

# Numerical study of rotor-stator interactions in a hydraulic turbine with Foam-extend

Cappato Romain<sup>1</sup>, François Guibault<sup>1</sup>, Christophe Devals<sup>1</sup>, Bernd Nennemann<sup>2</sup>

<sup>1</sup> École Polytechnique de Montréal, Montréal, QC, Canada

<sup>2</sup> Andritz Hydro Canada Inc, Pointe-Claire, QC, Canada

E-mail: francois.guibault@polymtl.ca

**Abstract.** In the development of high head hydraulic turbines, vibrations are one of the critical problems. In Francis turbines, pressure fluctuations occur at the interface between the blades of the runner and guide vanes. This rotor-stator interaction can be responsible for fatigue failures and cracks. Although the flow inside the turbomachinery is complex, and the unsteadiness makes it difficult to model, the choice of an appropriate setup enables the study of this phenomenon. This study validates a numerical setup of the Foam-extend open source software for rotor-stator simulations. Pressure fluctuations results show a good correspondence with data from experiments.

## 1. Introduction

In hydraulic turbines, the movement of the runner blades passing in front of the guide vanes creates pressure fluctuations which may be harmful to the machine. This phenomenon is called rotor-stator interactions (RSI). High head Francis turbines are particularly affected by this phenomenon. The vibrations induced by the pressure fluctuations can induce fatigue failure and are detrimental to the overall efficiency of the hydroelectric power station.

Many failures were initially related to materials problems, but experimental studies ([1], [2]) show that rotor-stator interactions are responsible for damage to turbine components. Several studies have therefore been carried out concerning these interactions [3]. Some theoretical studies [4] and [5] have allowed to anticipate the frequency and the amplitude of vibrations. Since the 2000s, numerical studies ([6], [7], [8]) are also available that have predicted the transient behavior of turbines. They were realised with commercial softwares such as CFX or Fluent. These studies are however costly in terms of calculation time, and thus are expensive in terms of commercial software licencing. Furthermore, not being able to access all simulation parameters through commercial software interfaces prevents users from tailoring simulations to their specific needs and making simulations more reliable.

This work investigates various aspects of the numerical setup required to perform accurate and robust URANS simulations of an RSI case using the open source CFD software Foam-extend. This study is based on previous work concerning hydraulic turbine simulations with OpenFoam ([9], [10]). The calculations are realised with the Foam-extend 3.2 version of the software. The



solver chosen for this study is a transient solver for incompressible flow on a moving mesh, coupled with a  $k-\varepsilon$  turbulence model. The geometry studied is a model of a Francis turbine, which consists of a runner (15 blades) and a distributor (20 guide vanes and 20 stay vanes). To enable the connection between the distributor outlet and the runner inlet, the General Grid Interface (GGI) method is used. All numerical results obtained are compared to experimental data and commercial software results.

## 2. Numerical approach

### 2.1. OpenFoam

Foam-extend is a high quality multi-physics platform used to simulate continuum mechanics problems including computational fluid dynamic (CFD). It is an open-source software. Resolution is based on the finite volumes method. The advantage of this software is that it provides direct access to model and solver implementations. All users may create and personalize solvers or libraries to solve their problems.

The OpenFoam community is working to grow the popularity of the OpenFoam project in academia and in industry. Recently, a new version of the product, Foam-extend, was released, that offers CFD researchers and engineers a new platform to develop their projects. Among the community of OpenFoam users, the Turbomachinery working group has developed tools to run turbomachinery simulations. The ERCOFTAC pump case, developed in [10] and [11], was an excellent starting point for our study.

### 2.2. RSI approach

The study of rotor-stator interactions is a complex problem. Before attempting to run simulations, it is important to understand which assumptions are allowed. If the distance between the runner and guide vanes is large enough, pressure fluctuations could be avoided, and a steady state simulation is sufficient to analyse the behaviour of the turbine. In the case of a Francis hydraulic turbine, simulations must be treated with a transient solver to capture the effect of RSI. However a steady state simulation is still run to initialize the transient case.

### 2.3. Steady-state study

The Multiple Reference Frame solver (MRF) is used to run steady state simulations. The MRF solver is a steady state solver for incompressible flow of Newtonian fluids with multiple reference frames. It allows simulating the rotation of the runner without a moving mesh. Coriolis and centrifugal source terms have been added to calculate the influence of the runner rotation. Equation 1 is solved to compute absolute Cartesian velocity in the rotating frame [12],

$$\nabla(\vec{u}_R \otimes \vec{u}_I) + \vec{\Omega} \times \vec{u}_I = -\nabla(p/\rho) + \nu \nabla \cdot \nabla(\vec{U}_I) \quad (1)$$

where  $\vec{u}$  is the velocity vector,  $\vec{\Omega}$  is the rotational speed vector,  $\rho$  the density, and  $p$  the pressure. The subscript  $I$  stands for the inertial and  $R$  for the rotating frame respectively.

### 2.4. Unsteady study

To capture RSI effects, simulations must be run in an unsteady flow regime. There are a number of transient solvers available in Foam-extend to carry out incompressible flow simulations. For most of these solvers, the mesh is moving. At each time step, the runner mesh moves prior to computing the momentum predictor, and the interface is reconstructed. The coordinate system remains fixed in this approach. The Cartesian velocities are still computed. The difference with the MRF approach is that the relative velocities are inserted in the convective terms [12], as

$$\begin{aligned} \int_S \rho \vec{u} \cdot \vec{n} dS &\rightarrow \int_S \rho (\vec{u} - \vec{u}_b) \cdot \vec{n} dS \\ \int_S \rho u_i \vec{u} \cdot \vec{n} dS &\rightarrow \int_S \rho u_i (\vec{u} - \vec{u}_b) \cdot \vec{n} dS \end{aligned} \quad (2)$$

where  $\vec{u}$  is the velocity vector,  $u_i = (u_x, u_y, u_z)$  are the Cartesian components of the velocity vector  $\vec{u}$ , and  $\vec{u}_b$  is the boundary (face) velocity.

The two transient solvers selected for our study are the transientSimpleDyMFoam and pimpleDyMFoam solvers. Both solvers are segregated solvers that iteratively solve the velocity and pressure fields in distinct algorithmic steps. The main difference between these two solvers lies in the pressure correction algorithm. The TransientSimpleDyMFoam solver uses the SIMPLE algorithm while PimpleDyMFoam uses the PIMPLE algorithm, which is a combination of the SIMPLE and PISO algorithms. The results given by these solvers are discussed in section 3.1.

### 2.5. Turbulence model

The main cause of RSI comes from the interference between runner blades and guide vanes. Viscous effects are secondary. Nevertheless, a URANS (Unsteady Reynolds Average Navier Stokes) approach is used. According to ([7], [13]), the k- $\epsilon$  turbulence model is adequate to perform CFD simulations for these types of configurations. It has been shown that the k- $\epsilon$  turbulence model is robust and reliable. Moreover, it enables the use of a not too fine spatial discretization.

The k- $\epsilon$  model is based on the resolution of transport equations for the turbulent kinetic energy k and the dissipation  $\epsilon$ . The k- $\epsilon$  model has been implemented and thoroughly validated in OpenFOAM, and requires less calculation time than other RANS turbulence models.

## 3. Simulation parameters

### 3.1. Resolution algorithms

For this study, segregated solvers with pressure-correction were used. Among these solvers, Foam-extend offers the possibility to use two main algorithms: the SIMPLE and PISO algorithms. For these solvers, the pressure field is obtained by deriving a pressure correction equation and enforcing mass continuity. [14]

The way this is achieved is as follow [15]. To initiate the Semi-Implicit Method for Pressure Correction (SIMPLE) algorithm, the pressure field is predicted. The velocity field is computed by solving the discretized momentum equations with the predicted pressure. This first solution is substituted into the equation of continuity in order to calculate correction factors. In the SIMPLE algorithm, the velocity corrections contributed by cells adjacent to the pole cell are neglected. This approximation does not affect the final solution if convergence is reached. It is acceptable for steady state simulations or if small time steps are used in an unsteady calculation. This step enables to correct the pressure and the velocity. In the last step, all other transport equations are solved. To avoid divergence, some under-relaxation is used. The new pressure is obtained as

$$p^{new} = p^* + \alpha p' \quad (3)$$

where  $\alpha$  is the pressure under-relaxation factor. The  $\alpha$  factor needs to be large enough to move the process forward but small enough to avoid divergence.

The Pressure Implicit with Splitting of Operators (PISO) algorithm may be seen as an extension of the SIMPLE algorithm with a further corrector step. The first steps of the SIMPLE algorithm are realised. The next step of the PISO algorithm consists in solving a second pressure correction equation without neglecting any term. In the PISO algorithm, no under-relaxation

factor is used. It has been shown that despite the increase of computational effort to solve the second pressure equation, the PISO algorithm is efficient and fast.

The PIMPLE algorithm is a combination of these two algorithms. With default parameters, the PISO part of the algorithm is used. To benefit from the SIMPLE part of the algorithm, relaxation factors have to be introduced. The PIMPLE algorithm acts like the PISO one, with under-relaxation correction at the end.

The choice between these two solution algorithms (SIMPLE for transientSimpleDyMFOAM and PIMPLE for pimpleDyMFoam) depends on user needs. If robust simulations are needed, the SIMPLE algorithm is simpler to use through the transientSimpleDyMFoam solver [16]. However if the need is for an optimized calculation, the PIMPLE algorithm (with pimpleDyMFoam) can be set up to benefit from the convergence of the SIMPLE algorithm and the precision and speed of the PISO one. In our study, the choice of a robust solution is preferred over an optimized calculation.

### 3.2. Discretization schemes

Another parameter to set is the choice of the time discretization scheme. The main schemes used are Euler and Backward. The difference between these two schemes is the degree of precision [17]. Euler is a first order scheme. To run transient calculations, a higher order scheme such as the Backward scheme is preferred.

### 3.3. Matrix solvers

In addition, matrix solvers need to be specified. Several matrix solvers are available in Foam-extend. They differ in the hypothesis they assume for the matrix to solve. Some of the major solvers are :

- PBiCG - preconditioned bi-conjugate gradient solver for asymmetric matrices;
- PCG - preconditioned conjugate gradient solver for symmetric matrices;
- GAMG - generalised geometric-algebraic multi-grid solver

Velocities and turbulence matrices are not symmetric, so they must be solved with the PBiCG solver. For pressure, the GAMG solver is a good choice in terms of reliability and computing speed in a steady state simulation. For transient simulations, the PCG solver is used for pressure, to insure stability.

## 4. Study case

### 4.1. Mesh and geometry

The mesh is created with an in-house mesh generation tool developed jointly by Polytechnique and Andritz. The rotor and the stator were meshed separately, and assembled with the OpenFoam tool, mergeMesh. The characteristics of the mesh used are given in Table 1

**Table 1.** Number of cells for the simulation

Mesh	Rotor	Stator	Total
Coarse (used for the initial approach)	1741920	2472440	4214360

To connect these two meshes, the General Grid Interface (GGI) was used. The two meshes are coupled in an implicit manner without the need to topologically change the mesh. Depending on relative face overlap, cells from the runner mesh communicate with a number of cells of the distributor mesh and vice versa. The GGI approach is based on a facet-area based interpolation

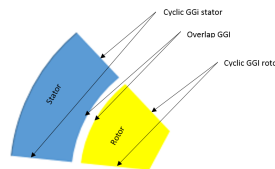
scheme across the interface. Weighting factors are calculated to set the interpolation. According to [18], second order accuracy and strict mass flow are preserved.

The basic GGI interface can only be used for steady state simulation without overlapping faces. To run unsteady simulations with or without an aligned layout, the overlapGGI interface is used [19]. This interface was developed for unsteady simulations, and enables overlapping faces for reduced models (cf. 4.2).

#### 4.2. Reduced geometry

Unsteady simulations consume a lot of calculation time. A reduction of the geometry is one of the solutions to speed up calculations. In almost all hydraulic turbines, the number of guide vanes is different from the number of blades. Running a simulation with only one runner blade and one guide vane passage involves averaging quantities at the interface. If the aim is to capture the effect of the rotor-stator interaction, this approach is not acceptable. However, to speed up calculations, some reduction could be made. According to [6], it is possible to reduce the domain as long as the ratio  $As.stat/As.rot = 1 \pm 0.01$  is maintained, where  $As.stat$  is the ratio of the number of guide vanes in the reduce model over the number of guide vanes in the complete geometry, and  $As.rot$  is the number of runner blades in the reduced model over the number of runner blades in the complete geometry

Some variants of the GGI approach are available in order to reduce the model (cf. Fig. 1). To set up cyclic conditions, the cyclicGgi approach is used. Weighting factors are used to connect the two cyclic patches. The only option to set is the rotation angle between the two faces. To set the two overlapping interfaces, at the distributor outlet and at the runner inlet, the overlapGgi interface is used. For partial geometries, the overlapGgi interface duplicates the interface with the option nCopies, and calculates the flux between these interfaces.



**Figure 1.** Cyclic interfaces for a reduced model

#### 4.3. Boundary conditions

**Table 2.** Boundary conditions of the case

Inlet	$U$	Flow rate $Q = 0.208m/s$
	$k, \varepsilon$	$k = 0.0007601$
		$\varepsilon = 0.0058249$
	$p$	<i>zeroGradient</i>
Outlet	$p$	fixedMeanValue = 0
	$U, \varepsilon, k$	inletOutlet
Wall	$U, p, k, \varepsilon$	wall functions
Distributor-runner interfaces	$U, p, k, \varepsilon$	Steady case : Ggi
		Unsteady case : OverlapGgi

#### 4.4. Parallel computations

To speed up calculations, simulation can also be run in parallel. With a software such as OpenFoam, domain decomposition for parallel processing must be performed carefully in order to obtain correct results. In our case, with two meshes merged using a GGI interface, a bad decomposition can lead to results that are completely wrong without any error message. Specifically, all GGI interfaces must be assigned to the same processor. The method used to do this consists in imposing a patch constraint through a method available in OpenFoam.

This method is especially crucial for the reduced model case. For this specific configuration, all GGI interfaces (Overlap GGI and cyclic GGI, see Fig. 1) must be assigned to a single processor. This condition on the decomposition is an impediment to speeding up calculations.

#### 4.5. Time step

Another parameter which needs to be examined is the time step. Indeed, for unsteady simulations the choice of the time step is crucial. If the time step is too large, pressure fluctuation could not be analysed. To be efficient, the time step should correspond to a  $0.2^\circ$  rotation of the runner.

#### 4.6. Case setup

According to all assumptions made in section 3, the settings of the steady and unsteady simulations are listed in table 3 and 4 respectively. For both simulations, pressure matrixes are solved with the PCG solver. For the velocity and the turbulence variables, the BiCGS solver is used. The steady state starts using a first order discretization scheme to ensure stability. After 100 iterations the higher order scheme Bacward is used to accelerate convergence.

As regards the tolerances, relative tolerance of 0.01 is used in the steady state simulation for the pressure variable to speed up the calculation. However, a smaller relative tolerance of 0.001 is set in the transient simulation to obtain better results.

**Table 3.** Setting for the steady simulation

Schemes	$U$	Gauss upwind	the first 100 iterations
		Gauss linearUpwind Gauss linear	Until the end of the simulation
	$k, \varepsilon$	Gauss upwind	
Solvers	$p$	PCG	
		preconditionner	DIC
		Tolerance	1e-08
		relTol	0.01
	$U, k, \varepsilon$	BiCGStab	
		tolerance	1e-07
		relTol	0.0

In a first stage, the transient solver was used without the p-U correction outer loop. As a result, the velocity convergence was good, but errors on the pressure appeared. To correct these errors and obtain small final residuals on both velocity and pressure, the transient solver needs to use several p-U correction outer loops. The velocity convergence is slower but stays acceptable, and the pressure is corrected. Our study on the optimisation of the calculation time shows that 3 outer loop corrections are necessary but sufficient to obtain a good agreement.

**Table 4.** Setting for the unsteady simulation

Schemes	$U$	Gauss linearUpwind Gauss linear	
	$k, \varepsilon$	Gauss upwind	
	$TimeDiscretization$	backward	
Solvers	$p, p_{corr}, p_{Final}$	PCG	
		preconditionner	DIC
		Tolerance	1e-05
		relTol	0.001
	$U, k, \varepsilon$	BiCG	
		preconditionner	DILU
		smoother	DILU
		tolerance	1e-07
		relTol	0.0
Control settings	TimeStep	0.00003	
	nOuterCorrectors	3	

## 5. Results

### 5.1. Steady simulation

Simulations have first been run with the complete geometry for the steady-state case. Results from Foam-extend were compared to those obtained using a commercial software. The comparison was based on average velocity, average relative velocity and static pressure on one hundred planes in the turbine. Foam-extend average values show good correspondence with results from the commercial software. This first approach is validated and allows initializing the transient calculations with the complete turbine steady-state results.

Although the complete turbine simulations present good results, errors occur when using a reduced model of the turbine (1/5 of the complete geometry). The trend in the average pressure and velocity are respected but values are not completely correct.

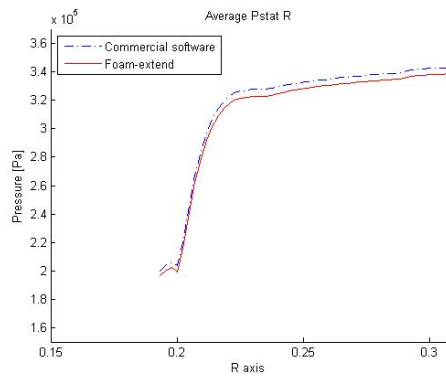
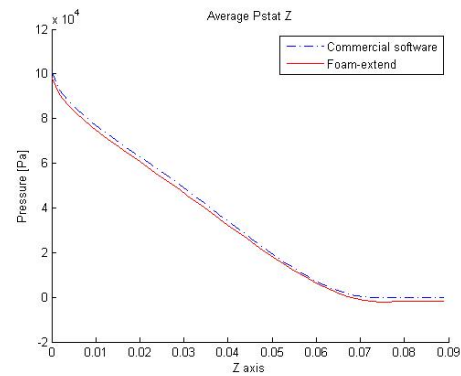
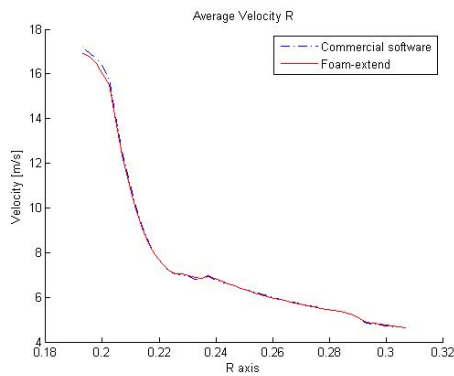
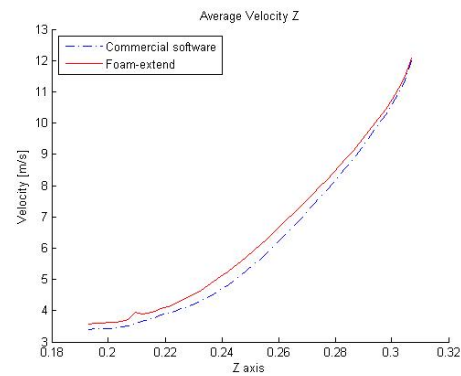
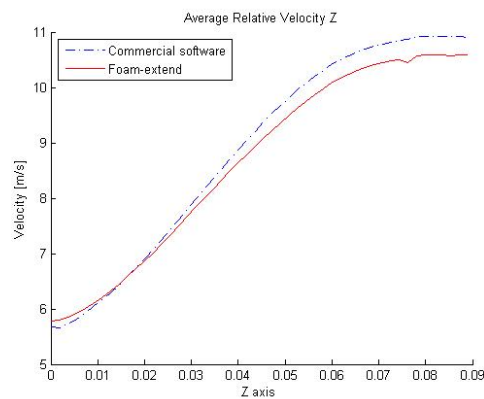
### 5.2. Unsteady simulation

Unlike the steady simulation, which averages values at the interface, the transient simulation enables to observe the rotor-stator interactions. The study is performed using the transientSimpleDyMFoam solver. The overlapGgi interface is used to connect the two meshes.

Figures 2 to 6 show the comparison between the Foam-extend calculation and solutions obtained using a commercial flow solver. These results show a good correspondence. The average pressure, the average velocity and the average relative velocity are calculated on cylinders in the radial direction and on planes in the axial direction. We get very similar trends between the two solvers.

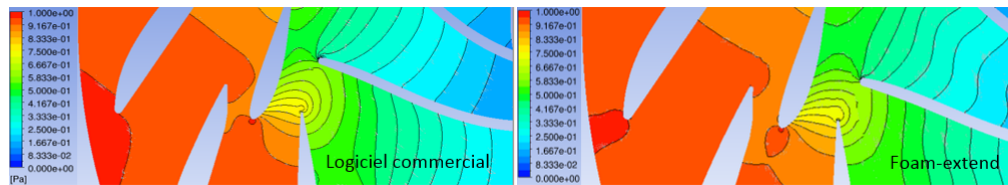
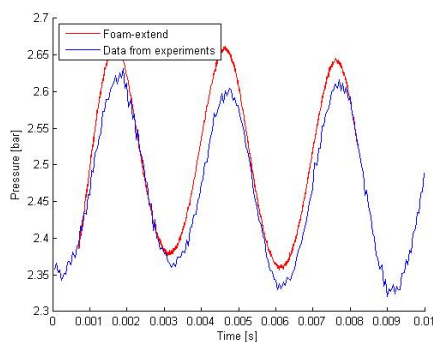
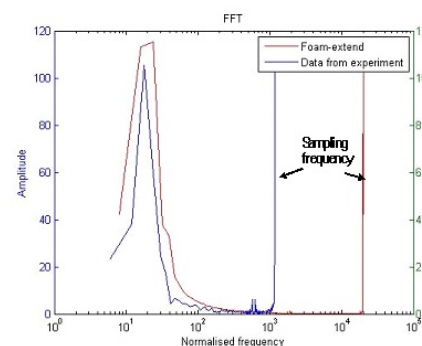
To observe rotor-stator interaction, it is more appropriate to look at pressure fluctuations obtained near the blades. Figure 7 shows wakes of pressure at the interface. The comparison between commercial software results and Foam-extend results shows a good agreement.

Futhermore numerical probes were implemented to follow the pressure side and the suction side of the blades. These probes are moving with the mesh and enable us to observe the fluctuations. The experimental study was conducted by the the laboratory for hydraulic machines in Lausanne. Pressure fluctuations were captured with on-board sensor located on the blades of the Francis turbine model. Numerical results obtained are compared to these experimental results.

**Figure 2.** Average static pressure on R**Figure 3.** Average static pressure on Z**Figure 4.** Average velocity on R**Figure 5.** Average velocity on Z**Figure 6.** Average relative velocity on Z

Time fluctuation and spectral analysis were made. All probes show a good correspondence with the experimental results. Figure 8 and 9 show results for one of the many probes followed.

The simulations with the reduced geometry (1/5 of the complete turbine), did not yield good results. Indeed, the flow behaviour quickly becomes unsteady and the simulations diverge.

**Figure 7.** Static pressure at the interface**Figure 8.** Time analysis**Figure 9.** Spectral analysis

## 6. Discussion

Results obtained with the transientSimpleDyMFoam solver show that the study of RSI, with a transient simulation is possible with Foam-extend. Indeed, the flow behaviour is well preserved. Furthermore the overlapGgi interface was validated to enable the work on RSI. The choice of other solvers like the pimpleDyMFoam solver is really unstable and difficult to set up. Many tests have been performed. The use of different relaxation factors (0.1 to 0.8), and several correction steps did not allow to reach convergence. The value of the velocity in the flow, increases and diverges after a few iterations. In the interest of time, not all combinations of these solution have been tried, but future work will further address this issue. The choice of the Simple algorithm-based solver appears to yield the most robust simulations among solvers tested.

Using Foam-extend, the calculation time required for these simulations is really long. With a commercial solver, it takes nearly 30 hours to get a converged result on a single processor. With Foam-extend, it takes more than 35 hours with 48 processors. This huge difference could be explained by two aspects : First, the cyclic GGI package didn't enable us to obtain good results with the reduced geometry. Secondly, the reference commercial code uses a coupled solver to run the simulations and not a segregated approach. Furthermore, to obtain a converged result, the p-U correction outer loop increases dramatically the calculation time.

For the partial geometry, there is more work to do to integrate cyclic interfaces in a stable calculation. Even if these interfaces seem to run well on some geometries, due to an excellent work of the turbomachinery group, it is still hard to implement robust simulations using these interfaces for any geometric configuration.

## 7. Conclusion

The flow in a Francis turbine was simulated with the Foam-extend software to investigate the rotor-stator interaction phenomenon. From this study, we may conclude that the Foam-extend

software is able to capture the RSI transient effect and proves to be a valid alternative to commercial softwares. But, even if results show a good correspondence, they may be improved. The lack of technical documentation is an impediment in improving these simulation. Even though there are tutorials and studies about rotating geometries with Foam-extend, to learn how to set up a case takes a lot of time. Furthermore, it's hard to understand the meaning of each Foam-extend parameter and the impact on the physical solution.

Future works will be dedicated to the optimization of the different parameters of the simulation in order to reduce calculation time and obtain faster convergence. Furthermore, an investigation about the setup of reduced geometries will be performed, in order to speed up simulations and enable the use of these types of simulation. Finally, the use of a coupled solver could also represent an interesting new lead.

## 8. Acknowledgement

The authors wish to thank Professor Håkan Nilsson from the Chalmers University, for his advice and help. The authors would like to thank the National Science and Engineering Research Council of Canada (NSERC) for its support of this research, under project RDCPJ 441752-12.

## References

- [1] Egusquiza E, Valero C, Huang X, Jou E, Guardo A and Rodriguez C 2012 *Engineering Failure Analysis* **23** 27–34
- [2] Rodriguez C G, Mateos-Prieto B and Egusquiza E 2014 *Shock and Vibration*
- [3] Zuo Z, Liu S, Sun Y and Wu Y 2015 *Renewable and Sustainable Energy Reviews* **41** 965–974
- [4] Tanaka H 2011 *International Journal of Fluid Machinery and Systems* **4** 289–306
- [5] Rodriguez C, Egusquiza E and Santos I 2007 *Journal of Fluids Engineering* **129** 1428–1435
- [6] Nennemann B, Vu T C and Farhat M *HYDRO 2005 LMH-CONF*
- [7] Keller M, Dörfler P, Sallaberger M and Sick M *HYDRO 2005*
- [8] Yan J, Koutnik J, Seidel U and Hübner B 2010 *IOP Conference Series: Earth and Environmental Science* vol 12 (IOP Publishing) p 012008
- [9] Stoessel L and Nilsson H 2015 *Journal of Physics: Conference Series* vol 579 (IOP Publishing) p 012011
- [10] Petit O, Nilsson H, Page M and Beaudoin M 2009 *3rd IAHR International Meeting of the Workgroup on Cavitation and Dynamic Problem in Hydraulic Machinery and Systems, Brno, Czech Republic*
- [11] Ubaldi M, Zunino P, Barigozzi G and Cattanei A 1996 *Journal of Turbomachinery* **118** 41–51
- [12] Nilsson H 2012 Turbomachinery training at OFW7
- [13] Fontanals A, Coussirat M, Guardo A and Egusquiza E 2010 *IOP Conference Series: Earth and Environmental Science* vol 12 (IOP Publishing) p 012089
- [14] Moukalled F, Mangani L and Darwish M 2016 *The Finite Volume Method in Computational Fluid Dynamics* (Springer)
- [15] Versteeg H and Malalasekera W 1995 *An introduction to computational fluid dynamics: the finite volume method* (Pearson Prentice Hall)
- [16] Xie S 2010 *Studies of the ERCOFTAC centrifugal pump with OpenFOAM* Master's thesis Chalmers University of Technology
- [17] Jasak H 1996 *Error analysis and estimation for the finite volume method with applications to fluid flows* Ph.D. thesis Imperial College London (University of London)
- [18] Jasak H and Beaudoin M 2011 *ASME-JSME-KSME 2011 Joint Fluids Engineering Conference* (American Society of Mechanical Engineers) pp 1801–1812
- [19] Moradnia P, Nilsson H, Page M, Beaudoin M, Torriano F, Morissette J F and Toussiant K 2012 Transient and steady-state air flow simulations in generators using openfoam Tech. rep. Chalmers University of Technology