-linear equations, this is a difficult task, not to say impossible. The classical Navier-Stokes solutions only exist for cases where a lot of simplifications (laminar, symmetries, two dimensional, etc.) to the equations are made like in the Poiseuille or Couette flows, and consequently they do not exercise all terms.

A general methodology called method of manufactured solutions (MMS) has been proposed to circumvent the issue. As a physically realistic solution is not needed, the code verification being purely a mathematical assessment activity, not a physical one, one can suppose an arbitrary and analytic solution field. Then, with a mathematical derivation, where symbolic mathematical systems like SymPy [8] can help, the unknowns are replaced in the differential terms with the assumed solution. The obtained result, which will be different than zero, corresponds to a source term field for the original governing equations and a set of boundary conditions. This source term field will have a complicated but analytical expression and can be set in the simulation code. With this setup the simulation is launched. As the exact solution is known, it is possible to compute an error with an appropriate norm (typically L_2 norm) which is the difference between the numerical result and the exact solution.

The work of the code verification consists then in verifying that the errors converge towards 0 at the specified order of convergence when meshes are refined. While the basic idea for the code verification is surprisingly simple it is faced with a lot of practical difficulties, especially when the various terms of the equations are discretised with different orders.

4. Code verification test case

A good code verification test case must exercise the various models that will be used in the targeted application. In the present work, the goal would be to determine the performance of the moving mesh feature of the code Ansys CFX 16.2 [9]. Indeed during a turbine start-up, the guide vanes are rotated from a fully closed position to a specified opening angle during a transient computation. Also in order to simplify and automate the mesh generation process at a very small opening, meshes composed of prism elements will be used.

With those considerations in mind, the validation test case is presented in figure 2. In order to limit the computational effort without loss of generality, a simple 2D case will be treated instead of a full 3D case. The domain is a square of 1 m sides which is meshed with triangles. Because truly 2D computations are not possible with CFX, the 2D mesh is extruded in the Z direction to obtain one layer of prismatic elements and a symmetry boundary conditions is imposed on both faces normal to Z. The governing equations are laminar Navier-Stokes, and thus the transport equation models will be verified.

$$\begin{aligned} \nabla \cdot \boldsymbol{u} &= 0 \\ \rho \frac{\partial \boldsymbol{u}}{\partial t} + \rho \boldsymbol{u} \cdot \nabla \boldsymbol{u} &= -\nabla p + \nabla \cdot \left[\mu \left(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T \right) \right] + \boldsymbol{f} \end{aligned} \tag{1}$$

Where $\boldsymbol{f}$ are the body forces, p is the pressure, t is the time, $\boldsymbol{u}$ is the water velocity with u and v the components along x and y coordinates respectively, μ is the water dynamic viscosity, and ρ is the water density.

In the present work, steady state and transient versions of the Navier-Stokes equations are verified. Moreover a third stage of complexity is studied: the moving mesh within the domain and thus the ALE (Arbitrary Lagrangian Eulerian) formulation of the Navier-Stokes equations will be analysed. This formulation has been already explored by Hay et al. [10] with a finite element in-house CFD code.

It should be noticed that the classical two-equations turbulence model like k-ϵ, k-ω or SST rely on the same transport equations as the Navier-Stokes equations Unfortunately, as explained by [11], the presence of limiter functions (like min or max) in those turbulent transport equations impairs their derivability, and makes the MMS approach harder to use: the manufactured solution has to stay on “one side” of the limiter function to maintain the derivability. One way to achieve it is to impose a solution which is close to some realistic physical solutions. The situation is even worse if wall functions are used. That is why it has been decided to assume that the turbulent equations are already verified.

4.1. Manufactured solution
The manufactured solution should be as smooth as possible so that it does not pose numerical difficulties. Besides, to help even more the numerical iterative convergence of Navier-Stokes equations (1), the material properties as well as the solution should produce terms that are in the order of magnitude of 1. By doing so, floating point computation rounding errors, that would happen more drastically if terms of different order of magnitude were added together, are kept to a minimum. With all these criteria, the following solution is proposed:

$$\begin{aligned} p &= (x-0.3)^2 + (y-0.2)^2 - xy & \mu &= 1\text{kg}/(\text{m}\cdot\text{s}) \\ u &= \frac{1}{\mu}\left((x-1.2) + (x-0.5)\,y + 0.5\right) & \rho &= 1\text{kg}/\text{m}^3 \\ v &= \frac{1}{\mu}\left(-(x-1.2)^2/4 + (y-0.2)^2 - xy\right) \end{aligned} \tag{2}$$

This solution is composed of polynomial type equations. All the terms of the Navier-Stokes equations will be exercised with non trivial gradients. The pressure and velocity fields are shown in figure 3, 4 and 5 respectively.

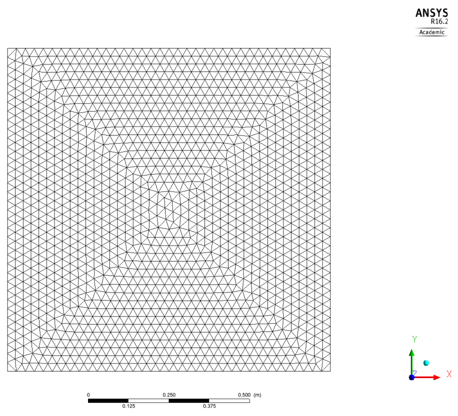

Figure 2. Domain and mesh example (3028 nodes) for code verification test case.

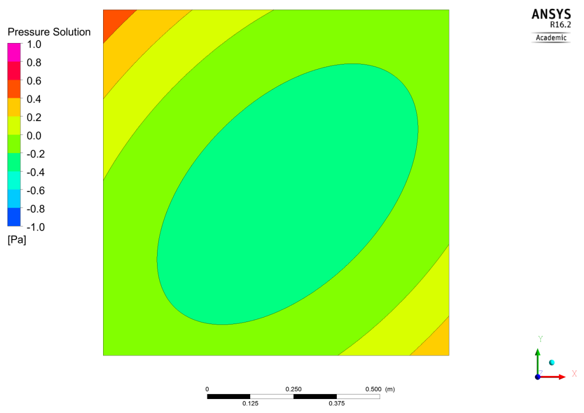

Figure 3. Manufactured pressure field

The momentum equations source terms are easily obtained with SymPy. This manufactured solution verifies the continuity equation for incompressible flows, and thus no source term is required for the continuity equation:

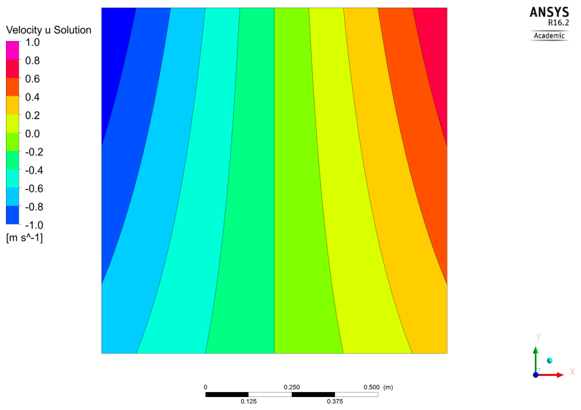

Figure 4. Manufactured velocity u

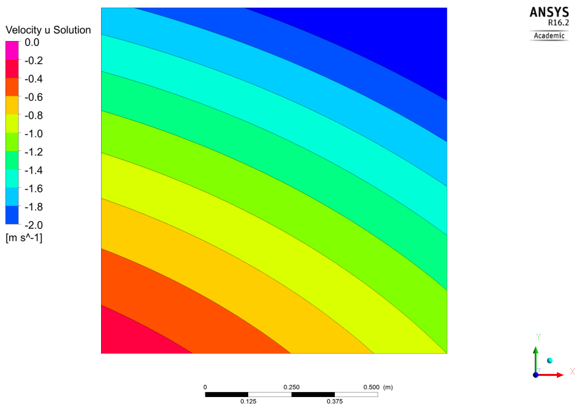

Figure 5. Manufactured velocity v

$$\frac{\partial u}{\partial x} = \frac{1}{\mu}(y+1), \quad \frac{\partial v}{\partial y} = -\frac{1}{\mu}(y+1) \quad \Rightarrow \nabla \cdot \mathbf{u} = 0 \tag{3}$$

4.2. Numerical solution and error analysis

A numerical setup is created with CFX. A special fluid material is created as explained in the previous subsection. At the domain boundaries, on the four sides of the square, the velocity vector corresponding to the manufactured solution is imposed. The solution has been derived such that three sides of the square domain, namely the right, the left and the bottom are defined as outlet boundary conditions while the top boundary condition is an inlet. Additionally, as per the method of manufactured solution, a general momentum source is added to the simulation domain to balance the manufactured solution. CFX expressions corresponding to the source term equations are used.

Great care has to be taken when choosing the numerical schemes of the equations as they will have a direct influence on the verification results. The advection term is set to a second order scheme with a specified blend factor of 1. Thus the velocity is expected to converge with an order 2. The case of the pressure is more vague. The CFX documentation [9] does not state clearly the expected order of convergence for the pressure gradient term. In order to have the highest possible order for the pressure, a tri-linear interpolation scheme between the control volume faces and the integration points is used. This scheme suggests a second order accurate interpolation while the default would normally be a simple linear interpolation and thus implying a first order scheme.

The simulation is run for a series of meshes presented in table 1. All the computations are made with double precision and are converged until residuals are close to machine zero (below 10^{-14} Max). For the three variables of interest (p, u and v) symbolised with g, an L_2 error norm is computed:

$$\|\epsilon_h\| = \sqrt{\frac{1}{N}\left(\sum_{i=1}^{N}(g_{CFD} - g_{exact})^2\right)} \tag{4}$$

The truncation error comes from the remainder of a Taylor series development of a variable g when the grid spacing $h \to 0$ and with a convergence order p:

$$\|\epsilon_h\| = g_p h^p + O(h^{p+1}) \tag{5}$$

Table 1. Mesh sizes

Mesh	Nodes	Mesh	Nodes	Mesh	Nodes
1	1498	4	12096	7	96210
2	3028	5	24052	8	192418
3	6010	6	48104	9	384842

For a systematic grid refinement in all dimensions, a grid refinement factor can be defined as:

$$r = \left(\frac{N_1}{N_2}\right)^{\frac{1}{d}} \tag{6}$$

where N_1 is the number of nodes in the fine mesh, N_2 is the number of nodes in the coarse mesh and d is the number of dimensions of the meshes. Then the coarse grid spacing can be expressed as function of the fine grid spacing $h_c = rh_f$. By rearranging equation (5) with (6) and neglecting the remainder of the Taylor series expansion, the observed order of convergence can be computed:

$$\hat{p} = \frac{\ln\left(\frac{\|\epsilon_{rh}\|}{\|\epsilon_h\|}\right)}{\ln(r)} \tag{7}$$

Where $\|\epsilon_{rh}\|$ is the error made with the coarse mesh while $\|\epsilon_h\|$ is the error made with the fine mesh case and r is the grid refinement factor.

5. Results

5.1. Steady state case

Errors L_2 norm convergence are presented in figure 6 in a log-log plot and their corresponding computed observed order are shown in figure 7. Two interesting observations can be made.

First the convergence is perfectly monotonous, i.e. there are no jumps or changes of slope. This fact is confirmed by the observed order of convergence plot where, for the three variables of interest, an almost perfect straight line is shown, albeit with some wiggles around meshes of 10,000 nodes for the v velocity component. This monotonous and stable convergence is a strong indication that the transport equations perform as expected in terms of convergence and robustness, even for the prism meshes that were a source of difficulties for some commercial codes about ten years ago [12].

Second, the pressure convergence shows a perfect order one while the observed order for the the velocity indicates a second order scheme as expected. Even if the interpolation scheme for the pressure has been selected to be tri-linear, and thus suggesting a second order accurate scheme but without clearly stating it in the documentation, something else in the pressure formulation must be only first order accurate. After raising the question to the Ansys technical support, it seems that, indeed, the pressure diffusion, in the Rhie and Chow coupling method of the momentum and continuity equations, is only first order accurate by default. The goal of this exercise is not to use the highest order of convergence but instead verifying that the code works as expected. Because after some investigation, it was figured out that a convergence order of one for the pressure is indeed expected by default, the verification is considered to be a success.

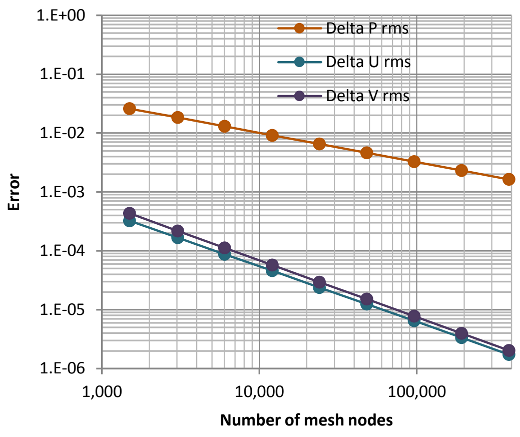

Figure 6. Truncation error convergence

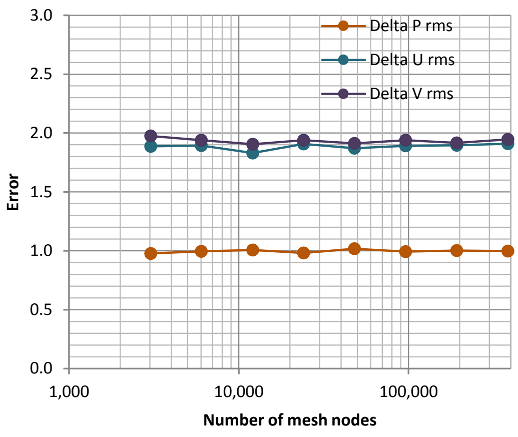

Figure 7. Observed order of convergence

5.2. Transient case

For the transient case, a new manufactured solution is needed. The steady state manufactured solution (2) has been used as a starting point. The expressions for u, v and p were multiplied by a pulsating function g that depends only on time t:

$$g = (1 - \cos(2\pi t))/2 \tag{8}$$

The rest of the methodology is the same as the steady state case, only with more complicated expressions for source terms and boundary conditions. The pulsating function stays always positive to make sure that the boundary conditions can be kept as purely inlet or outlet.

In the unsteady formulation of the Navier-Stokes equations, there are two independent meshes, a spatial 2D mesh and a temporal 1D "mesh" that both have an influence on the results of u, v and p variables. To make a comparison between the numerical results and the exact solution, the time $t = 1.1$ s has been arbitrarily chosen. For each spatial mesh presented in table 1, seven time discretisation levels have been used, from 11 time steps up to 704 time steps for a complete simulation. Each level of discretisation has twice the number of time steps than the previous level.

The time discretisation scheme chosen is the classical second order backward Euler scheme. Moreover, to limit the numerical rounding error, which can accumulate during the course of the simulation, the inner loops residuals are converged to almost zero machine before going to the next time step.

The error analysis for this case where two discretisations are present and that potentially have mixed convergence orders is slightly more complex than the steady state case. The equation (5) can be generalised to take into account the space h_x and time h_t discretisation:

$$\|\epsilon^{h_t}_{h_x}\| = g_x h_x^p + g_t h_t^q + O(h_x^{p+1}) + O(h_t^{q+1}) \tag{9}$$

In the previous equation, if the remainder terms are omitted, the spatial grid spacing h_x is kept constant, and three spacing levels for the time discretisation h_t, $r_t h_t$, $r_t^2 h_t$ are used, the term $g_x h_x^{\hat{p}}$ will be constant and can be eliminated. The observed order of convergence for the time discretisation is:

$$\hat{q} = \frac{\ln\left(\frac{\|\epsilon^{h_x}_{r_t^2 h_t}\| - \|\epsilon^{h_x}_{r_t h_t}\|}{\|\epsilon^{h_x}_{r_t h_t}\| - \|\epsilon^{h_x}_{h_t}\|}\right)}{\ln(r_t)} \tag{10}$$

The results of this error analysis, with the finest mesh of 384,842 nodes (mesh 9 in table 1) is presented in figure 8. In the subfigure 8a, pressure and velocity errors are plotted with respect to the number of time steps. In the range of times steps between 22 and 176, which corresponds to time discretisation levels 2 to 5 (included), u, v, p show an almost monotonous and linear convergence. The slope of the curves are almost parallel, which suggest that the time discretisation order of convergence is the same for all variables. Naturally the magnitude of the error is different between the pressure and the velocity, but that is not a problem for this analysis, only the slope is what matters. At both ends of the curves one can notice interesting features.

First for the time discretisation with 11 time steps, a strong slope discontinuity can be observed both for velocity and pressure convergence. For the velocity the error is even lower than for 22 time steps. This slope discontinuity indicates that the time discretisation is too coarse to properly solve the equations. The solution is not in the so-called asymptotic range where the Taylor series expansion of the numerical errors is valid. The lower error for velocity can be explained by pure chance: the Taylor series remainder in equation (9) must be probably very high, and the higher order time terms are partially cancelled with the spatial remainder terms which overall produce a lower error.

Second, at the highest time discretisation levels, again, there is a change of slope. At least the last two time discretisation levels for the pressure and the u component of velocity are not in the asymptotic range anymore. In that range, the time discretisation refinement gain is “burried” in the space discretisation errors.

In subfigure 8b, the time order of convergence computed with equation (10) is shown. For one computed order of convergence, three consecutive time discretisation levels are needed. Those levels used are indicated on the figure x axis labels.

The three first time discretisation levels do not have a observed order. Due to the fact that in this range the solution is not in the asymptotic region, equation (10) is undefined: the logarithmic function in the numerator has a negative parameter. For the finest time discretisation levels 5 to 7, the computed order are not valid because as for the coarsest one, those levels are not in the asymptotic region.

For the intermediate time discretisation levels that are well in the asymptotic region, namely levels 3 to 5 group and levels 4 to 6 group, the observed order is almost equal to 2 for the three variables of interest. This corresponds to the expected order. Because the observed order of convergence for discretisation levels that are within the asymptotic region show the expected convergence order, one can safely say that the time discretisation of CFX is verified.

5.3. Moving meshes

The final test case is essentially the same as the one presented in the previous subsection on the transient computations. The main difference is that all inner domain mesh nodes are forced to slightly move along a prescribed, and non trivial, trajectory. Thus all control volumes undergo a shape and size change at each time step which exercise thoroughly the Arbitrary Lagrangian Eulerian (ALE) formulation [9] of the Navier-Stokes equations. To keep the case simple enough and close to the previous one, the domain boundaries nodes stay in place during the whole computation.

The imposed trajectory, $\xi(t)$ and $\eta(t)$ with respect to time, along the x and y axis respectively of the mesh nodes are:

$$\begin{aligned} \xi(t) &= (t - \tanh(t))xy(x-1) \\ \eta(t) &= (t - \tanh(t))xy(y-1) \end{aligned} \tag{11}$$

The results for the time convergence are presented in figure 9, for the same mesh as the previous case. For that test case, results are slightly harder to interpret. On the positive side,

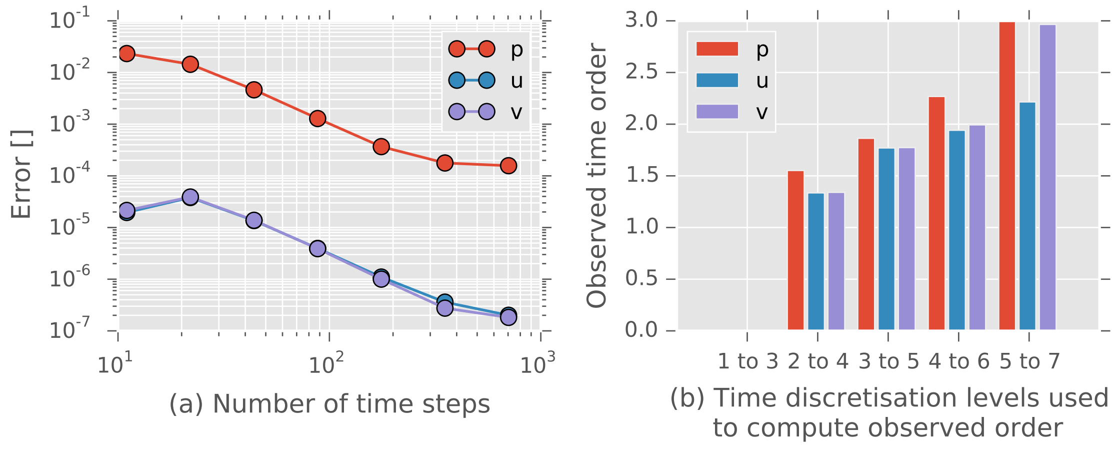

Figure 8. Order of convergence analysis for time discretisation

the convergence is monotonous and the error is always decreasing. On the more negative side, on subfigure (a), the asymptotic region is harder to identify properly. Based on the computed observed order of convergence shown in subfigure (b), it seems that the time convergence of the pressure tends to follow a second order while the velocity temporal convergence follows only a first order.

This observation is compliant with the work of Étienne et al. [13] on the time integrators in the context of an ALE formulation. They noted that a lot of codes showed the same behavior where the optimal temporal order of convergence was achieved on fixed grids but dropped to a first order for moving meshes. In their mathematically involved work, which is well outside of present work scope, they proposed an alternative formulation for the ALE which would permit, according to their claim, an ideal temporal convergence.

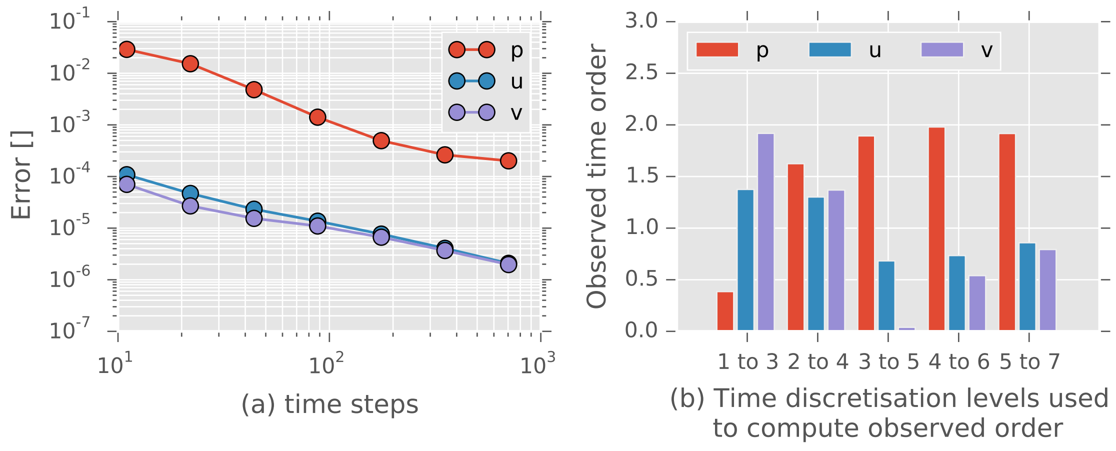

Figure 9. Order of convergence analysis for time discretisation with moving mesh

The conclusion for this ALE formulation verification is only a partial success. Indeed, one could expect that the velocity would maintain a second order temporal convergence. Unfortunately this is not the case, only a first order is achieved. To attenuate this criticism, formal analysis of the expected ALE order of convergence is not a trivial task. Moreover, a lot of CFD codes, commercial or in-house behave along the same lines.

The expected impact of this first order of temporal convergence of the velocity on more realistic CFD computations, like a turning guide vane for instance, will be to have a more diffusive behavior of the equations. This fact will have to be kept in mind when results will be interpreted from those simulations.

6. Conclusion

In this paper, the Verification and Validation methodology has been presented. This method aims to increase the reliability and the trust that can be placed into complex CFD simulations. It is a three step process: code verification, solution verification and solution validation. The first step has been detailed in greater details throughout this work.

The goal of the code verification activities was to demonstrate that the simulation code, Ansys CFX in this case, works as advertised. The present work showed that this is the case for the steady state and transient formulations of the Navier-Stokes equations. Unfortunately this is not completely the case for the ALE formulation which is used for the moving mesh capability. The temporal convergence of the velocity is limited to a first order where a second order would have been expected. It does not prevent to make good simulations when moving mesh features are used, but it is probably a good thing that the CFD practitioner be aware of this limitation.

References

[1] ASME 2006 ASME V&V 10-2006, Guide for verification and validation in computational solid mechanics
[2] ASME 2009 ASME V&V 20-2009, Standard for Verification and Validation in Computational Fluid Dynamics and Heat Transfer
[3] AIAA 1998 Guide for the verification and validation of computational fluid dynamics simulations
[4] Roache P J 2009 *Fundamentals of Verification and Validation* (Hermosa Publishers) ISBN 9780913478127
[5] Oberkampf W L and Roy C J 2010 *Verification and validation in scientific computing* (Cambridge University Press)
[6] Deschênes C, Houde S, Aeschlimann V, Fraser R and Ciocan G D 2014 *IOP Conference Series: Earth and Environmental Science* **22**
[7] Magnan R, Cupillard S, Gauthier G, Giroux A M, Page M and Deschênes C 2014 *IOP Conference Series: Earth and Environmental Science* **22**
[8] SymPy Development Team 2014 SymPy: Python library for symbolic mathematics URL `http://www.sympy.org`
[9] Ansys Inc 2015 ANSYS CFX-Solver Theory Guide Release 16.2
[10] Hay A, Yu K R, Étienne S, Garon A and Pelletier D 2014 *Computers and Fluids* **100** 204–217 ISSN 00457930
[11] Eça L, Hoekstra M, Hay A and Pelletier D 2007 *International Journal for Numerical Methods in Fluids* **54** 119–154
[12] Abanto J, Pelletier D, Garon A and Trepanier J Y 2005 *43 rd AIAA Aerospace* 1–33
[13] Étienne S, Garon A and Pelletier D 2009 *Journal of Computational Physics* **228** 2313–2333 ISSN 00219991 URL `http://dx.doi.org/10.1016/j.jcp.2008.11.032`