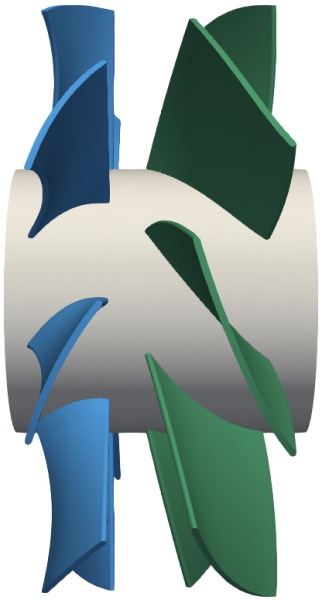


Blade Element Momentum Method for a Counter-Rotating Pump-Turbine

Study of applicability

Master's thesis in Applied Mechanics

CRISTÓBAL IBÁÑEZ URIBE

DEPARTMENT OF MECHANICS AND MARITIME SCIENCES

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2021
www.chalmers.se

Master's Thesis 2021:16

Blade Element Momentum Method for a Counter-Rotating Pump-Turbine

Study of applicability

CRISTÓBAL IBÁÑEZ URIBE

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mechanics and Maritime Sciences
Division of Fluid Dynamics
Chalmers University of Technology
Gothenburg, Sweden 2021

Blade Element Momentum Method for a Counter-Rotating Pump-Turbine
Study of applicability
CRISTÓBAL IBÁÑEZ URIBE

Supervisor: Dr. Hamidreza Abedi, Mechanics and Maritime Sciences, Chalmers.
Supervisor: PhD student Jonathan Fahlbeck, Mechanics and Maritime Sciences, Chalmers
Examiner: Prof. Håkan Nilsson, Mechanics and Maritime Sciences, Chalmers.

Master's Thesis 2021:16
Department of Mechanics and Maritime Sciences
Division of Fluid Dynamics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Blade geometry of both counter-rotating runners conforming the pump-turbine.

Typeset in LaTeX
Gothenburg, Sweden 2021

Blade Element Momentum Method for a Counter-Rotating Pump-Turbine
Study of applicability
CRISTÓBAL IBÁÑEZ URIBE
Department Mechanics and Maritime Sciences
Chalmers University of Technology

Abstract

Blade Element Momentum (BEM) methods are widely used for initial aerodynamic analysis of wind and tidal turbines, as well as aircraft and marine propellers. Mainly because they use less computational resources and can give fairly accurate results. This thesis studies the applicability of such method for load prediction on the runner blades of a model scale axial shaft-driven Counter-Rotating Pump-Turbine (CRPT).

One of the main assumptions and modifications adopted in the BEM method was to omit the axial induction factor, a ,and let the axial velocity be constant. This considering that the volumetric flow rate, q, and cross-sectional area, A, must remain unchanged throughout the turbomachine. Additionally, an attempt to include the constant pressure difference across the rotors as extra loading being exerted on the blade, as well as a suggestion on how to apply the BEM method on the downstream rotor are presented and evaluated.

The data needed to implement the BEM method is created by running several CFD simulations using OpenFOAM. These simulations produce 2D aerodynamic airfoil-like characteristics of lift and drag coefficients, C_l and C_d, at different angles of attack (AoA) and Re numbers in both pump and turbine mode for different blade profiles along the radial direction. Furthermore, validation cases at different operating conditions are also simulated with OpenFOAM assuming steady-state flow. This is done for different geometries. One of them considers the turbomachine with both runners operating simultaneously and the other geometries isolate each runner individually.

It is concluded that it is possible to use the BEM method with a reasonable amount of error for certain operating conditions. Most importantly the behaviour of the dimensionless thrust and power coefficients, C_T and C_P at different tip speed ratios, TSR, tends to follow the same trend as CFD. Further work needs to be done in order to fully validate such a method for this type of turbomachine.

Keywords: BEM, CFD, Counter-Rotating, Pump-Turbine, Hydro-power, OpenFOAM.

Preface

This projects evaluates the applicability of the classical Blade Element Momentum (BEM) method, widely used in wind turbine industry, to calculate the thrust and torque loadings on both runners of a model scale Counter-Rotating Pump-Turbine (CRPT) machine. The objective is to develop a BEM-based MATLAB routine that could eventually be incorporated to a predictive system control that keeps track of the reference power. Several CFD simulations with OpenFOAM were carried out to create the data necessary to run and also validate the BEM-method proposed. The examiner was Håkan Nilsson, full professor at the Department of Mechanics and Maritime Sciences, Division of Fluid Dynamics at Chalmers.

Acknowledgements

First I would like to thank my examiner Professor Håkan Nilsson for giving me the opportunity to work on this thesis. Your assertive comments and suggestions have helped this project reach to a good ending. Thank you for your patience and great ability to understand and get what I am trying to say, even though I many times stumble upon my own thoughts while explaining.

I would also like to express my gratitude to Dr. Hamidreza Abedi. Thank you for always making the time to meet me and for the discussions and explanations about the BEM method that have enriched this work. Thank you also for your encouragement and positive comments about my own potential as a student.

Furthermore, I am grateful to Phd student Jonathan Fahlbeck. Thank you for helping me with my first steps with OpenFOAM and for providing the validation case set-ups for this project. I am grateful for your quick response and for always making the time to help me clarify the doubts and many questions I had.

Lastly, I want to thank Julia. Thank you for your endless support, especially during the times I have struggled during this project.

The computations were enabled by resources provided by the Swedish National Infrastructure for Computing (SNIC) at Chalmers Centre for Computational Science and Engineering (C3SE) partially funded by the Swedish Research Council through grant agreement No. 2018-05973.

Cristóbal Ibáñez Uribe, Gothenburg, June 2021

Nomenclature

Abbreviations

ALPHEUS	Augmenting Grid Stability Through Low Head Pumped Hydro Energy Utilization and Storage
BEM	Blade Element Momentum
CFD	Computational Fluid Dynamics
CRPT	Counter-Rotating Pump Turbine
MRF	Multiple Rotating Frames of reference
OpenFOAM	Open Source Field Operation And Manipulation
RANS	Reynolds Averaged Navier Stokes
RPT	Reversible Pump-Turbine

Dimensionless Quantities

C_d	Drag coefficient
C_l	Lift coefficient
C_n	Axial force coefficient
C_P	Power coefficient
C_Q	Torque coefficient
C_T	Thrust coefficient
C_t	Tangential force coefficient
J	Advance ratio
TSR	Tip speed ratio

Greek Letters

α, AoA	Angle of attack
β	Twist angle
η	Efficiency
ν	Kinematic viscosity
Ω	Runner rotational speed
ϕ	Flow angle
ρ	Density
σ	Solidity
F	Prandtl loss factor

Roman Letters

$\dot{m}$	Mass flow
A	Area

a	Axial induction factor
a'	Tangential induction factor
a'_{d}	Tangential induction downstream runner
a'_{u}	Tangential induction upstream runner
F_n	Axial (normal) force component
F_t	Tangential force component
H	Head difference
P	Power
p	Pressure
p_{tot}	Total pressure
Q	Torque
q	Volumetric flow
T	Thrust
U_0	Freestream incoming axial velocity
u_{R}	Axial velocity at the rotor plane
u_θ	Tangential velocity
Z	Number of rotor blades

Contents

List of Figures

List of Tables

1
Introduction

1.1 Background

Electric load balance when there is high electrical power demand and low supply can be obtained by means of pumped-storage hydroelectricity. In this sense the EU-project ALPHEUS [1], where Chalmers is one of the academic partners, was conceived. Its main purpose is to improve reversible water pump-turbine (RPT) technology, as well as the civil infrastructure needed to make pumped hydro-storage economically viable in shallow coastal environments and places with flat topography, like e.g. Belgium or the Netherlands.

Given that a CFD simulation of the flow through the RPT is computationally expensive and a time consuming process, the purpose of this thesis is to develop a faster method that can be incorporated to a multi-dimensional model-based predictive control system that will keep track of the reference power. The Blade Element Momentum (BEM) method, widely used in wind power assessment, fulfills this requirement and can be implemented in a simple MATLAB routine. Adaptations to the classical BEM method can be tested to try and make it adequate to the current turbomachine.

1.2 Aim

The aim of this thesis is to develop and validate a BEM-based code in MATLAB for a Counter-Rotating Pump-Turbine (CRPT) design. The code will output the total thrust, T, torque, Q, and power, P, for any combination of incoming volumetric flow, q, and rotational speed of the rotors, Ω.

The validation is carried out using the open source CFD software OpenFOAM. The intention is to evaluate and assess the validity of implementing a BEM method for the current turbomachine. No such efforts are known to have been carried out for either a water pump or a turbine configuration. Hopefully this effort could be reviewed in the future, or serve as inspiration to fully validate such a method that can save a lot of computational resources and be coupled to a predictive control system of a turbomachine.

1.3 Limitations

In this work only the current design in model scale will be used to carry out the BEM-based code and the validation. No focus will be put in optimizing the existing design in terms of improving its efficiency or performance.

In-depth analysis of the flow through the CRPT and full flow characterization are not the main purpose of the turbomachine's CFD simulations. The post-processing of these simulations is focused on extracting what is needed to compare and validate the BEM calculations at different operating conditions. Namely the loadings on the rotor blades.

2

Theoretical Framework

2.1 Counter-Rotating Pump-Turbine

Reversible Pump-Turbine designs used in pumped-storage schemes are generally based on the Francis turbine design [4]. However these have limited performance under low-head scenarios when there is a small height difference between both reservoirs. A shaft-driven axial CRPT configuration becomes a promising alternative [10]. In pump mode it can operate at lower rotational speeds, preferable head characteristics and at high efficiency for a wider range of operating conditions than a conventional single rotor configuration [5].

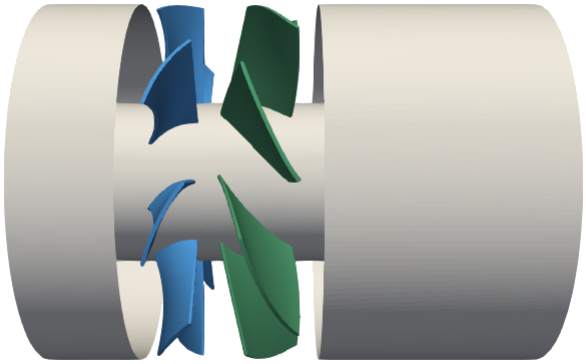


Figure 2.1: Schematic view of the CRPT geometry. Runner 1 in blue and Runner 2 in green color.

Figure 2.1 shows a schematic view of both counter-rotating runners inside the guiding pipe where water flows through in any given direction depending on which operating mode the machine is working on. In this sense, when the machine is operating in pump mode the flow goes from left to right according to the figure, where Runner 1 (blue) comes first upstream and Runner 2 (green) comes after downstream. In turbine mode the flow goes from right to left according to the picture and the upstream rotor becomes Runner 2 (green), therefore Runner 1 (blue) becomes the downstream rotor.

The current geometry is in reduced model scale and the runners' diameter is 276 mm. Runner 1 contains 8 blades and Runner 2 is comprised of 7 blades. Furthermore,

both rotors are independently driven and can therefore rotate at different rotational speeds. This feature gives a wide range of possible operating conditions where the ratio between both runners' rotational speeds can be adjusted.

2.2 CRPT Efficiency

The power that the pump can deliver to the flow, or the power the turbine can extract, can be defined as the total torque exerted on the rotor blades by the flow, Q in [Nm], multiplied by the rotational speed of the runner, Ω in [rad/s] as

$$P = Q\Omega. \tag{2.1}$$

Here P is the power in [W]. The pressure head, ΔH, is the difference in total pressure across the rotor, Δp_{tot}, expressed in units of meters [m] and is expressed as

$$\Delta H = \frac{\Delta p_{\text{tot}}}{\rho g}. \tag{2.2}$$

Here ρ is the water density and g is gravity acceleration. A hydraulic pump efficiency is defined as the ratio between the power imparted on the fluid and the power supplied to drive the pump. On the other hand, a hydraulic turbine efficiency is defined as the ratio between power extracted and the total power available. These can be written as

$$\eta_{\text{pump}} = \frac{q\rho g\Delta H}{P} \qquad \text{and} \qquad \eta_{\text{turb}} = \frac{P}{q\rho g\Delta H}. \tag{2.3}$$

Here q represents the volumetric flow rate in [m^3/s]. The head jump across the rotors can be non-dimensionalized as the head coefficient. This coefficient is commonly used for turbomachines and is defined as

$$\psi = \frac{gH}{\Omega^2 D^2}. \tag{2.4}$$

2.3 Simplified Axial Momentum Theory

A wind turbine extracts the kinetic energy in the wind and converts it to useful mechanical power. This energy extraction can be modelled by an idealized representation of the rotor plane as an actuator disk. In 1D axial momentum theory the static pressure drop across the rotor and the axial velocity are considered uniform over the rotor area. Furthermore, there is no rotational velocity in the wake and the static pressure far upstream and far downstream are thought as equal and equivalent to the ambient pressure.

Figure 2.2 displays a one-dimensional model for the streamtube that encloses the rotor. The streamtube expands as the axial velocity decelerates because of the presence of the actuator disk that extracts energy at the rotor plane.

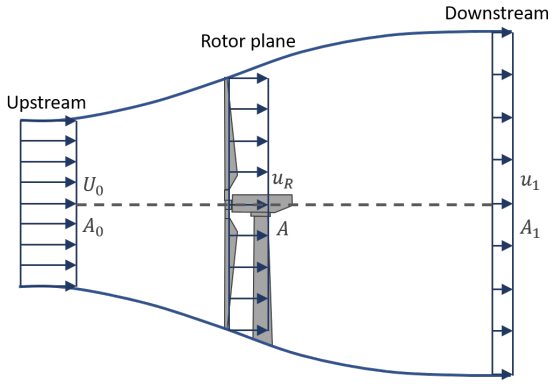


Figure 2.2: Control volume 1D actuator disk. Inspired by Sørensen [12].

From the continuity equation it is required that the mass flow remains constant at each cross section: upstream, rotor and downstream planes. The mass flow through these cross-sections is given by

$$\dot{m} = \int \rho u \mathrm{d}A = \rho U_0 A_0 = \rho u_R A = \rho u_1 A_1. \tag{2.5}$$

Here, u is the velocity component normal to the cross-section area and U_0 is the mean of u at the cross-section area far upstream A_0; u_R and A are the respective mean axial velocity and cross-section area at the rotor plane and u_1 and A_1 the same variables at the far downstream plane.

Since the static pressure in the wake far downstream is equal to the one far upstream, then, from an axial momentum balance in the given streamtube control volume, the thrust can be expressed as

$$T = \dot{m}\left(U_0 - u_1\right) = \rho u_R A \left(U_0 - u_1\right). \tag{2.6}$$

By applying the Bernoulli equation from far upstream until right before the rotor plane and similarly from right after the rotor plane until far downstream in the wake, we obtain

$$\text{Upstream: } p_0 + \frac{1}{2}\rho U_0^2 = p^+ + \frac{1}{2}\rho u_R^2 \tag{2.7}$$

$$\text{Downstream: } p^- + \frac{1}{2}\rho u_R^2 = p_1 + \frac{1}{2}\rho u_1^2. \tag{2.8}$$

Since $p_1 = p_0$, then Eq. 2.7 and 2.8 can be combined to obtain the pressure drop $\Delta p = p^+ - p^-$ across the rotor plane, yielding

$$\Delta p = \frac{1}{2}\rho\left(U_0^2 - u_1^2\right). \tag{2.9}$$

The thrust force in the streamwise direction originates from the pressure drop at the rotor and can be expressed using Eq 2.9, as

$$T = \Delta p A = \frac{1}{2}\rho\left(U_0^2 - u_1^2\right) A. \tag{2.10}$$

Equating both thrust expressions in Eq. 2.10 and Eq. 2.6 shows that the axial fluid velocity at the rotor plane can be expressed as the arithmetic average between the axial velocity far upstream and far downstream as

$$u_R = \frac{U_0 + u_1}{2}. \tag{2.11}$$

The axial induction factor can be defined as the percentage the axial flow is being decelerated at the rotor plane in relation with the incoming velocity. It is expressed as

$$a = \frac{U_0 - u_R}{U_0}. \tag{2.12}$$

From Eq. 2.12 we get that the axial velocity at the rotor plane can be expressed as $u_R = U_0(1-a)$. Replacing u_R in Eq. 2.11 we get that the velocity far downstream can be written as $u_1 = U_0(1-2a)$. Replacing both these expressions in Eq. 2.6 yields

$$T = 2\rho A U_0^2 a(1-a). \tag{2.13}$$

2.4 Blade Element Theory

The blade element theory considers the local events taking place at the actual rotor blade profiles. Figure 2.3a displays the resultant force and its components induced on a blade airfoil profile at a given radius from the center of rotation. Figure 2.3b shows where the blade element would be located at a given radius from the center of rotation.

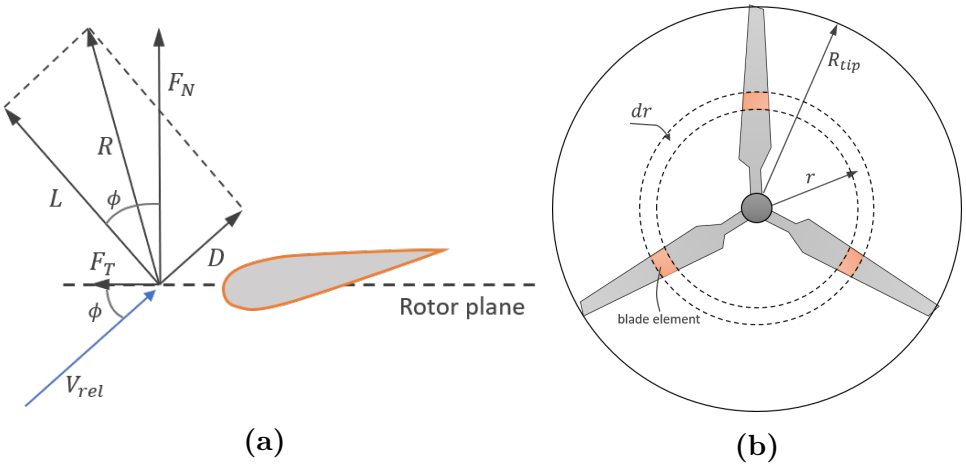


Figure 2.3: Cross sectional airfoil element and three-bladed rotor with radius R. Inspired by Hansen [6].

The axial and tangential force coefficients on the blade profile can be defined as

$$C_n = \frac{F_n}{1/2\rho c V_{rel}^2} \qquad \text{and} \qquad C_t = \frac{F_t}{1/2\rho c V_{rel}^2}. \tag{2.14}$$

Here c corresponds to the chord length of the airfoil profile and F_n and F_t are expressed in force per length. These coefficients are composed by the projections of lift and drag coefficients to the respective directions. These can be expressed as

$$C_n = C_l \cos\phi + C_d \sin\phi \tag{2.15}$$
$$C_t = C_l \sin\phi - C_d \cos\phi. \tag{2.16}$$

The thrust and torque on the control volume of thickness dr (Figure 2.3b) can be expressed as

$$\mathrm{d}T = ZF_n \mathrm{d}r \tag{2.17}$$
$$\mathrm{d}Q = rZF_t \mathrm{d}r. \tag{2.18}$$

Here Z is the number of blades. Replacing F_n and F_t, an expression for the thrust and torque on the blade element is obtained as

$$\mathrm{d}T = \frac{1}{2}\rho Z c V_{\mathrm{rel}}^2 C_n \mathrm{d}r \tag{2.19}$$
$$\mathrm{d}Q = \frac{1}{2}\rho Z c V_{\mathrm{rel}}^2 C_t r \mathrm{d}r. \tag{2.20}$$

Figure 2.4 shows in detail the velocity triangle formed locally on a blade profile for the turbine configuration. The axial velocity at the rotor plane is $U_0(1-a)$ as the flow decelerates when it passes through the rotor. The tangential velocity of the flow can be expressed with the tangential induction factor a' as $\Omega r a'$. In this sense the fluid has gained some tangential velocity which points in the opposite direction of rotation. This because the tangential force exerted on the blade by the flow points in the direction of rotation (Figure 2.3a), therefore, by reaction, the same force is exerted back on the flow by the blade.

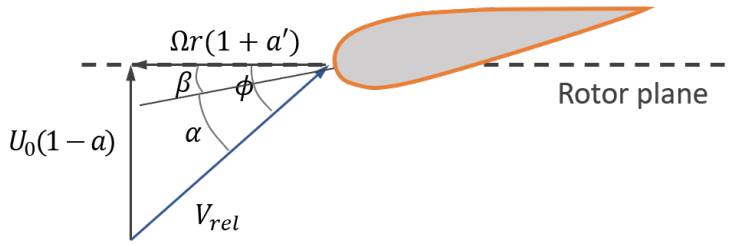


Figure 2.4: Turbine velocity triangle at the rotor plane. Inspired by Hansen [6].

The angle between the incoming relative velocity and the rotor plane is defined as the flow angle ϕ. The angle between the chord line and the rotor plane is known as the twist of the blade profile, β. Finally the local angle of attack, α becomes

$$\alpha = \phi - \beta. \tag{2.21}$$

The Re number of the airfoil profile is defined as

$$Re = \frac{V_{\mathrm{rel}} c}{\nu}. \tag{2.22}$$

Here ν is the fluid's kinematic viscosity and c the chord length of the airfoil. From the velocity triangle, the incoming relative velocity as seen by the moving rotor blade can be defined be either

$$V_{\text{rel}} = \frac{U_0(1-a)}{\sin\phi} \qquad \text{or} \qquad V_{\text{rel}} = \frac{\Omega r(1+a')}{\cos\phi}. \tag{2.23}$$

Therefore, by taking these expressions and replacing in Eqs. 2.19 and 2.20, the thrust and torque at the blade element can be related to the induction factors a and a' as

$$\mathrm{d}T = \frac{1}{2}\rho Z c \frac{U_0^2(1-a)^2}{\sin^2\phi} C_n \mathrm{d}r \tag{2.24}$$

$$\mathrm{d}Q = \frac{1}{2}\rho Z c \frac{U_0(1-a)\Omega r(1+a')}{\sin\phi\cos\phi} C_t r \mathrm{d}r. \tag{2.25}$$

2.5 Classical Blade Element Momentum (BEM) Theory

As the name suggest, the BEM theory combines both momentum theory (section 2.3) and blade element theory (section 2.4) to come up with expressions for the induction factors a and a'. These factors are used to express the velocity components of the flow at the rotor plane.

Eq. 2.13 can be expressed as a thrust differential considering an annular area within the actuator disk, where $\mathrm{d}A = 2\pi r \mathrm{d}r$. These translates to

$$\mathrm{d}T = 4\pi\rho U_0^2 a\,(1-a)\,r\mathrm{d}r. \tag{2.26}$$

From an angular momentum balance in an annular ring and replacing u_θ, which is the tangential velocity the flow has gained after the rotor plane, by $2a'\Omega r$ we get an expression for the differential torque as

$$\mathrm{d}Q = r u_\theta \mathrm{d}\dot{m} = r u_\theta \overbrace{\rho u_R 2\pi r \mathrm{d}r}^{\mathrm{d}\dot{m}} = 4\pi\rho U_0 \Omega a'\,(1-a)\,r^3 \mathrm{d}r. \tag{2.27}$$

The differential power is defined as the torque multiplied by the rotational speed

$$\mathrm{d}P = \Omega \mathrm{d}Q. \tag{2.28}$$

Equating Eq. 2.26 with Eq. 2.24 and Eq. 2.27 with Eq. 2.25 we are able to write and expression for the axial and tangential induction factors respectively, as

$$a = \frac{1}{\dfrac{4\sin^2\phi}{\sigma C_n} + 1} \qquad \text{and} \qquad a' = \frac{1}{\dfrac{4\sin\phi\cos\phi}{\sigma C_t} - 1}. \tag{2.29}$$

Here, σ is the solidity which is defined as the ratio of the annular area covered by the blade profile,

$$\sigma = \frac{Zc}{2\pi r}. \tag{2.30}$$

Dimensionless coefficients

Commonly for wind turbine application the thrust and power can be non-dimensionalized as thrust and power coefficients. For power coefficient, C_P, the denominator represents the maximum power the wind turbine can extract from the available kinetic energy. This coefficients are defined as

$$C_T = \frac{T}{\frac{1}{2}\rho A U_0^2} \qquad \text{and} \qquad C_P = \frac{P}{\frac{1}{2}\rho A U_0^3}. \tag{2.31}$$

Additionally, the tip speed ratio is defined as the ratio between the tangential velocity of the tip of the blade and the incoming farfield axial velocity. It is expressed as

$$TSR = \frac{\Omega R_{\text{tip}}}{U_0}. \tag{2.32}$$

2.5.1 Prandtl Hub/Tip Correction

The basic momentum theory ignores the hub and tip vortices that occur due to the presence of limited number of blades. Various correction methods exist. Here the simple analytical expression developed by Prandtl is used. The correction factor can be composed by both the tip and hub correction factors. These are defined as

$$F_{\text{tip}} = \frac{2}{\pi}\arccos\left(\exp\left(-f_{\text{tip}}\right)\right) \qquad \text{and} \qquad F_{\text{hub}} = \frac{2}{\pi}\arccos\left(\exp\left(-f_{\text{hub}}\right)\right). \tag{2.33}$$

Here the auxiliary functions are defined as

$$f_{\text{tip}} = \frac{Z}{2}\left(\frac{R_{\text{tip}} - r}{r\sin\phi}\right) \qquad \text{and} \qquad f_{\text{hub}} = \frac{Z}{2}\left(\frac{r - R_{\text{hub}}}{R_{\text{hub}}\sin\phi}\right). \tag{2.34}$$

The total Prandtl correction factor is defined as

$$F = F_{\text{tip}} F_{\text{hub}}. \tag{2.35}$$

This factor is applied directly to the thrust and torque equations from the momentum theory as

$$\mathrm{d}T = 4F\pi\rho U_0^2 a\left(1-a\right) r\mathrm{d}r \tag{2.36}$$

$$\mathrm{d}Q = 4F\pi\rho U_0 \Omega a'\left(1-a\right) r^3\mathrm{d}r. \tag{2.37}$$

This inclusion translate to that the induction factors (Eq. 2.29) will carry the correction factor as

$$a = \frac{1}{\frac{4F\sin^2\phi}{\sigma C_n} + 1} \qquad \text{and} \qquad a' = \frac{1}{\frac{4F\sin\phi\cos\phi}{\sigma C_t} - 1}. \tag{2.38}$$

2.5.2 Propellers

In the case of propellers the BEM theory follows the same steps to reach an expression for the axial and tangential induction factors, a and a', as described so far. But as contrary to the effect of the turbine, in a propeller the the axial flow velocity U_0 accelerates as it passes through the rotor and the tangential speed induced after it points in the same direction of the blade rotation. Furthermore, as opposed to a turbine configuration, the static pressure increases after the rotor plane.

Figure 2.5 exhibits the velocity triangle encountered in a propeller airfoil section. As described the axial flow accelerates and the induced tangential velocity points in the same direction of rotation. The local angle of attack α is now defined as $\alpha = \beta - \phi$ instead.

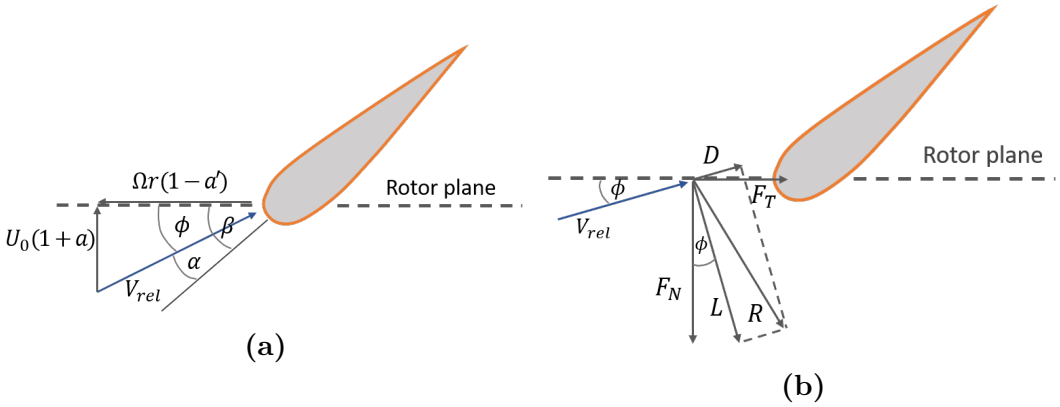


Figure 2.5: Propeller velocity triangle at the blade element

.

This configuration will only change some signs in the BEM equations that were derived for turbine mode. By introducing C as a variable that takes the value of 1 for a propeller and -1 for a turbine, the BEM equations can be written for any of propeller or turbine mode. The local angle of attack α can be written as

$$\alpha = C(\beta - \phi). \tag{2.39}$$

The velocity triangle for both modes can be expressed as

$$\tan(\phi) = \frac{U_0(1 + Ca)}{\Omega r(1 - Ca')}. \tag{2.40}$$

The force coefficients can be written as

$$C_n = C_l\cos\phi - CC_d\sin(\phi) \tag{2.41}$$

$$C_t = C_l\sin\phi + CC_d\cos(\phi). \tag{2.42}$$

Finally the induction factors expressions for any mode become

$$a = \frac{1}{\frac{4F\sin^2\phi}{\sigma C_n} - C} \qquad \text{and} \qquad a' = \frac{1}{\frac{4F\sin\phi\cos\phi}{\sigma C_t} + C}. \tag{2.43}$$

For propeller applications [3], results for different operating conditions are usually expressed with the advance ratio J which is defined as

$$J = \frac{U_0}{N_{\text{RPS}} D}. \tag{2.44}$$

The dimensionless coefficients of thrust and torque for propeller applications are defined as

$$C_T = \frac{T}{\rho N_{\text{RPS}}^2 D^4} \qquad \text{and} \qquad C_Q = \frac{Q}{\rho N_{\text{RPS}}^2 D^5}. \tag{2.45}$$

From where the power coefficient and efficiency of the propeller can be expressed as

$$C_P = 2\pi C_Q \qquad \text{and} \qquad \eta = \frac{C_T}{C_P} J. \tag{2.46}$$

2.5.3 Iteration Procedure

Since each control volume along the blade is assumed to be independent from the other, the solution at different blade elements are solved separately. For each blade element the classical fixed point iteration procedure follows these steps:

1. Initialize the induction factors a and a'.
2. Compute the flow angle ϕ from Eq. 2.40
3. Calculate the angle of attack α (Eq. 2.39) and the Re number (Eq. 2.22).
4. Read $C_l(\alpha, Re)$ and $C_d(\alpha, Re)$ from table lookup
5. Compute C_n and C_t from Eq 2.41.
6. Calculate a and a' with Eq. 2.43
7. Go back to Step 2 if a and a' have changed more than a certain tolerance value. Else stop iterating.
8. Calculate the loadings in the blade element Eq. 2.19 and 2.20

A faster and more reliable way of solving the BEM equations is rearranging the problem into a root-finding problem. In MATLAB the function `fzero` can be used for this purpose. According to Ning [9] a good choice to form the residual function for the root-finding problem is

$$f(\phi) = \frac{\sin(\phi)}{(1 + Ca)} - \frac{U_0 \cos(\phi)}{\Omega r(1 - Ca')} = 0. \tag{2.47}$$

Therefore the roots of this function can be found by varying the flow angle ϕ. Appendix A.2 displays the script used to perform this procedure. The sequence that is being followed internally with the help of `fzero` is:

1. Specify a flow angle ϕ Eq. 2.40
2. Calculate the angle of attack α (Eq. 2.39) and the Re number (Eq. 2.22).
3. Read $C_l(\alpha, Re)$ and $C_d(\alpha, Re)$ from table lookup
4. Compute C_n and C_t from Eq 2.41.
5. Calculate a and a' with Eq. 2.43.
6. Check the residual function Eq. 2.47
7. Go back to Step 1 if the residual function is not close enough to zero. Else stop.
8. Calculate the loadings in the blade element Eqs. 2.19 and 2.20.

The function `fzero` uses the Brent method as a root-finding algorithm to vary ϕ and reach the solution.

2.6 BEM Method Modifications

Applying the BEM method to the current axial CRPT configuration does require certain modifications. The main one given because the rotors are enclosed in a confined tunnel or pipe. In this case the cross sectional area, as well as the incoming volumetric flow, q, must remain constant. A good assumption in order to maintain this requirement is that, since $q = U_0/A$, the axial velocity will also remain constant and will not be accelerated nor slowed down as it results in a wind turbine (Figure 2.2) or propeller configuration.

In this sense the BEM method is reduced to solving only for the tangential induction factor a', whereas the axial induction factor a becomes zero. This wat the axial velocity at the rotor plane (as well as everywhere else) is constant and equal to the incoming axial velocity U_0.

From an angular momentum balance and from the blade element theory, the differential torque expressions considering $a = 0$ become

$$\mathrm{d}Q = 4F\pi\rho U_0 \Omega a' r^3 \mathrm{d}r \tag{2.48}$$

$$\mathrm{d}Q = \frac{1}{2}\rho Z c \frac{U_0 \Omega r(1 - Ca')}{\sin\phi\cos\phi} C_t r \mathrm{d}r. \tag{2.49}$$

Equating both this results together gives an expression for the tangential induction factor, which turns out to be the same as in equation 2.43, yielding

$$a' = \frac{1}{\dfrac{4F\sin\phi\cos\phi}{\sigma C_t} + C}. \tag{2.50}$$

Since the axial velocity remains constant, the resultant velocity triangle in both pump and turbine mode can be written as

$$\tan\phi = \frac{U_0}{\Omega r\,(1 - Ca')}. \tag{2.51}$$

Therefore the residual function, as proposed by Ning [9], becomes

$$\sin\phi - \frac{\cos\phi\ U_0}{\Omega r(1 - Ca')} = 0. \tag{2.52}$$

2.6.1 Velocity Triangles

Figure 2.6 exhibits the velocity triangles for the current blade profiles of the CRPT runners in pump mode. As seen the flow encounters Rotor 1 first and Rotor 2 after. As discussed in the previous section the incoming axial velocity U_0 is assumed constant.

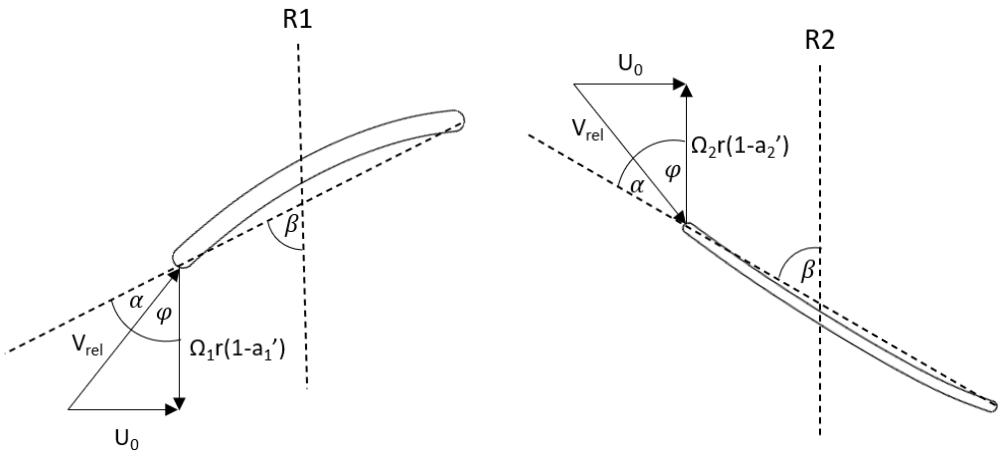


Figure 2.6: Schematic view velocity triangles pump mode.

Figure 2.7 shows the velocity triangles of the profiles for both runners in turbine mode. Here Rotor 2 comes first, Rotor 1 after and the flow is reversed as compared to pump mode.

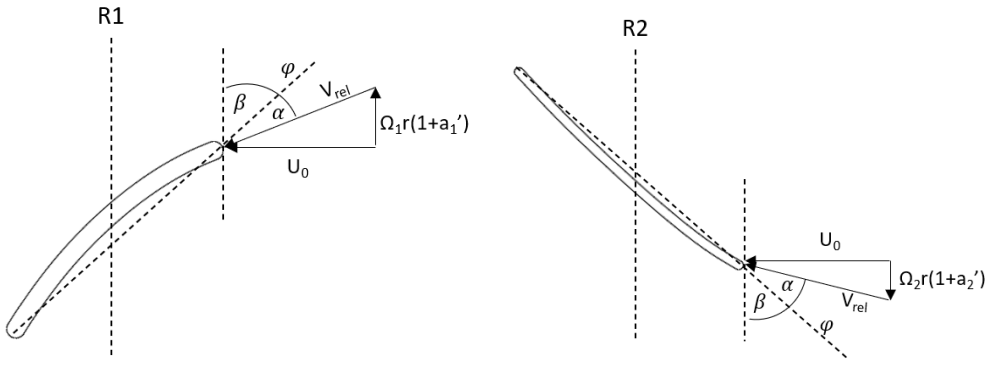


Figure 2.7: Schematic view velocity triangles turbine mode.

It is worth noticing that the rotational speeds, in both modes, are different from each other (Ω_1 and Ω_2) as the turbomachine allows this to happen. Furthermore both rotor planes have different tangential induction factors (a_1' and a_2'), which become the unknowns of this problem.

2.6.2 Additional Pressure Correction

The classical BEM method considers there is a static pressure discontinuity across the rotor plane. Nevertheless, this pressure drop (turbine) or jump (propeller) recovers back to the same ambient pressure as the far upstream as it happens for wind turbines or propeller configurations. For the case of an axial water pump or turbine the static pressure discontinuity that happens across the runner is maintained and does not recover to ambient pressure. Therefore different static pressure values can be measured far upstream and downstream, creating a constant pressure difference.

For the case of a single rotor pump enclosed in a pipe, the total pressure before and after the rotor, $P_{\text{tot},0}$ and $P_{\text{tot},1}$, can be written as

$$\text{Upstream: } P_{\text{tot},0} = p^- + \frac{1}{2}\rho U_0^2 \tag{2.53}$$

$$\text{Downstream: } P_{\text{tot},1} = p^+ + \frac{1}{2}\rho U_0^2 + \frac{1}{2}\rho u_\theta^2. \tag{2.54}$$

Here, u_θ is the tangential velocity the flow has gained after the rotor plane and U_0 is maintained constant. The radial component of velocity is thought to be negligible. Subtracting these expressions, yields

$$P_{\text{tot},1} - P_{\text{tot},0} = \rho g(H_1 - H_0) = \Delta p + \frac{1}{2}\rho u_\theta^2. \tag{2.55}$$

Here, $\Delta p = p^+ - p^-$. Therefore, the static pressure difference across the rotor plane in pump mode can be written as

$$\Delta p = \rho g \Delta H - \frac{1}{2}\rho u_\theta^2. \tag{2.56}$$

Here ΔH represents the total head difference. By performing the same balance before and after a rotor in turbine mode, the static pressure difference across the runner is expressed as

$$\Delta p = \rho g \Delta H + \frac{1}{2}\rho u_\theta^2. \tag{2.57}$$

By applying the extra pressure difference on the twisted blade element we get an extra contribution for thrust and torque, as

$$dT = \Delta p_{\text{rot}} \cos\beta \ \mathrm{d}A = \Delta p \cos\beta \ c \ \mathrm{d}r \tag{2.58}$$

$$dQ = \Delta p_{\text{rot}} \sin\beta r \ \mathrm{d}A = \Delta p \sin\beta \ c \ r \ \mathrm{d}r. \tag{2.59}$$

Here c is the chord length of the blade element and β the twist angle.

2.6.3 Downstream Rotor

In BEM theory the induced tangential velocity on the flow experiences a sudden jump that goes from $u_\theta = 0$ before the rotor to $u_\theta = 2a'\Omega r$ right after it, independent of the operational mode. Therefore, as seen in the velocity triangles at the rotor

plane (Figures 2.6 and 2.7), the tangential velocity of the flow is thought to be $u_\theta = a'\Omega r$ in between. This represents a linear interpolation approach.

Unlike for the upstream runner where the flow does not come with any swirl, for the downstream runner the flow will come with the tangential velocity that has been induced by the previous upstream rotor. In order to find an expression for the downstream induction factor a'_d, that will let us calculate the loadings at the downstream rotor, we can linearly interpolate using the information from the upstream swirl.

Recalling the interpolation formula for a value $y(x)$, where $x_a < x < x_b$, yields

$$y = y_a + (y_b - y_a)\frac{x - x_a}{x_b - x_a}. \tag{2.60}$$

Replacing the y values with the tangential velocities before, after and at the rotor plane ($u_{\theta,\mathrm{u}}$, $u_{\theta,\mathrm{d}}$ and $u_{\theta,\mathrm{R}}$), and assuming equidistant length between this locations, gives

$$u_{\theta,\mathrm{R}} = u_{\theta,\mathrm{u}} + \frac{(u_{\theta,\mathrm{d}} - u_{\theta,\mathrm{u}})}{2}. \tag{2.61}$$

The tangential velocity after the downstream rotor can be expressed as the tangential velocity upstream of that rotor plus some differential that will be induced by that runner, $u_{\theta,\mathrm{d}} = u_{\theta,\mathrm{u}} + \Delta u_\theta$. Replacing in the previous equation, gives

$$u_{\theta,\mathrm{R}} = u_{\theta,\mathrm{u}} + \frac{\Delta u_\theta}{2}. \tag{2.62}$$

Replacing the upstream tangential velocity by $u_{\theta,\mathrm{u}} = -2a'_\mathrm{u}\Omega_\mathrm{u} r$ and the tangential velocity at the downstream rotor by $u_{\theta,\mathrm{R}} = a'_\mathrm{d}\Omega_\mathrm{d} r$ the following is obtained:

$$a'_\mathrm{d}\Omega_\mathrm{d} r = -2a'_\mathrm{u}\Omega_\mathrm{u} r + \frac{\Delta u_\theta}{2}. \tag{2.63}$$

Solving for the tangential velocity differential across the rotor Δu_θ leads to

$$\Delta u_\theta = 4a'_\mathrm{u}\Omega_\mathrm{u} r + 2a'_\mathrm{d}\Omega_\mathrm{d} r. \tag{2.64}$$

Using this tangential velocity differential in the balance of angular momentum of an annular ring element, yields

$$\mathrm{d}Q_\mathrm{d} = r\Delta u_\theta \mathrm{d}\dot{m} = r\left(4a'_\mathrm{u}\Omega_\mathrm{u} r + 2a'_\mathrm{d}\Omega_\mathrm{d} r\right)\underbrace{\rho U_0 2\pi r \mathrm{d}r}_{\mathrm{d}\dot{m}}. \tag{2.65}$$

Equating this expression with the blade element theory, gives

$$\underbrace{4\rho\pi U_0\left(2a'_\mathrm{u}\Omega_\mathrm{u} + a'_\mathrm{d}\Omega_\mathrm{d}\right) r^3 \mathrm{d}r}_{\text{Momentum balance}} = \underbrace{\rho\sigma\pi r \frac{U_0\Omega_\mathrm{d} r(1 - Ca'_\mathrm{d})}{\sin\phi\cos\phi} C_t \mathrm{d}r}_{\text{Blade Element}}. \tag{2.66}$$

Solving for the induction factor of the downstream rotor, yields

$$a'_\mathrm{d} = \frac{2k'a'_\mathrm{u}\frac{\Omega_\mathrm{u}}{\Omega_\mathrm{d}} - 1}{-k' - C}. \tag{2.67}$$

Here $C = 1$ for pump mode and $C = -1$ for turbine mode, with the auxiliary term being

$$k' = \frac{4F \sin \phi_{\mathrm{d}} \cos \phi_{\mathrm{d}}}{\sigma C_t}. \tag{2.68}$$

The pressure correction proposed in section 2.6.2 needs to take into account the upstream swirl in the Bernoulli equation.

$$\Delta p_{\mathrm{d}} = \rho g \Delta H_{\mathrm{d}} - C \frac{1}{2} \rho \left(u_{\theta,\mathrm{u}}^2 - u_{\theta,\mathrm{d}}^2 \right), \tag{2.69}$$

where

$$u_{\theta,\mathrm{u}}^2 - u_{\theta,\mathrm{d}}^2 = u_{\theta,\mathrm{u}}^2 - \left(u_{\theta,\mathrm{u}} + \Delta u_\theta \right)^2 = 2 u_{\theta,\mathrm{u}} \Delta u_\theta + \Delta u_\theta^2. \tag{2.70}$$

2.7 Polar Curve Extrapolation

In order to implement the BEM method, airfoil characteristics of C_l and C_d need to be computed. The CFD simulations would only provide this coefficient for a limited range of AoA, therefore it is necessary to extrapolate the initial data in order to obtain the lift and drag coefficients for the whole spectrum of angles of attack, AoA. This is necessary in order to allow the BEM-code to eventually access this data if needed during the iteration procedure.

In the present study the Viterna method is used for data extrapolation. The method uses the following formulation to extrapolate lift and drag coefficients after stall, where

$$C_l = A_1 \sin 2\alpha + A_2 \frac{\cos^2 \alpha}{\sin \alpha} \tag{2.71}$$

$$C_d = B_1 \sin^2 \alpha + B_2 \cos \alpha. \tag{2.72}$$

This particular formulation is used from the stalling angle of attack up until $\alpha = 90^{\mathrm{o}}$ to extrapolate. To complete the values at $\alpha > 90$ and $\alpha < \alpha_{\min}$ the extrapolation performed is reflected. According to Mahmuddin et al. [8], even though the method computation results are not an accurate representation of the true physics, it provides a reasonable estimate for early in the design process.

2.8 Turbulence Model

The turbulence model used for the simulations of the flow past the 2D blade profiles is Spalart - Allmaras one equation model. This model can be represented by

$$\frac{\partial \tilde{\nu}}{\partial t} + u_j \frac{\partial \tilde{\nu}}{\partial x_j} = c_{b1}(1 - f_{t2}) \tilde{S} \tilde{\nu} + \frac{1}{\sigma} \left[\frac{\partial}{\partial t} \left((\nu + \tilde{\nu}) \frac{\partial \tilde{\nu}}{\partial x_j} \right) + c_{b2} \frac{\partial \tilde{\nu}}{\partial x_i} \frac{\partial \tilde{\nu}}{\partial x_i} \right] - \left[c_{w1} f_w - \frac{c_{b1}}{\kappa} f_{t2} \right] \left(\frac{\tilde{\nu}}{d} \right)^2. \tag{2.73}$$

According to the turbulence modelling resource from NASA [14], the boundary conditions for the viscosity-like variable $\tilde{\nu}$ and turbulent kinematic viscosity ν at the wall and farfield are

$$\tilde{\nu}_{\text{wall}} = 0 \qquad \tilde{\nu}_{\text{farfield}} = 3\ \nu_\infty \text{ to } 5\ \nu_\infty \tag{2.74}$$

$$\nu_{\text{wall}} = 0 \qquad \nu_{\text{farfield}} = 0.210438\ \nu_\infty \text{ to } 1.294234\ \nu_\infty. \tag{2.75}$$

Here ν_∞ is the water kinematic viscosity at the farfield. In this case the lowest range values for $\tilde{\nu}$ and ν at the farfield boundary were chosen as boundary conditions for the steady-state flow simulations past the blade profiles.

The turbulence model used for the CFD simulations of the turbomachine is the two equation model $SST\ k-\omega$. This model is represented as

$$\frac{\partial k}{\partial t} + u_j \frac{\partial k}{\partial x_j} = P_k - \beta^* \omega + \frac{\partial}{\partial x_j}\left[(\nu + \sigma_k \nu_T)\frac{\partial k}{\partial x_j}\right] \tag{2.76}$$

$$\frac{\partial \omega}{\partial t} + u_j \frac{\partial \omega}{\partial x_j} = \alpha S^2 - \beta \omega^2 + \frac{\partial}{\partial x_j}\left[(\nu + \sigma_\omega \nu_T)\frac{\partial \omega}{\partial x_j}\right] + 2(1-F_1)\sigma_{\omega 2}\frac{1}{\omega}\frac{\partial k}{\partial x_i}\frac{\partial \omega}{\partial x_i}. \tag{2.77}$$

Here the kinematic eddy viscosity is: $\nu_T = \dfrac{a_1 k}{\max(a_1 \omega, SF_2)}$.

No changes are made in any of the turbulence models coefficients default values.

2.9 OpenFOAM

As its name indicates OpenFOAM (Open Source Field Operation And Manipulation) is an open source software mainly used for CFD applications.

The blade profile simulations that create the data of C_l and C_d needed for the BEM method are performed with OpenFOAM v1912. The CRPT validation cases are run using FOAM-extend-4.1. This version of the software has some features that are not available in other OpenFOAM versions, such as e.g. GGI interface, mixingPlane, among others.

2.9.1 `simpleFoam` solver

This is a RANS solver for incompressible steady-state simulations of turbulent flow. The pressure-velocity coupling is solved using the SIMPLE algorithm. This solver is the one used to simulate the flow around the given rotor blade profiles to obtain lift and drag coefficients.

2.9.2 `MRFSimpleFoam` solver

This solver is for steady-state, incompressible flows with multiple rotating frames of reference regions. It is a much faster and less complex approach than to solve using a moving mesh in a transient-state. `MRFSimpleFoam` is used for the CFD simulations that serve as validating cases for the BEM method.

2.9.3 `profile1DfixedValue` boundary condition

This boundary condition implements a cylindrical field defined by a 1D fixed value profile that could be along the radial or vertical direction for a typical RANS computation [11]. In the current study, this boundary condition is used to define the inlet of the downstream rotor in both operating modes for the individual geometries (described later) of both runners. Appendix C.2 shows the inputs used for this boundary condition. Appendix C.3 displays how the boundary conditions assigned.

3
Methodology

3.1 Blade Profile Representation

For the BEM method to be implemented it is necessary to characterize the rotor blades as a collection of 2D profiles stacked at different radial positions. Figure 3.1 shows both rotor individual blades in a 3D view representation.

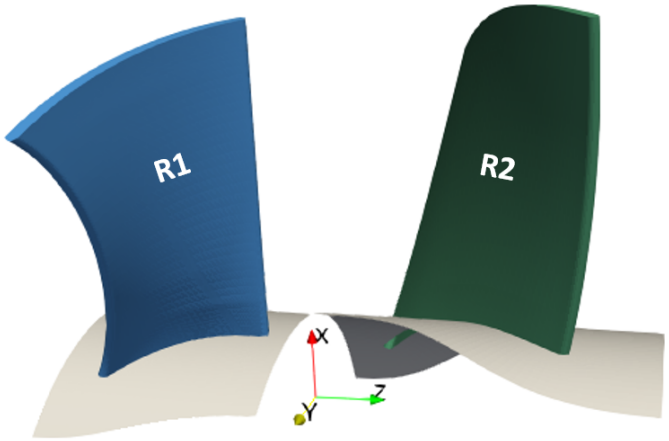
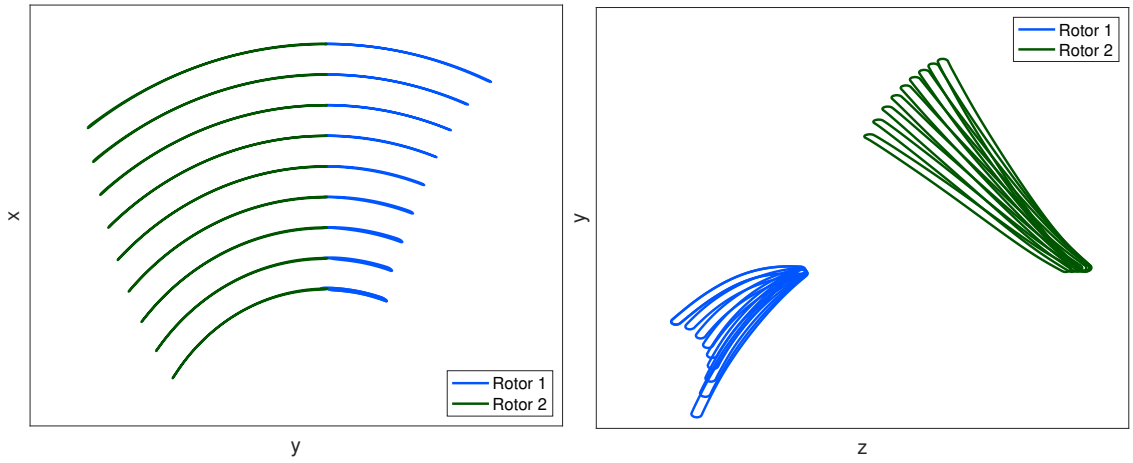


Figure 3.1: View of individual rotor blades.

The only blade geometric data available are input files that the meshing software Ansys TurboGrid reads to create the shape of the blades, hub and shroud. These are `.crv` files that separate each individual rotor blade into 9 different profiles along the radial direction. Profile 1 is located close to the hub and profile 9 close to the shroud. Figure 3.2 displays the profiles shapes that are formed from the points contained in the `.crv` files mentioned. The geometric information is in cartesian coordinates. The front view shows how the blades sit on the cylindrical hub and how the profiles are stacked one on top of the other from hub to shroud. The top view shows how each profiles has a different shape, chord length and is twisted differently.

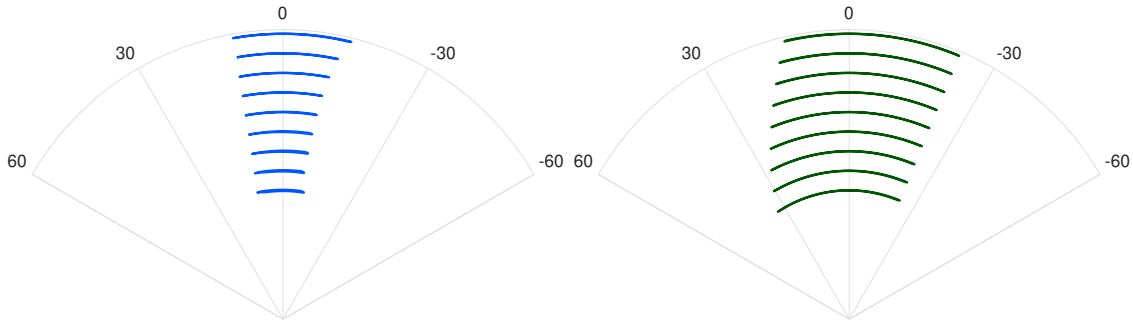


(a) Front view **(b)** Top view

Figure 3.2: Rotor blades profiles different views, cartesian coordinates.

As seen in Figure 3.2a the different profiles have a curved shape. Also both blades are not aligned at the same azimuth angle on top of the hub. This is why obtaining 2D profiles along the radial direction becomes a bit more cumbersome. Projecting the curved shapes as they are already shown in Figure 3.2b would result in a less accurate 2D representation than if the blade would be aligned at the zero azimuth angle. Therefore, it is necessary first to rotate both blades around its rotation axis, before taking the 2D projections in the yz axis.

By keeping the z-coordinate intact the cartesian coordinates from Figure 3.2a can be transformed to cylindrical coordinates. Then, all blade profiles can be rotated a certain angle around the rotation axis to align them better with the zero azimuth angle. Figure 3.3 displays both runner blades profiles rotated and in cylindrical coordinates.



(a) Rotor 1 **(b)** Rotor 2

Figure 3.3: Rotated blade profiles, front view, cylindrical coordinates.

To project the profiles and obtain the 2D representations as seen from a top view like in Figure 3.2b it is necessary to transform the coordinates from cylindrical back to cartesian. The projected profiles are shown in Figure 3.4.

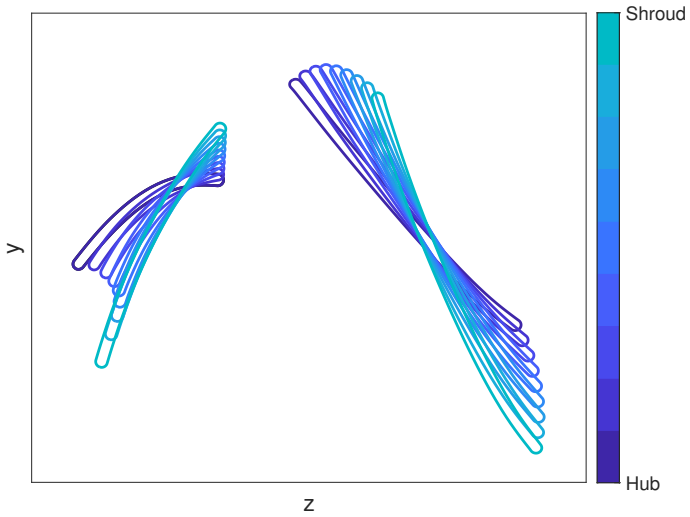


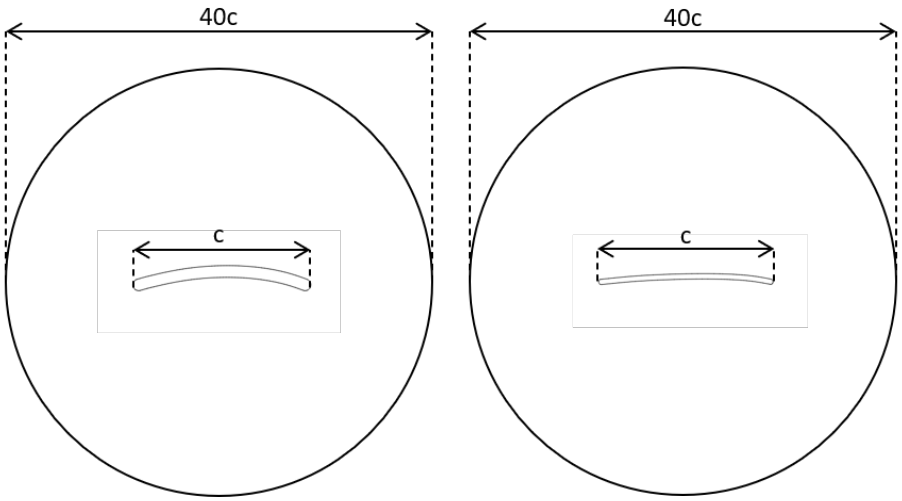
Figure 3.4: Profiles shapes 2D representations

It is seen that each blade profile is twisted differently along the radial direction. The twist, β, as mentioned previously is the angle between the chord line that unites the leading edge with the trailing edge of the 2D profile, and the plane of rotation. In this case the profiles are more twisted towards the hub and less twisted towards the shroud. It is also shown how the profiles have different chord lengths.

3.1.1 Blade Profiles Mesh

Next step is to transfer the profile shapes to a meshing software, but before that these need to be normalized by their chord length and untwisted so they look horizontal. The software used to create a mesh around each profile is Ansys ICEM CFD. Given that each blade is characterized by 9 different profiles the current effort required creating 18 different grids. Luckily, ICEM CFD allows to record a meshing session that outputs a script of the process. This fact allowed to partially automate the meshing procedure. Nevertheless, since each profile is slightly different, a lot of manual intervention was needed.

Given that the current turbomachine accepts flow in both axial directions, it is necessary to have a computational domain which allows the same level of detail irrespective of the flow direction.

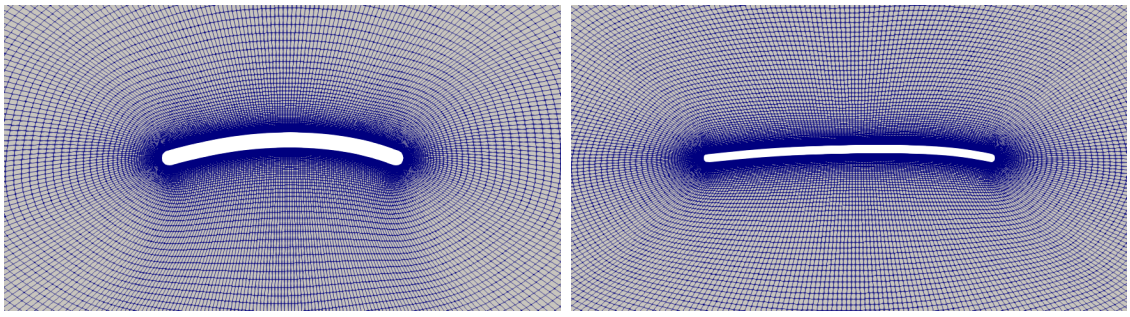


(a) Rotor 1 **(b)** Rotor 2

Figure 3.5: Computational domain for simulation of profile characteristics.

Figure 3.5 shows the computational domain chosen to mesh the different blade profiles. The circular shape of the grid allows for equal mesh density regardless of the flow direction. The radius of the circular domain is 20 times the chord length of the profile, which is 1 because they are normalized. This ensures that the farfield remains distant from the airfoil-like profile to avoid suppressing the flow field variations induced by the body.

Figure 3.6 displays a detail of the grid around the mid-height profile of both runner blades. The last step in the meshing process was to apply a Laplace smoothing procedure to the grid so non-orthogonality could be improved as well as other quality factors.



(a) Rotor 1 **(b)** Rotor 2

Figure 3.6: Mesh detail of mid-height profiles (profiles 5)

The number of cells in each mesh for each profile is $\sim$ 55500. The y^+ value varies according to each simulation. Nevertheless by looking at a high AoA and Re number simulation it can be said that overall the average of this value is not larger than $y^+ \sim 4$ for every simulation.

3.1.2 Blade Profiles Polar Curves

Usually a wind turbine blade is composed by previously known 2D airfoil shapes (e.g. NACA type airfoil). Since the current blade profiles are not documented airfoils shapes, it is necessary to create the information of C_l and C_d at different AoA needed in BEM for each one of the 9 profiles conforming each blade.

Figures 3.7 and 3.8 show the aerodynamic characteristics of the airfoil-type blade profiles. Each point in those curves represents one CFD steady-state simulation, where the incoming flow has a certain angle of attack, AoA, with respect to the chord line of the profile as shown in Figure 3.6. This effort required heavy scripting to create, organize and post-process each case.

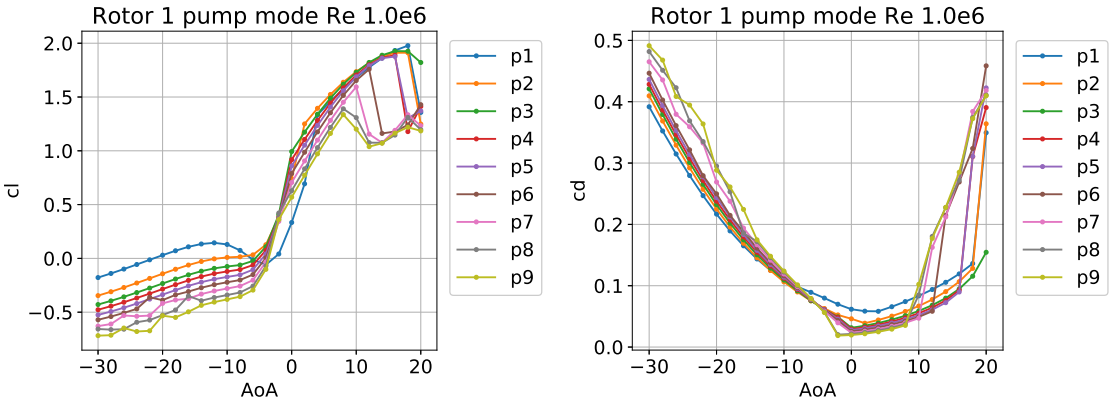


Figure 3.7: Rotor 1 profile characteristics in pump mode

As the angle of attack goes to the extremes of high and low values, the steadiness of the solution from the CFD simulations decreases. For higher or lower AoA the lift and drag coefficients values oscillate around a certain value. Therefore, AoA higher or lower than shown in Figures 3.7 and 3.8 cannot be simulated as unsteadiness of the coefficients throughout the iterations is encountered.

The total lift and drag coefficients from the resultant pressure and viscous forces exerted on the blade profiles by the flow were obtained by using the OpenFOAM function object `forceCoeffs`. For every different angle of attack, there is a different lift and drag direction that needs to be modified in the `controlDict` file, where the function object inputs are specified. An example of a script that performs this procedure is shown in Appendix C.1.

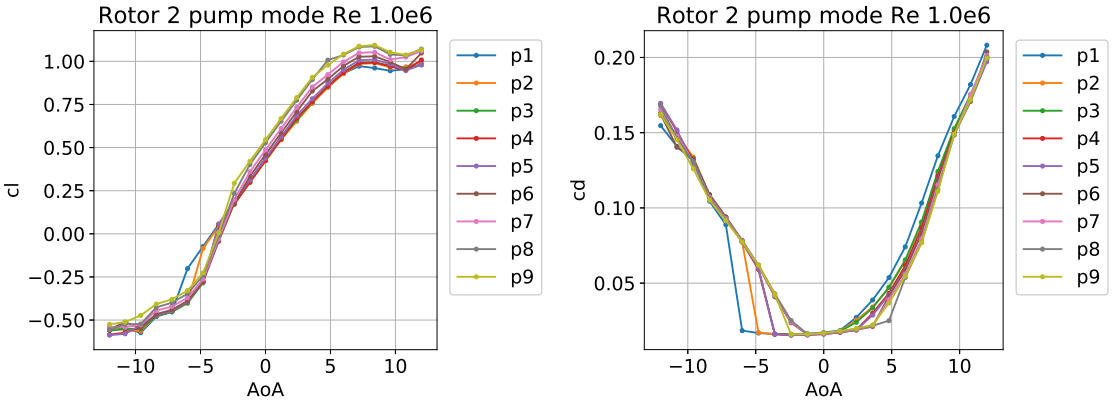


Figure 3.8: Rotor 2 profile characteristics in pump mode

As mentioned, the same kind of information shown in Figures 3.7 and 3.8 needs to be obtained for when the flow comes in the other direction, i.e in turbine mode. If the airfoil-type profiles were precisely symmetric it would not be necessary to revert the flow because the C_l and C_d curves would be the same. All results in both operating modes are shown in Appendix B. In total 3 different Re numbers where run, $Re = 0.5e-6$, $Re = 1.0e-6$ and $Re = 2.0e-6$, which represent 2538 different single simulations.

In order to implement the BEM method it is desirable to have the complete range of C_l and C_d for all possible AoA. This is needed mainly so the BEM-code can work for any input of incoming velocity and rotation speed of the rotor like e.g. off-design conditions were high AoA above stall or low AoA may be encountered. The extrapolated curves are also shown in Appendix B.

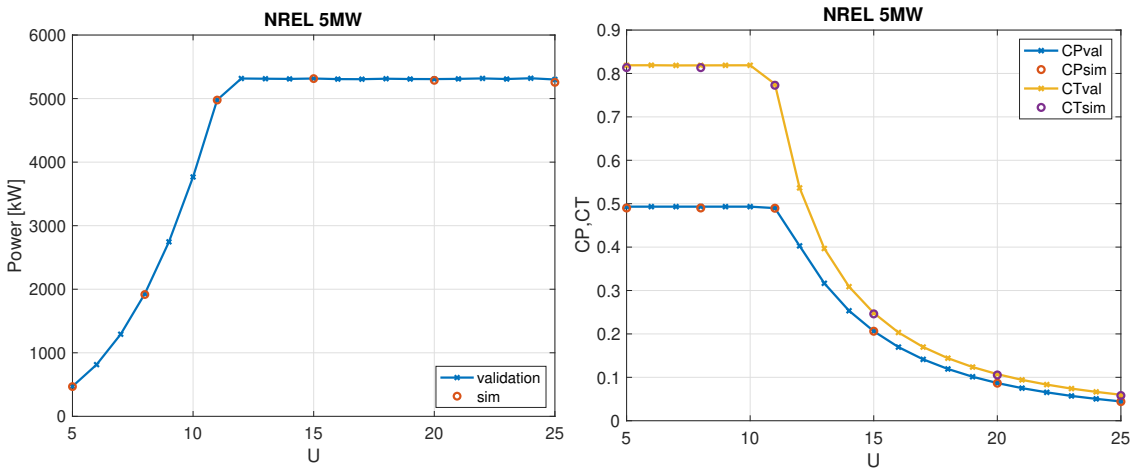
3.2 Working BEM Code

The BEM-based algorithm is written in MATLAB. Validating examples of both turbine and propellers are carried out to ensure that the code works properly. These validating cases do not implement the modifications proposed to the BEM method in section 2.6 since they are not required.

The first validation case correspond to data from a very precise BEM simulation of the NREL 5 MW wind turbine machine. This machine corresponds to a conventional 3 bladed offshore wind turbine. Its detailed specifications are well documented and the case is often used as a baseline to investigate a representative typical multi megawatt land- or sea-based wind turbine [7].

Figure 3.9 displays the results obtained by the written BEM code for the NREL 5 MW wind turbine as compared to the validation data. The first figure shows the total power obtained vs the different freestream incoming axial velocities. The second figure exhibits the total power and thrust coefficient as expressed in Eq.

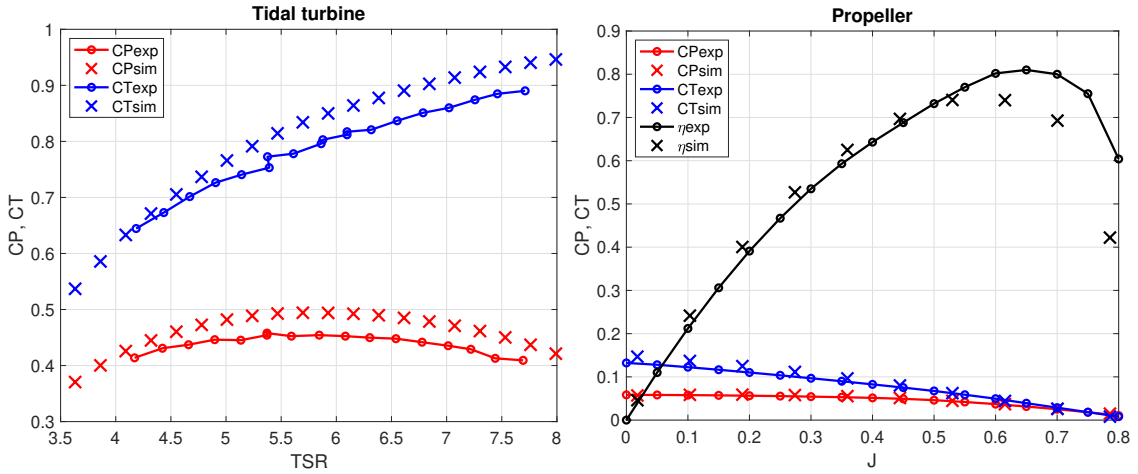
2.31. It is shown that the simulated cases (circle marker) agree very well with the validation data.



(a) (b)

Figure 3.9: Power generated and thrust and power coefficient vs incoming air velocity, NREL 5MW wind turbine.

Another validation case for the BEM-based code was performed on a marine tidal turbine. The available validating results were obtained from measurements performed in a towing tank [2]. Furthermore the code was assessed for a propeller configuration, where the validating data was also obtained from experiments [13]. Both comparisons are displayed in Figure 3.10. For the tidal turbine the thrust and power coefficient, and the tip speed ratio are defined in Eqs. 2.31 and 2.32. For the propeller configuration the advance ratio J goes in the x-axis and the dimensionless coefficients from Eqs. 2.45 and 2.46 are used.



(a) (b)

Figure 3.10: Air propeller validation [13]. Marine tidal turbine validation [2]

Since the validation cases from Figure 3.10 correspond to experimental data obtained

from experimental measurements it can be said that the BEM method provides a reasonable amount of error and a good approximation.

3.3 CRPT Validation Cases

As mentioned previously validation data for the loadings on the CRPT blades is obtained by means of CFD simulations using OpenFOAM. The version used for these cases is FOAM-extend 4.1. As said this version allows for the use of e.g `mixingPlane` interface. The solver used is `MRFSimpleFoam` for steady-state.

For validation 3 different geometries were provided. One corresponds to a coupled case set-up were both runners are placed together as the real CRPT would be. The solution of this case would be coupled and the blade loadings would be additionally affected by the other runner operation. The other two geometries are individual case set-up were both Runner 1 and Runner 2 are placed individually in a guiding pipe or tunnel.

Coupled runners set-up

Figure 3.11 displays the full case geometry with both runners placed next to the other and separated by a given distance. The interface located in-between both rotors is a `mixingPlane` interface. This interface computes and transfers circumferential average values. Since Rotor 1 has 8 blades and Rotor 2 has 7 blades periodicity of the runners requires a cyclic GGI type of boundary conditions on the sides. This way one blade passage can be simulated for each runner in order to save computational resources.

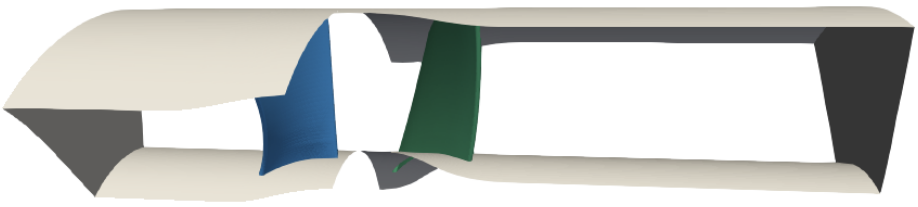


Figure 3.11: Runner 1 and 2 in a coupled case set-up

Individual runners set-up

Given that the BEM method implementation does not differentiates if the upstream rotor has a coupled downstream counter-rotating rotor after it or not, it is decided that in order to better compare and analyse results with the BEM method, individual rotor geometries would be of use. Figure 3.12 shows both individual runners

computational domain. Similar to the coupled set-up only one blade passage is simulated with cyclic boundary conditions.

(a) Runner 1 **(b)** Runner 2

Figure 3.12: Runners individual case set-up geometry

3.4 Blade Element Radial Positions

The BEM method needs to be applied at certain radial locations on the blade which will define the blade elements. Figure 3.13 displays the radial positions along the blades of the CRPT runners in the radial direction. The blade is divided into ten equally distributed blade elements for BEM simulation purposes. The dashed red lines represent the limits of each blade element which in turn are defined by the radial points locations.

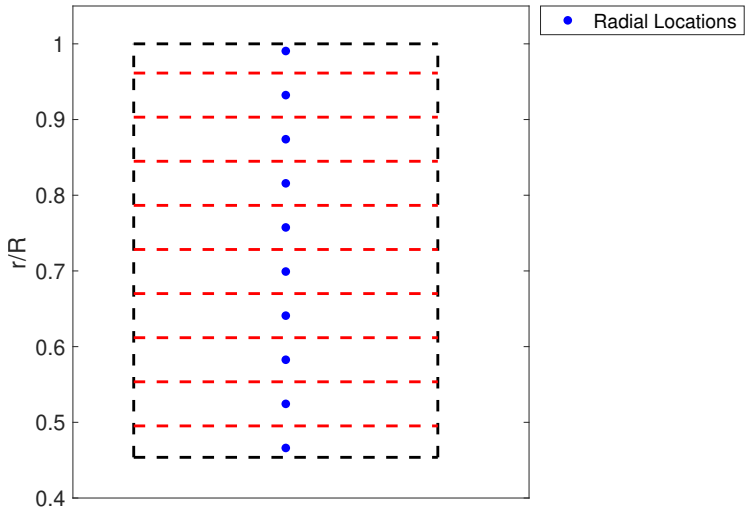


Figure 3.13: Radial points and blade elements for both rotor blades of the CRPT

4

Results and Discussion

In the this chapter a comparison between the blade loadings obtained from the CRPT steady-state simulations in OpenFOAM and the blade loadings predicted by BEM is shown. The operating conditions ran consider that Runner 1 rotates faster than Runner 2 in both pump and turbine mode. Specifically, Runner 2 has a rotational speed that represents 75% of the rotational speed of Runner 1 in all different cases simulated.

Positive values of torque, Q, and thrust, T, in any operating mode and in any runner represent the direction that loading should have in a normal operating condition. If for example, the torque is negative (i.e. negative power) in one of the runners in turbine mode, this means that runner requires power instead of generating it. This would of course be an inefficient operating condition. Same principle applies for pump mode, were a negative torque implies power generated instead of required.

Results in this chapter for both pump and turbine mode are expressed in the dimensionless coefficients commonly used in wind industry. These correspond to thrust and power coefficient, C_T and C_P from Eq. 2.31. The tip speed ratio, TSR, (Eq. 2.32) is depicted in the graphs' x-axis. This type of representation allows for an easier way of visualising and comparing the data obtained.

The BEM results shown have different labels. BEM label stands for the regular BEM-equations applied including the Prandtl correction factor regularly used for tip and hub vortex losses. The label $CORR$ represents the results of the extra loading (on top of the BEM results) due to the pressure difference as proposed in section 2.6.2. In a similar way label BEM_noF omits the F correction factor (section 2.5.1) in order to understand the implications of this factor on the results. Lastly, $CORR\_noF$, represents the extra loading on the blade due to the pressure difference over the BEM_noF results. The $\_noF$ subscript for the downstream rotor means deactivating F factor for that rotor, but also using the upstream information with no F factor from the upstream rotor.

4.1 Pump Mode

As mentioned previously, in pump mode the flow faces Runner 1 first and Runner 2 second. Figure 4.1 displays an schematic representation where the flow direction and rotation direction of the runners is shown. In this mode, the flow goes in the positive direction of the z-axis. Furthermore, Runner 1 rotates clockwise and Runner 2 in the counterclockwise direction around z-axis.

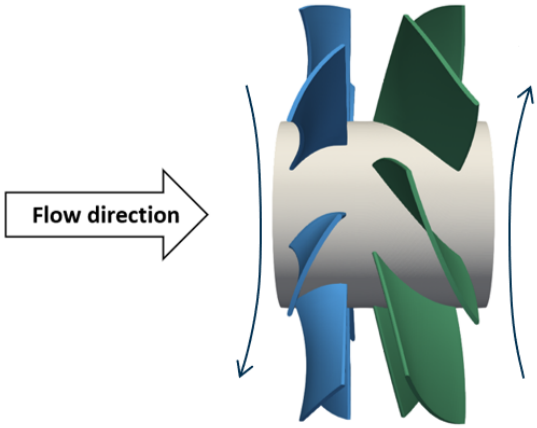


Figure 4.1: Flow direction and direction of rotation of runners in pump mode

4.1.1 Coupled runners set-up

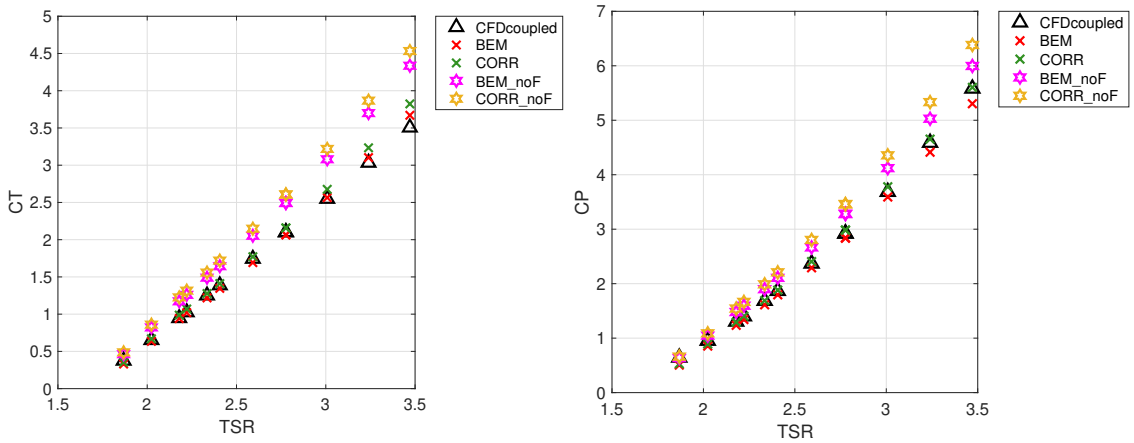
Table 4.1 displays the values obtained for individual pressure head drop ΔH and efficiencies, η, for both rotors in a coupled set up in pump mode. It can be seen that for some operating conditions the head drop across the downstream Runner 2 is negative which means that rotor would be acting as a pump. These cases can be regarded as non-feasible.

Table 4.1: Individual head rise and efficiencies in pump mode

q [m^3/s]	N1 [RPM]	N2 [RPM]	dH_1 [m]	dH_2 [m]	η_1 [%]	η_2 [%]
0.296	1200	900	5.22	3.31	90.87	79.78
0.296	1300	975	6.51	4.59	89.68	77.86
0.296	1400	1050	7.94	5.96	87.94	74.15
0.296	1500	1125	9.47	6.97	86.23	67.85
0.37	1200	900	3.84	0.21	89.11	17.54
0.37	1300	975	5.23	1.91	91.01	68.01
0.37	1400	1050	6.65	3.6	91.34	78.37
0.37	1500	1125	8.18	5.2	90.97	79.92
0.44	1200	900	2.19	-4.23	78.5	-
0.44	1300	975	3.5	-2.19	84.4	-
0.44	1400	1050	5.03	-0.24	88.81	-
0.44	1500	1125	6.64	1.79	90.64	58.11

Rotor 1 upstream

Figure 4.2 displays the thrust and power coefficients, C_T and C_P, vs the tip speed ratio, TSR, obtained from results for the upstream Rotor 1. As mentioned, this coefficients are the ones usually used for wind turbines (section 2.5). It can be seen that there is certain tendency or behaviour where for high TSR there is larger blade loading. A high TSR value means the rotor is rotating fast as compared to the incoming freestream velocity.



(a) **(b)**

Figure 4.2: C_T and C_P vs TSR Rotor 1 pump mode

In both figure a better agreement for total thrust and power can be seen for BEM simulations which use the correction factor F. Furthermore omitting this correction factor increases the total loading on the blade.

Rotor 2 downstream

Figure 4.3 displays the thrust and power coefficients, C_T a C_P, vs TSR for the downstream Rotor 2 in pump mode. It can be seen that all different BEM simulation over-predict thrust and torque (i.e. power) as compared to the CFD results. Also no major difference between omitting or maintaining the F factor is seen. It is worth recalling that for the downstream rotor the proposed Eq. 2.67 is being used. Good agreement with the trend of results is seen.

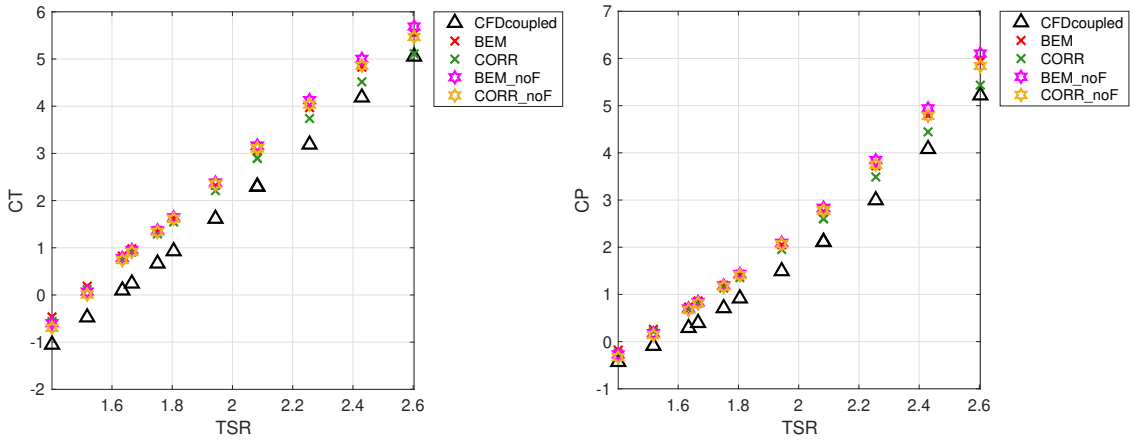
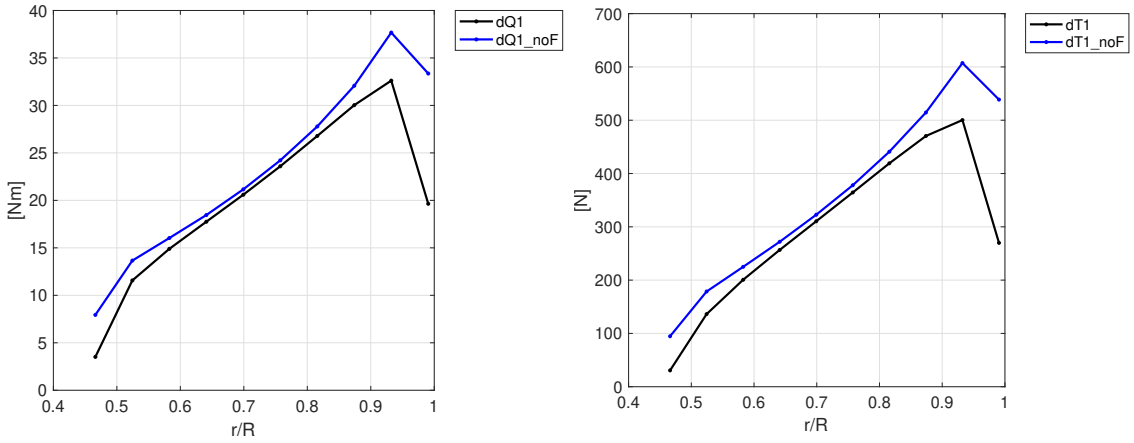


Figure 4.3: C_T and C_P vs TSR Rotor 2 pump mode

Case: q = 0.37 m^3/s, N_1 = 1500 RPM, N_2 = 1125 RPM

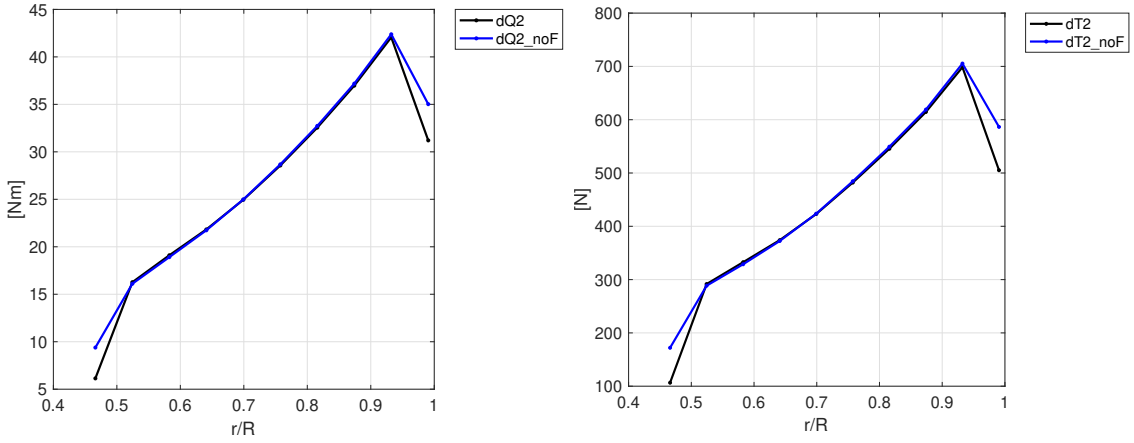
Figure 4.4 displays the torque, dQ ,and thrust, dT, at the blade elements (Fig. 3.13) of the upstream Runner 1 computed by BEM for the specific case. It is worth mentioning that the total thrust and torque on the blade is equal to the sum of all blade element contributions. It is shown that the larger load contributions comes from the blade elements close to the tip of the blade as compared to close to the hub. Also there is a decline in loading as the blade element gets closer to the tip.



(a) (b)

Figure 4.4: dQ and dT contribution along upstream Rotor 1 blade calculated by BEM, pump mode, specific case

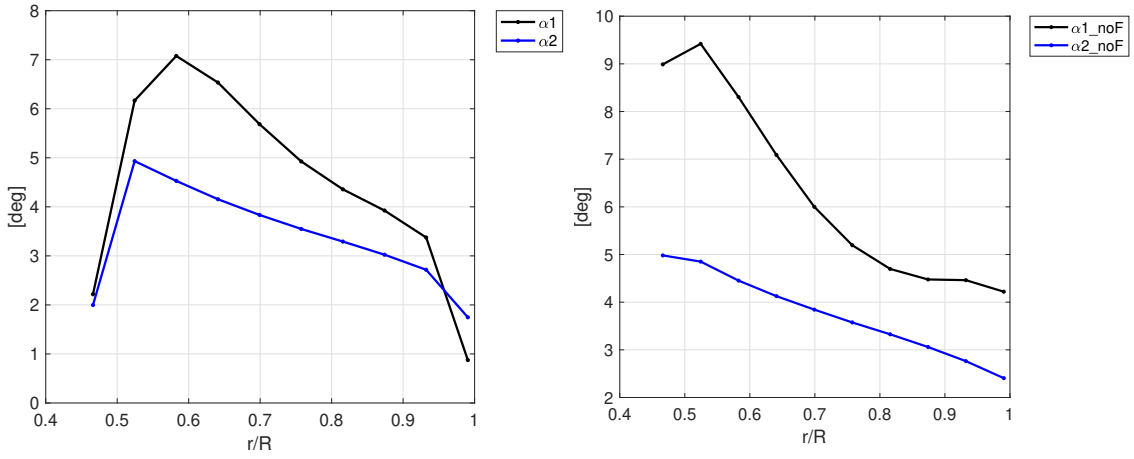
Furthermore, it can be seen that when the F correction factor is active both dQ and dT are smaller in comparison to when no F factor (i.e. $F = 1$ everywhere) is present. Figure 4.5 shows the same blade element contribution as the previous figure, but for the downstream Rotor 2. A smaller difference can be seen between deactivating or maintaining the F factor for the downstream Rotor 2.



(a) (b)

Figure 4.5: dQ and dT contribution along downstream Rotor 2 blade calculated by BEM, pump mode, specific case

Figure 4.6 displays the angle of attack, α, computed by BEM in each blade element along the radial direction for both rotors. Figure 4.6a shows a comparison between AoA obtained for Rotor 1 and Rotor 2 with the F factor active. Figure 4.6b shows the same variable with no F factor considered. It is shown that for this particular operating condition the angle of attack calculated by BEM is lower in the downstream rotor than in upstream one.



(a) **(b)**

Figure 4.6: *AoA* along rotor blades predicted by BEM, pump mode, specific case

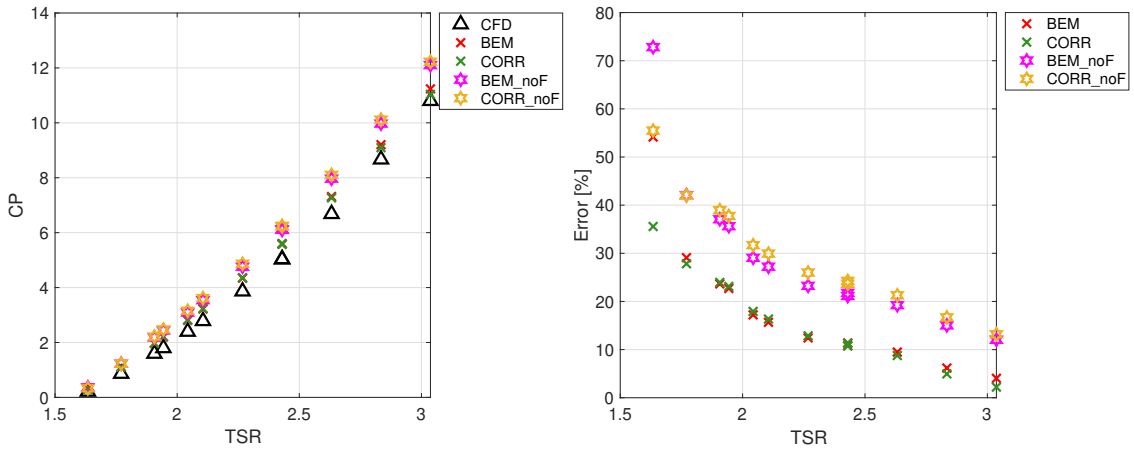
Total power comparison

Table 4.2 exhibits the individual power for each runner ($P = \Omega Q$) and the sum of both these contributions as calculated by CFD. The tip speed ratio, TSR, in the last column makes use of the averaged rotational speed of both runners. It is worth noticing that some operating conditions produce negative power in the downstream runner. This means that runner is somehow acting as a turbine instead. Despite this fact, the total power still remains positive.

Table 4.2: Individual and total power in pump mode

q [m^3/s]	N_1 [RPM]	N_2 [RPM]	$P_{1,CFD}$ [kW]	$P_{2,CFD}$ [kW]	$P_{tot,CFD}$ [kW]	TSR [-]
0.296	1200	900	16.642	12.033	28.674	2.43
0.296	1300	975	21.02	17.087	38.107	2.63
0.296	1400	1050	26.161	23.279	49.44	2.83
0.296	1500	1125	31.834	29.76	61.594	3.04
0.37	1200	900	15.605	4.402	20.007	1.94
0.37	1300	975	20.801	10.197	30.998	2.11
0.37	1400	1050	26.389	16.629	43.019	2.27
0.37	1500	1125	32.557	23.551	56.108	2.43
0.44	1200	900	12.037	-8.075	3.962	1.63
0.44	1300	975	17.845	-1.672	16.173	1.77
0.44	1400	1050	24.412	5.425	29.838	1.91
0.44	1500	1125	31.568	13.282	44.85	2.04

Figure 4.7a shows the total power coefficient (i.e. using the total power) from CFD as compared to the total power coefficient from BEM. Figure 4.7b displays the relative error of the calculated total power value from BEM as compared to CFD. It can be seen that that the trend of results is being followed and that for higher TSR the relative error is lower. Overall this variable is being overestimated by BEM.



(a) **(b)**

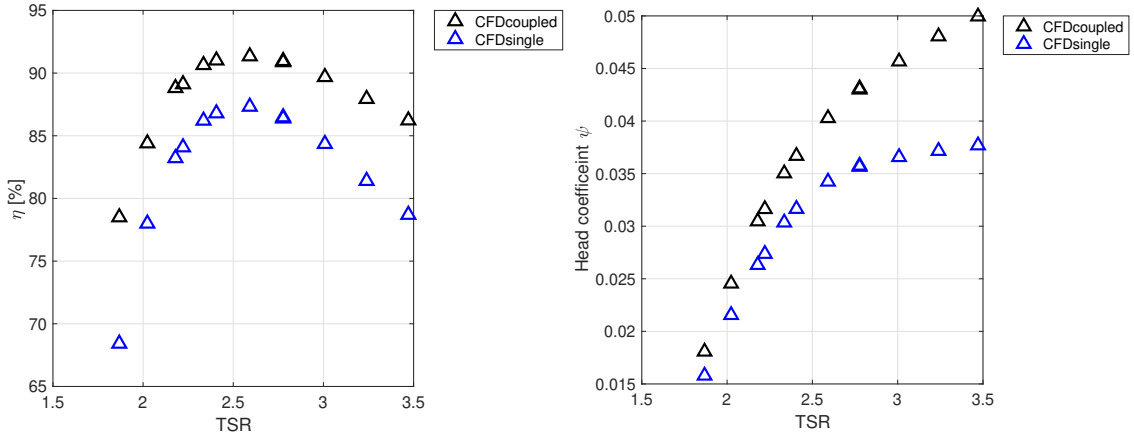
Figure 4.7: Total C_P and relative error of the total power pump mode

4.1.2 Individual runners set-up

The reason behind setting up an individual mesh for a single runner is that the BEM method does not differentiates if the upstream runner is coupled with a counter-rotating rotor downstream or if it is alone. Even though good agreement between BEM and the validating data for the upstream Runner 1 can be obtained for the couple case set-up in the previous section, a more accurate validation would use an individual geometry set-up to compare with BEM.

Rotor 1 upstream

Figure 4.8 displays the efficiency, η, and the head difference, ΔH, obtained in the coupled case set-up for upstream Rotor 1 as compared with the same values obtained in individual runner geometry set-up. It can be seen that the individual set up gives lower values of η and ΔH than the coupled case for the same operating conditions of Rotor 1.



(a) **(b)**

Figure 4.8: η_1 and ΔH_1 for R1 coupled set-up vs individual set-up, pump mode

Figure 4.9 shows the same results displayed previously in Figure 4.2 plus the results obtained from the single case CFD simulation for Rotor 1 in pump mode. It can be seen that the thrust and torque (i.e. power) are lower than the same loadings obtained from the coupled CFD simulation, and that the BEM results are in less agreement now for larger TSR.

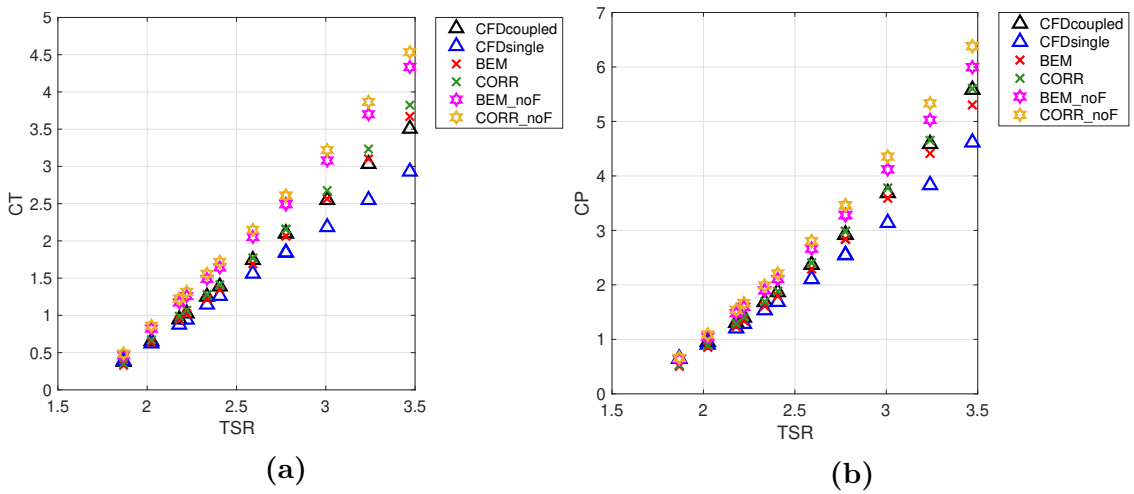


Figure 4.9: C_T and C_P upstream Runner 1, pump mode, single set-up comparison

Tangential and axial velocities after Rotor 1 from CFD

Figure 4.10 displays the circumferential averaged tangential velocities induced on the flow right after Runner 1 in pump mode. The vertical axis represents the dimensionless radius r/R where 1 represents the shroud. It can be seen that a larger swirl magnitude is being induced towards the tip and the hub, whereas for the mid-radius the tangential velocity magnitude is smaller.

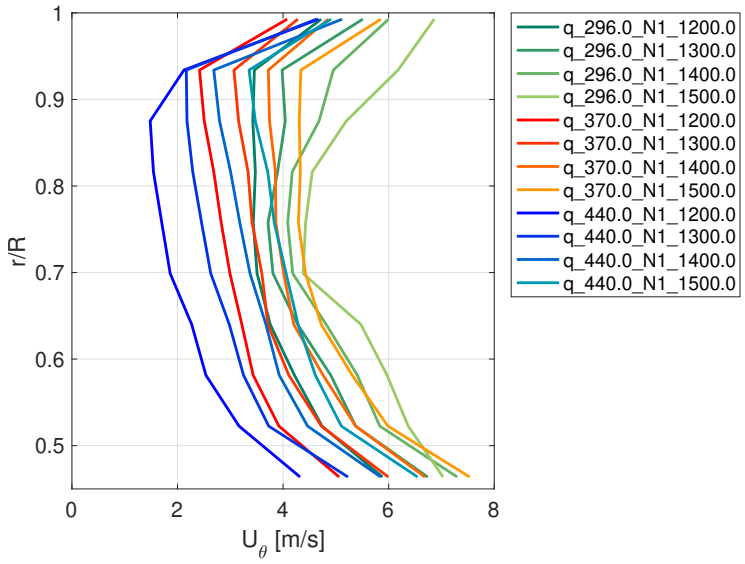


Figure 4.10: Tangential velocities along the radial direction from CFD right after Runner 1 individual set-up, pump mode

Figure 4.11 displays the axial velocity right after Runner 1 for the individual case set-up in pump mode. It is shown that very close to the shroud this velocity lowers as it approaches the wall. Furthermore, the profiles are close to the upstream axial velocity U_0 except from very close to the shroud. This is important as one main assumption is that the axial velocity remains constant throughout the turbomachine.

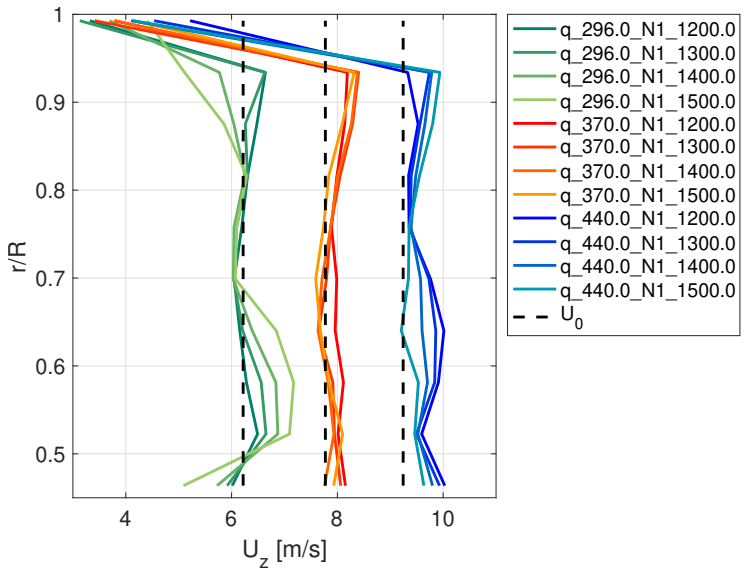


Figure 4.11: Axial velocities along the radial direction from CFD right after Runner 1 individual set-up, pump mode

These results were obtained by post-processing the CFD simulations with probe points. These points were located at certain radii and around the axis of rotation. The profiles in Figures 4.10 and 4.11 represent the circumferential average profiles.

CFD velocities replaced into BEM equations Rotor 1

The tangential and axial velocities displayed in the preceding section can be used to obtained the induction factors a and a' at the rotor plane that would induce those velocities. According to BEM $u_\theta = 2a'\Omega r$ and $u_z = U_0(1+a)$ in the case of a pump. The induction factors are used to compute the angle of attack, α from the velocity triangles and the corresponding loadings on the blade directly using the BEM equations and corresponding C_l and C_d. In this case there is no iteration procedure since a and a' are given and only need to be evaluated to obtain thrust and torque in the blade element, dT and dQ.

Figure 4.12 shows a comparison between the results obtained by CFD and the results obtained by BEM when the velocities after Rotor 2 are transformed to induction factors and replaced in the BEM equations.

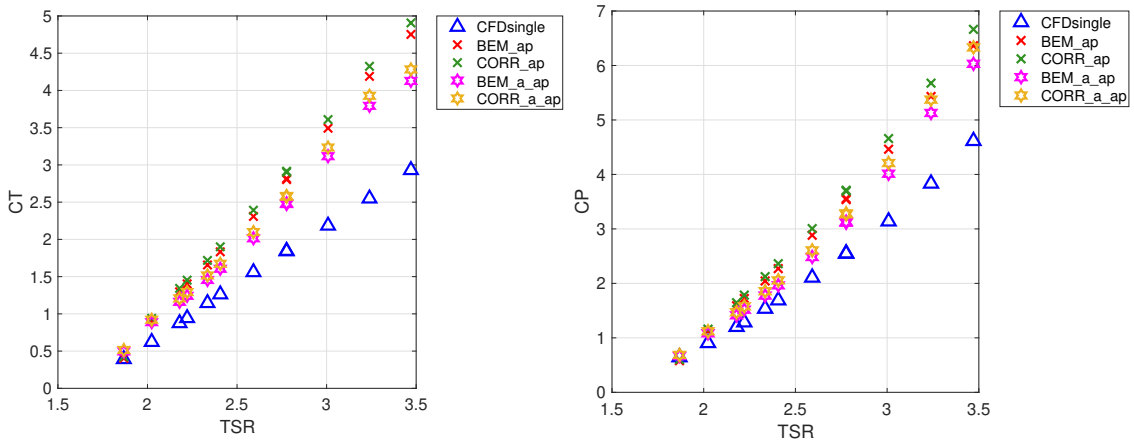
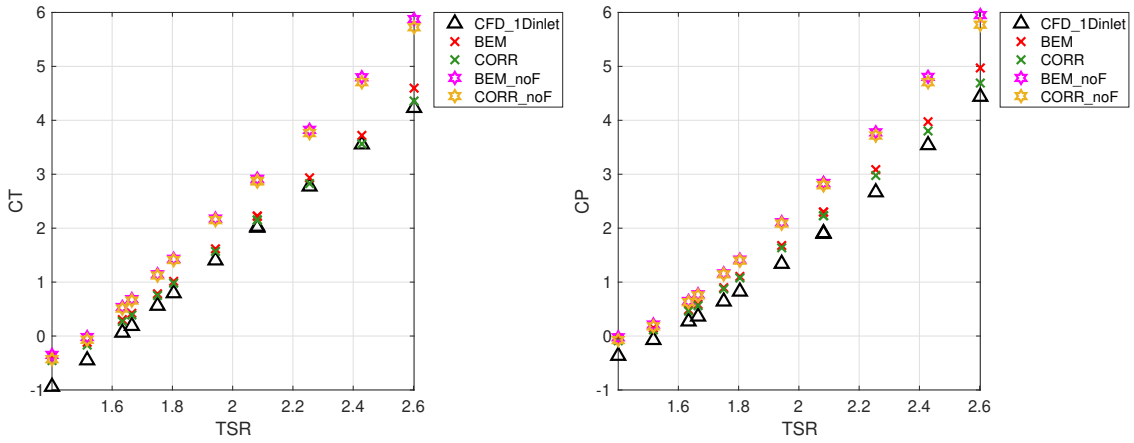


Figure 4.12: C_T and C_P calculated from replacing a and a' deduced from CFD upstream Runner 1, pump mode

As seen, performing this procedure with the velocity profiles obtained after Rotor 1 from CFD does not give the same thrust and torque (i.e power) results obtained from CFD. Overall these results are over-predicting the CFD values, especially at higher TSR.

Rotor 2 downstream `profile1DfixedValue` boundary condition

The following results were obtained by making use of `profile1DfixedValue` boundary condition. The tangential velocity profiles used as inlet condition are the ones obtained right after Runner 1 in pump mode individual geometry and shown in Figure 4.10. No axial velocity profile is implemented, instead the constant value $U_0 = q/A$ was used. The idea is to compare the BEM code for downstream Runner 2 with the same input as in OpenFOAM. This is done specifically to test the validity of the derived Eq. 2.67 for the downstream tangential induction factor a'_{d}.



(a) **(b)**

Figure 4.13: C_T and C_P downstream Runner 2 obtained with `profile1DfixedValue`, pump mode

It can be seen that for this case BEM results that include the F correction factor accommodate the CFD results better than no F factor. Overall this results are overestimating the CFD values.

4.2 Turbine Mode

In turbine mode the flow faces Runner 2 first and Runner 1 second. In this mode, the flow goes in the negative z-direction. Furthermore, Runner 2 rotates clockwise and Runner 1 in the anti-clockwise direction around z-axis if looked from left to right. Figure 4.14 depicts this situation.

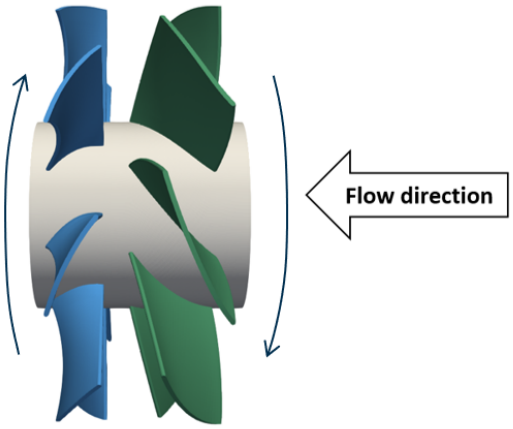


Figure 4.14: Flow direction and direction of rotation of runners in turbine mode

4.2.1 Coupled runners set-up

Table 4.3 shows the results obtained for individual pressure head drop, ΔH, and efficiencies, η, for both rotors in a coupled set up in turbine mode. It can be seen that for one operating condition the head drop across downstream Runner 1 is negative which means that rotor would be acting as a pump. These case can be regarded as non-feasible for the individual rotor. Furthermore, some upstream efficiencies are larger than 100%. In this cases even though the solution is converged, the available hydraulic power, $q\rho g\Delta H_2$, is not well represented by the individual head drop ΔH_2 in this off-design condition for the coupled solution.

Table 4.3: Head rise and efficiency coupled set up turbine mode

q [m^3/s]	N1 [RPM]	N2 [RPM]	dH_1 [m]	dH_2 [m]	η_1 [%]	η_2 [%]
0.224	850	637.5	0.77	1.6	59.05	89.19
0.224	1000	750	-0.3	0.66	-	121.69
0.24	700	525	2.29	2.37	92.08	87.85
0.24	850	637.5	1.39	2.07	85.13	88.18
0.24	1000	750	0.16	1.33	-	96.43
0.28	700	525	3.7	3.27	74.35	88.07
0.28	850	637.5	2.98	3.18	92.74	88.01
0.32	700	525	7.08	3.46	38.86	106.89
0.32	850	637.5	4.42	4.26	83.73	88.32
0.32	1000	750	3.71	4.09	92.92	88.15
0.336	700	525	8.8	3.8	35.05	106.36
0.336	850	637.5	5.23	4.71	76.62	88.21
0.336	1000	750	4.43	4.61	92.59	88.06

Rotor 2 upstream

Figure 4.15 displays a comparison of the thrust and power coefficients obtained from CFD and BEM for the upstream Runner 2 in turbine mode. It is shown that overall the BEM solution underestimates both values of thrust and power (i.e. torque), but when the pressure correction is implemented these loadings increase as expected. Furthermore, the F correction factor was both applied and deactivated (_noF) to check the effect on the results. No F correction factor increases both thrust and torque as expected.

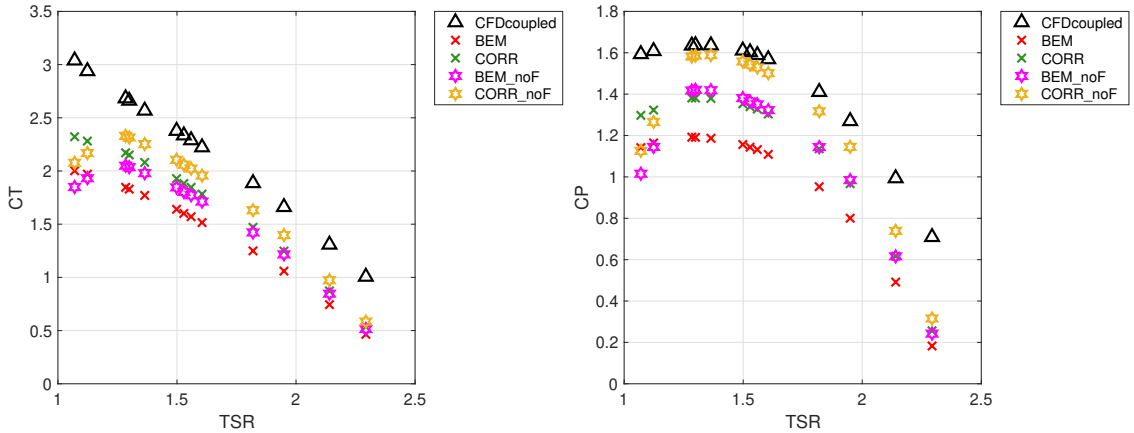
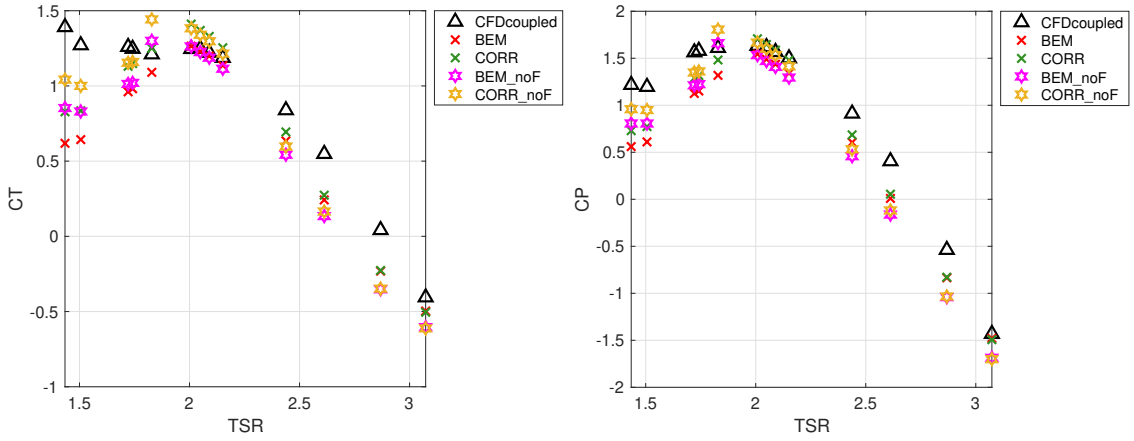


Figure 4.15: C_T and C_P vs TSR Rotor 2 turbine mode

Rotor 1 downstream

Figure 4.16 shows C_T and C_P for the downstream Rotor 1 in turbine mode. A larger disagreement can be seen for low TSR when it comes to thrust, but overall the BEM result follow the same trend as CFD. It is worth noting that when the power or thrust becomes negative for the downstream runner, i.e. the operating condition can be regarded as non feasible, the BEM-method still captures this trend.

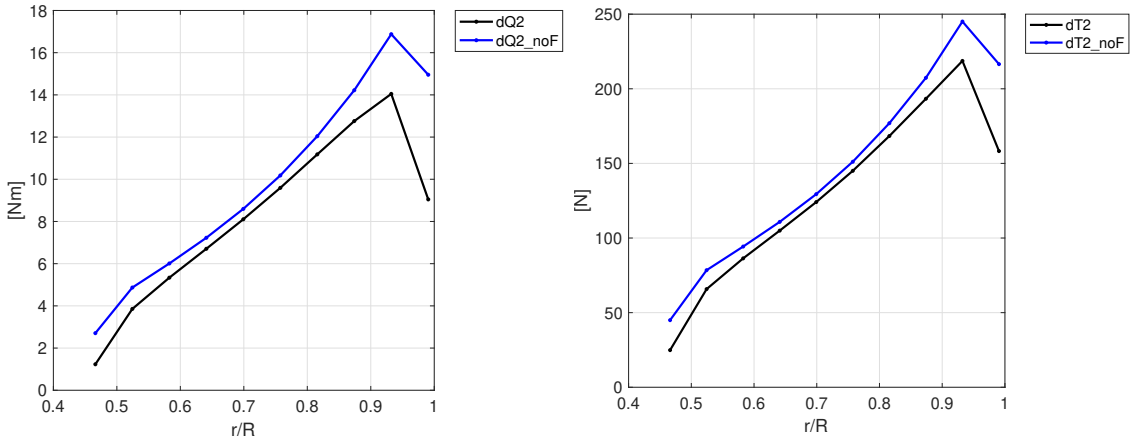


(a) **(b)**

Figure 4.16: C_T and C_P vs TSR Rotor 1 turbine mode

Case: q = 0.28 m^3/s, N_1 = 850 RPM, N_2 = 637.5 RPM

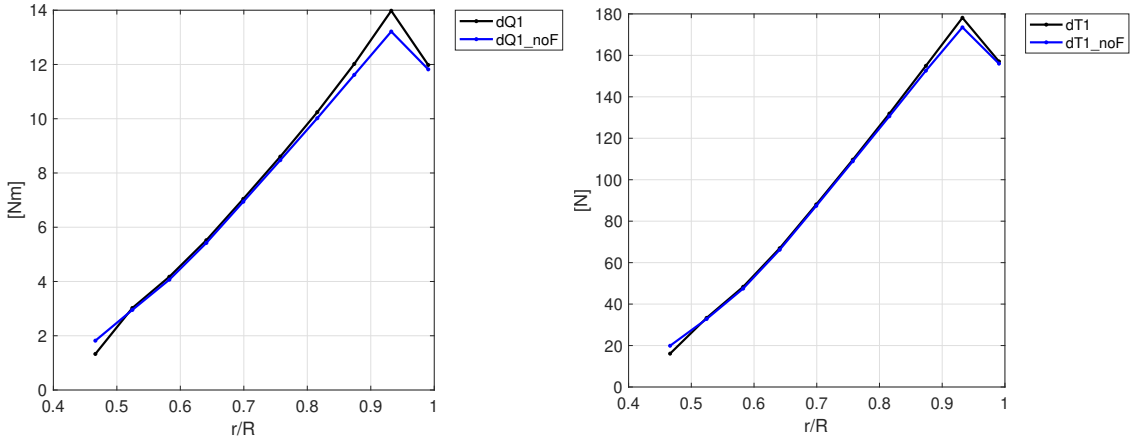
Figure 4.17 displays the blade elements torque, dQ ,and thrust, dT, at the upstream Runner 2 computed by BEM for the particular operating condition. It is worth recalling that the total thrust and torque on the blade is equal to the sum of all blade element contributions. Same as for pump mode, it is shown that the larger load contribution comes from blade elements close to the tip of the blade as compared to close to the hub. The same decline in loading as the blade element gets closer to the tip can be seen as compared to pump mode.



(a) **(b)**

Figure 4.17: dQ and dT contribution along upstream Rotor 2 blade calculated by BEM, turbine mode, specific case

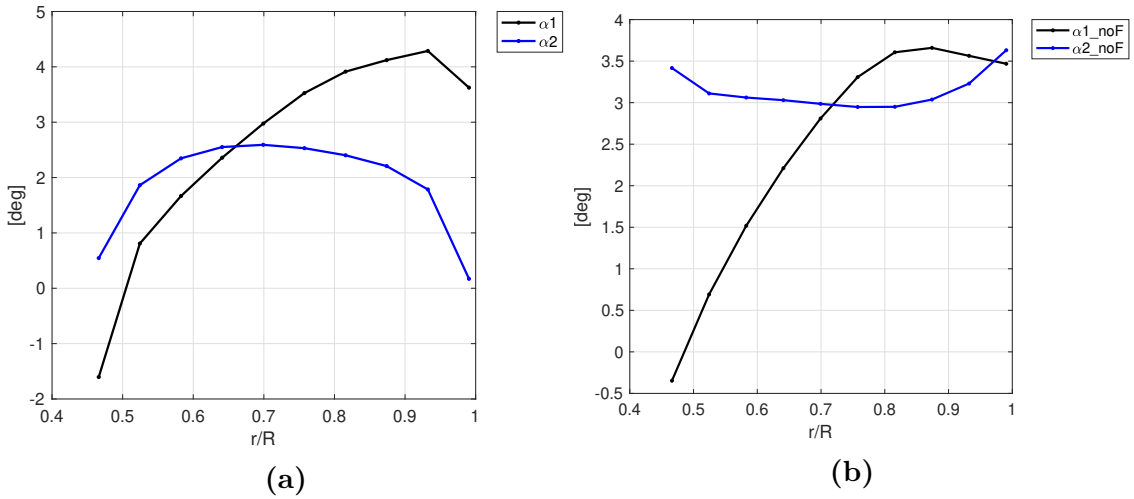
Furthermore, it can be seen that when the F Prandltl correction factor is active both dQ and dT are smaller in comparison to when no F factor is present. Figure 4.18 shows the blade element contribution as the previous figure, but for the downstream Rotor 1. Less of a difference is perceived with respect to deactivating the F factor in the downstream Rotor 1 for turbine mode.



(a) **(b)**

Figure 4.18: dQ and dT contribution along downstream Rotor 1 blade calculated by BEM, turbine mode, specific case

Figure 4.19 displays the angle of attack, α, computed by BEM in each blade element along the radial direction for both rotors. Figure 4.6b shows that when deactivating the F factor for upstream Rotor 2 the angles of attack towards the hub and tip are increased in relation to the middle.



(a) **(b)**

Figure 4.19: *AoA* along rotor blades predicted by BEM, turbine mode, specific case

Total power comparison

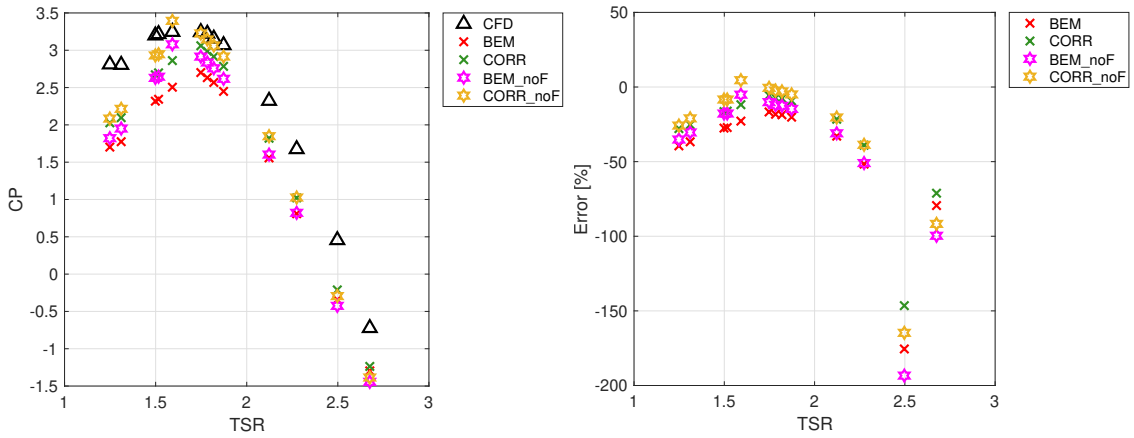
The total power extracted by the turbomachine in turbine mode is the sum of the individual power of each runner. The individual power is calculated as $P = \Omega Q$. In this sense if one of the runners gives a negative torque (i.e: power required), but the other runner does not, the total contribution could still be positive representing extracted power. Table 4.4 displays the total power of the combined runners from the CFD simulation.

Table 4.4: Individual and total power turbine mode

q [m^3/s]	N_1 [RPM]	N_2 [RPM]	$P_{1,CFD}$ [kW]	$P_{2,CFD}$ [kW]	$P_{tot,CFD}$ [kW]	TSR_{avg} [-]
0.224	850	637.5	1.001	3.138	4.139	2.27
0.224	1000	750	-3.54	1.753	-1.787	2.68
0.24	700	525	4.963	4.897	9.86	1.75
0.24	850	637.5	2.772	4.288	7.059	2.12
0.24	1000	750	-1.635	3.02	1.384	2.5
0.28	700	525	7.547	7.893	15.44	1.5
0.28	850	637.5	7.565	7.67	15.234	1.82
0.32	700	525	8.617	11.589	20.206	1.31
0.32	850	637.5	11.602	11.788	23.39	1.59
0.32	1000	750	10.805	11.305	22.11	1.87
0.336	700	525	10.152	13.292	23.445	1.25
0.336	850	637.5	13.177	13.653	26.83	1.52
0.336	1000	750	13.502	13.367	26.87	1.78

Figure 4.20 displays the total power coefficient C_P at different TSR. To calculate the TSR the rotational speed used is the average between both rotational speeds Ω_1 and Ω_2. Figure 4.20b displays the relative error encountered if the values of total

power are compared with the obtained total power from CFD. Larger relative error is perceived for higher TSR values. At these TSR the power extracted is also a small or even negative values. It is worth recalling that a high TSR means that the incoming flow is small as compared to the rotational speed of the runner.



(a) (b)

Figure 4.20: Total C_P and relative error of the total power turbine mode

4.2.2 Individual runners set-up

Same as in pump mode, by setting up individual geometries a more accurate comparison can be made with BEM for the upstream runner. Also for the downstream runner the same inlet conditions can be simulated individually in CFD and in BEM with the help of `profile1DfixedValue` boundary condition.

Rotor 2 upstream

Figure 4.21 shows a comparison between the individual efficiency η and head coefficient ψ obtained from the coupled geometry and the individual setup for upstream Runner 2 in turbine mode. It can be seen that for the particular operating conditions the efficiency goes over 100% for low and high values of TSR as seen in last column of Table 4.3. This individual efficiency from the coupled set-up is not completely reliable as the available hydraulic power for that individual runner is not well captured if ΔH_2 is used.

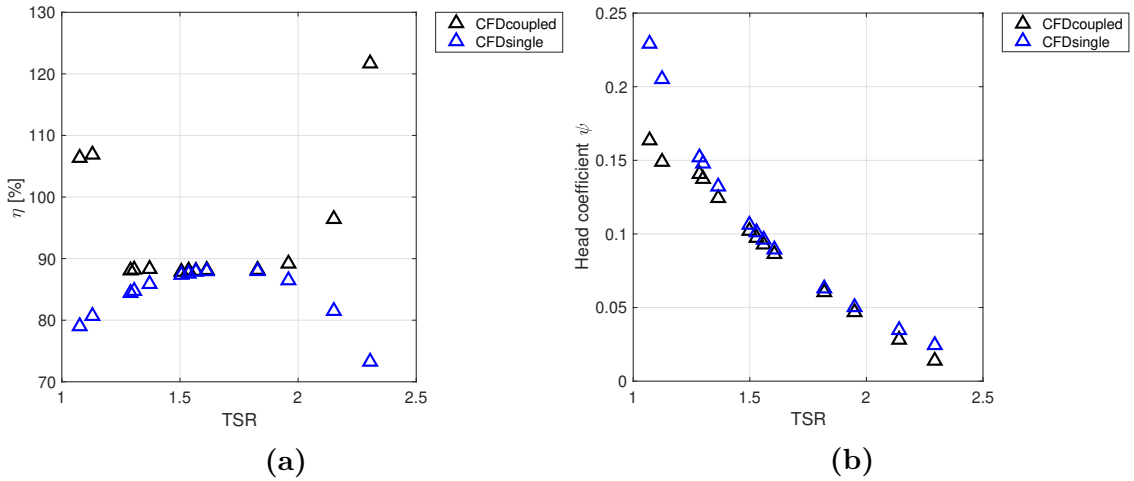


Figure 4.21: η_2 and ΔH_2 R2 coupled set-up vs individual set-up turbine mode

Figure 4.22 depicts the same results displayed in Figure 4.15 but with the additional C_T and C_P values obtained from the single geometry setup. It can be seen that thrust and torque (i.e. power) do not considerably differ from the values obtained in the coupled geometry for the upstream Runner 2. This fact is in contrast with what it was obtained for upstream Runner 1 in pump mode for both coupled and single geometries (Figure 4.9), where the difference was larger.

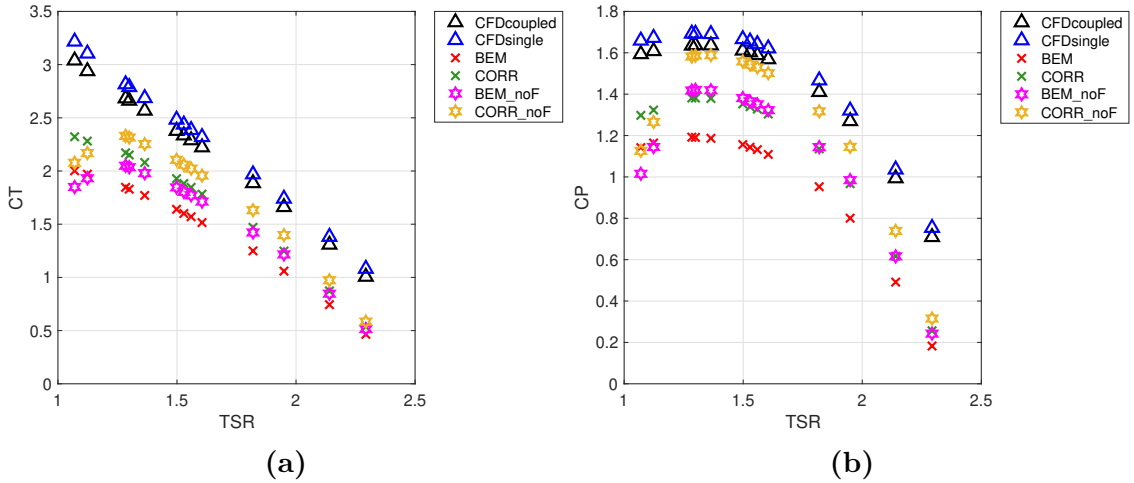


Figure 4.22: C_T and C_P upstream Runner 2 in turbine mode single case comparison

Tangential and axial velocities after Rotor 2 from CFD

Figure 4.23 shows the tangential velocities induced on the flow right after Runner 2 in turbine mode. This values were obtained from the single geometry setup simulations. Unlike the previous Figure 4.10 the current operational conditions induce higher tangential velocities towards the hub.

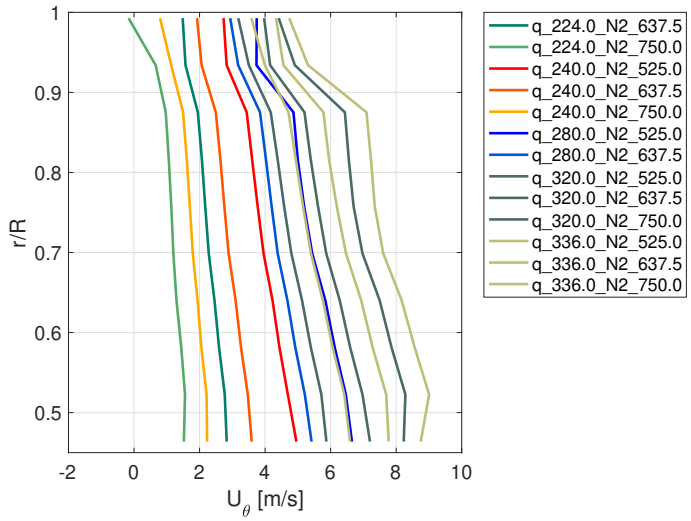


Figure 4.23: Tangential velocities along the radial direction right after Runner 2 in turbine mode

Figure 4.24 shows the circumferential average of the axial velocity right after Runner 2 for the individual case set-up in turbine mode. It is shown that the average profiles are not strictly uniform along the axial direction. For some operating conditions a lower axial velocity is perceived close to the shroud and a higher one occurs approaching towards the hub, as compared to the incoming axial velocity U_0.

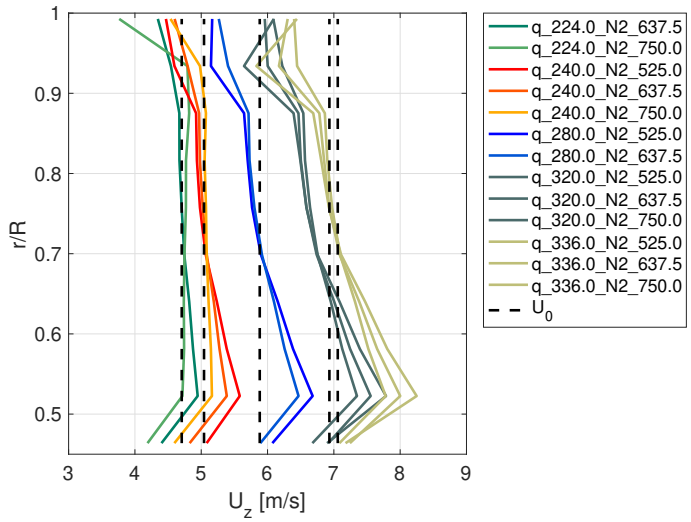


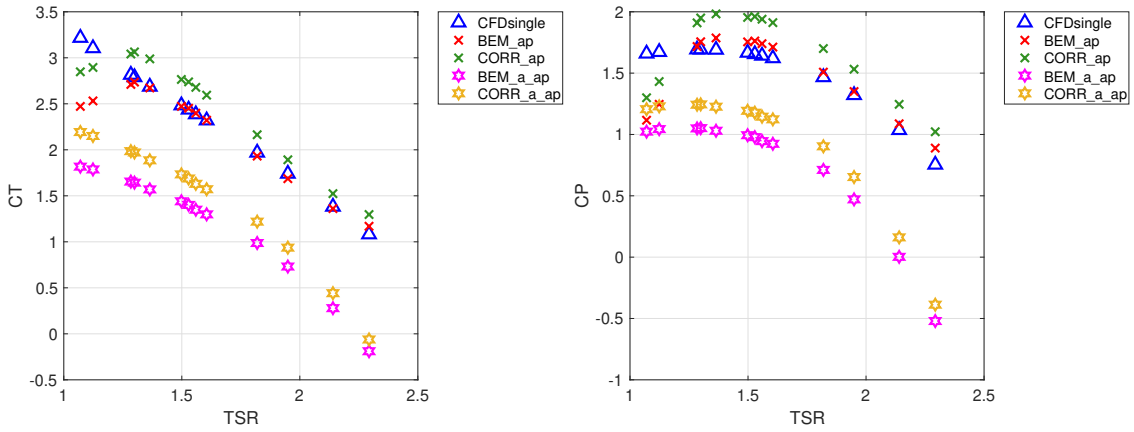
Figure 4.24: Axial velocities along the radial direction right after Runner 2 in turbine mode

CFD velocities replaced into BEM equations Rotor 2

Tangential and axial velocities displayed in Figures 4.23 and 4.24 can be used to obtained the induction factors a and a' at the rotor plane. This because according to BEM $u_\theta = 2a'\Omega r$ and $u_z = U_0(1-a)$ in turbine mode. The induction factors are used to compute the angle of attack, α and the corresponding loadings on the

blade directly. In this case there is no iteration procedure since a and a' are given and only need to be evaluated to obtain thrust, dT, and torque, dQ, at each blade element.

Figure 4.25 shows a comparison between the results obtained from CFD and the results given by BEM when the velocities after Rotor 2 are transformed to induction factors and replaced in the BEM equations. As seen, better agreement is obtained when the axial induction factoris set to zero ($a = 0$), i.e: when U_0 remains constant.

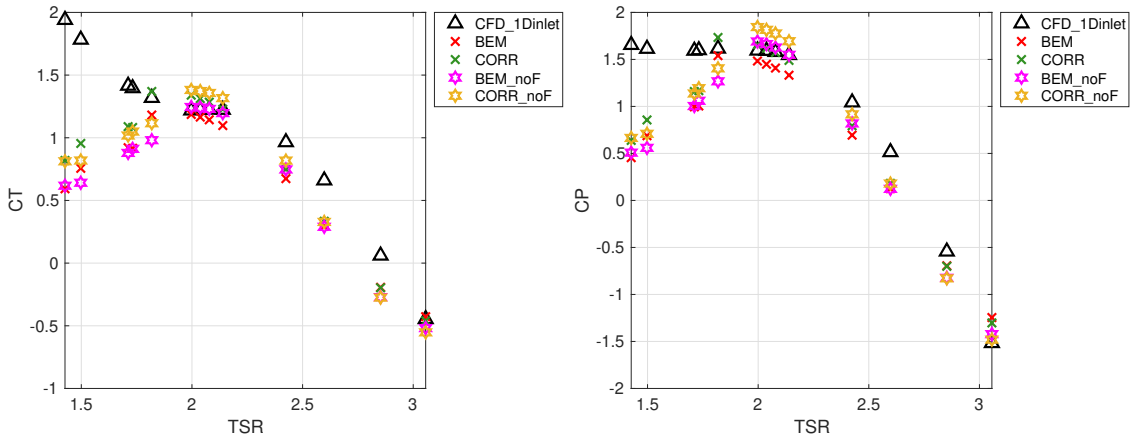


(a) **(b)**

Figure 4.25: C_T and C_P as calculated from replacing a and a' deduced from CFD upstream Runner 2 turbine mode

Rotor 1 downstream `profile1DfixedValue` boundary condition

Same as with the downstream rotor in pump mode, the following results were obtained using the `profile1DfixedValue` boundary condition. The tangential velocity profile used are the ones obtained right after upstream Runner 2 in turbine mode individual geometry (Figure 4.23). For axial velocity the constant value U_0 is considered. In a similar manner, the BEM code was run using the same tangential velocity profiles and constant axial velocity as inputs for the downstream Rotor 1. It is shown that there is a better agreement for a certain range of TSR, whereas for low TSR the trend is not being completely followed.



(a) **(b)** caption

Figure 4.26: C_T and C_P downstream Runner 1 obtained with `profile1DfixedValue`, turbine mode

5
Conclusion

The present work represents a first attempt to characterize the rotor blades of a Counter-Rotating Pump-Turbine (CRPT) to implement the classical BEM method in order to obtain the total thrust, T, torque, Q, and power, P, on the rotor blades. It is concluded that it is possible to use the BEM method with a reasonable amount of error, but most importantly that the trend seen for thrust and power coefficient, C_T and C_P at different TSR is being followed. Therefore, further work is required for full validation of the current methodology.

For pump mode the loadings in upstream Runner 1 predicted by BEM maintaining the F factor agree well with CFD data from the coupled case set-up (Figure 4.2). Nevertheless, when the individual CFD simulations were compared they differed from the loadings obtained from the coupled CFD for this rotor, especially as the TSR increases (Figure 4.9). Therefore, even though good agreement can be obtained for the coupled case, an individual geometry analysis raised up more uncertainty.

For the downstream Runner 2 in pump mode the formula proposed gives results that are overestimating the values obtained from CFD with a certain offset. Nevertheless, the same trend is being followed (Figure 4.3). Furthermore, no major difference is perceived when the F factor is deactivate.

In turbine mode the BEM results underestimate the loadings on the upstream Runner 2, but by deactivating the loss correction factor, F, the loadings can raise up to match the validation data, with the exception of extreme values of TSR (Figure 4.15). The CFD individual set up simulations gave similar results of C_T and C_P as the coupled CFD simulation for Runner 2 in turbine mode (Figure 4.22). Therefore, it can be said that in this mode the upstream runner is less sensitive to the operation of the downstream runner. Results for the downstream Runner 1 in turbine mode (Figure 4.16) are better captured for a certain TSR range by the different BEM simulations.

Comparing the total power coefficient in pump mode (Figure 4.7) it is concluded that overall the BEM calculations overestimate this variable as compared to CFD. Less relative error of the total power value is obtained for higher TSR. For turbine mode, the total power (Figure 4.20) is mainly being underestimated, and for a certain range of TSR the values are well captured.

The Prandtl correction factor F can be used to adjust the results. The omission of

this factor in the BEM equations increases the total blade thrust, T, and torque, Q, (i.e. power), while its presence decreases them. Generally, this effect is more notorious in the upstream runner than in the downstream runner. It is thought that this factor that could be of good use for future result tuning.

The extra loadings due to pressure correction use the head drop across the rotor ΔH as a known information to compute the static pressure difference Δp. The individual ΔH is taken as an input from the CFD simulations. This is used in combination with the tangential velocity, u_θ, before and after each runner as predicted by BEM to compute Δp thanks to the Bernoulli equation. In reality ΔH for each rotor is not previously known if not for CFD. Therefore, the pressure difference across the rotor remains unknown and the extra loading proposed becomes impractical to apply.

6
Further Work

Further work needs to be done to fully validate such a method for this type of turbomachine. For example, further investigation and comparison with individual rotor geometries could be done to better understand what kind of corrections are needed for the upstream runners. This given that it becomes inappropriate to isolate the effect of one runner in a full case geometry because the solution is coupled by the operation of both runners. It could be useful to further characterize the flow to understand where separation or secondary flows occur and that might be affecting the load distribution on the blade.

In order to make the current BEM-code faster it would be appropriate to reduce the levels of interpolation being carried out in the procedure. In this work the structural information of each blade is given at certain radii. Each one of this profiles has a different C_l and C_d curve for different Re numbers. The interpolation procedure first interpolates according to the radial location of the blade element and then by the Re number. It would be interesting to compare the BEM method results by the profile characteristics of these profiles at only $Re = 1 \times 10^6$. It is thought that there is no significant difference between the different Re of one profile characteristic. If no significant difference is observed, this simplification could save computation time if necessary.

In this work, the Viterna extrapolation method is used over the Montgomerie method because is the only method available in the the Python tool `AirfoilPrep.py` at the moment. Even though for the current operating conditions the upstream rotors do not present stall conditions, or out-of-range AoA, this might be the case for a different operating condition or even for the downstream rotor. For this cases it would be interesting to compare the different extrapolation methods and see how the results differ.

In the present work the profile characteristics for different AoA were obtained without any consideration of the proximity or spacing between blades. If such cascade effect were to be considered the C_l and C_d curves would need to be corrected accordingly. By knowing the spacing between rotor blades a cascade representation for different profiles could be obtained and simulated in a similar way as the individual profiles has been simulated in this thesis.

Additionally, different configurations of radial locations and blade elements along the blade could be tried out. As seen from the detailed information of thrust and torque

contributions, dT and dQ, along the rotor blades, larger contributions come from locations closer to the tip. It is thought that changing the radial points distributions to a denser configuration towards the tip could improve the level of detail of the loadings at this location.

Lastly, the current turbomachine geometry is in a reduced model scale. Therefore, a comparison of the implementation of the BEM method in prototype scale size could be implemented and the differences in relation with model scale could be assess in the future.

Bibliography

[1] *ALPHEUS - Augmenting Grid Stability Through Low Head Pumped Hydro Energy Utilization and Storage* [n.d.]. `https://alpheus-h2020.eu`.

[2] Bahaj, A., Molland, A., Chaplin, J. and Batten, W. [2007]. Power and thrust measurements of marine current turbines under various hydrodynamic flow conditions in a cavitation tunnel and a towing tank, *Renewable Energy* **32**(3): 407–426.

[3] Carlton, J. S. [2012]. *Marine propellers and propulsion*, 3 edn, Elsevier Science & Technology.

[4] Dixon, S. L. and Hall, C. A. [2014]. *Fluid mechanics and thermodynamics of turbomachinery*, Elsevier, Butterworth-Heinemann.

[5] Furukawa, A., Shigemitsu, T. and Watanabe, S. [2007]. Performance test and flow measurement of contra-rotating axial flow pump, *Journal of Thermal Science* **16**(1): 7–13.

[6] Hansen, M. O. L. [2015]. *Aerodynamics of wind turbines*, 3rd edn, Routledge.

[7] Jonkman, J. M., Butterfield, S., Musial, W. and Scott, G. [2009]. *Definition of a 5-MW Reference Wind Turbine for Offshore System Development*, National Renewable Energy Laboratory Colorado.

[8] Mahmuddin, F., Klara, S., Sitepu, H. and Hariyanto, S. [2017]. Airfoil lift and drag extrapolation with viterna and montgomerie methods, *Energy Procedia* **105**: 811–816.

[9] Ning, A. [2021]. Using blade element momentum methods with gradient-based design optimization, *Structural and Multidisciplinary Optimization* .
URL: *https://link.springer.com/10.1007/s00158-021-02883-6*

[10] Qudaih, M., Engel, B., Truijen, D., Kooning, J., Stockman, K., Hoffstaedt, J., Jarquin Laguna, A., Ansorena Ruiz, R., Goseberg, N., Bricker, J., Fahlbeck, J., Nilsson, H., Bossi, L., Joseph, M. and Zangeneh, M. [2020]. The Contribution of Low-head Pumped Hydro Storage to a successful Energy Transition, *Virtual 19th Wind Integration Workshop*.

[11] *Sig Turbomachinery Library OpenFoamTurbo profile1DfixedValue* [2008]. `https://openfoamwiki.net/index.php/Sig_Turbomachinery_Library_OpenFoamTurbo_profile1DfixedValue`. Accessed 2021-18-05.

[12] Sørensen, J. N. [2016]. *General Momentum Theory for Horizontal Axis Wind Turbines*, Research Topics in Wind Energy, Springer.

[13] Theodorsen, T., Stickle, G. and Breevoort, M. [1937]. Report no. 594, characteristics of six propellers including the high-speed range, *Journal of the Franklin Institute* **224**(5): 671.

[14] *Turbulence Modeling Resource, The Spalart-Allmaras Turbulence Model* [2021]. `https://turbmodels.larc.nasa.gov/spalart.html#sa`. Accessed 2021-18-05.

A

BEM code

A.1 Example of a main file

```
%% running pump mode rotor 1 and rotor 2 afterwards
clc
clear
close all
%%
%fluid data
fluid.('rho') = 998; %m3/s
fluid.('nu') =  1e-6; %

%structural data
arr1 = readmatrix('structData/R1geom.txt');
blade1.('r') = arr1(:,1)*1e-3;
blade1.('c') = arr1(:,2)*1e-3;
blade1.('beta') = arr1(:,3);
blade1.('Rhub') = arr1(1,1)*1e-3;
% blade1.('Rtip') = arr1(end,1)*1e-3;
blade1.('Rtip') = 137.3*1e-3;
blade1.('z') = 8;

arr2 = readmatrix('structData/R2geom.txt');
blade2.('r') = arr2(:,1)*1e-3;
% blade2.('c') = arr2(:,2)*1e-3;
blade2.('c') = [77.1463; 81.7824; 85.4908; 88.3614; 90.4867; 91.9801; 92.9745;
    93.6490; 93.6666]*1e-3;
blade2.('beta') = arr2(:,3);
blade2.('Rhub') = arr2(1,1)*1e-3;
% blade2.('Rtip') = arr2(end,1)*1e-3;
blade2.('Rtip') = 137.3*1e-3;
blade2.('z') = 7;

% radial positions
r = linspace(64,136,10)*1e-3;

array = readmatrix('validation/pumpModeData');

%input velocity and rot speed
Qflow = array(:,1);
A = ((138e-3)^2-(62.3e-3)^2)*pi; %[m2] annulus area model scale
v = Qflow./A;
N1 = array(:,2);
N2 = array(:,3);
omega1 = N1*pi/30;
omega2 = N2*pi/30;
pitch = 0;

T1val = -(array(:,16));
T2val = -(array(:,17));
Q1val = -(array(:,14));
Q2val = (array(:,15));
% P1val = array(:,9);
% P2val = array(:,10);
P1val = Q1val.*omega1;
```

```
52 P2val = Q2val.*omega2;
53
54 eff1 = array(:,12);
55 eff2 = array(:,13);
56
57 dH1 = array(:,6);
58 dH2 = array(:,7);
59
60 setting.('machine')='CRPT';
61 setting.('flag')='pump';
62
63 %%
64 T2 = zeros(length(Qflow),1); T2corr = T2; T1 = T2; T1corr =T2;
65 Q2 = zeros(length(Qflow),1); Q2corr = Q2; Q1 = Q2; Q1corr =T2;
66 P2 = zeros(length(Qflow),1); P2corr = P2; P1 = P2; P1corr =T2;
67
68 % R1 --> R2 (pump)
69 for i=1:length(Qflow)
70 %     i=8
71     setting.('downstream') = 'no';
72     setting.('rot') = 1;
73     [dataR1] = BEM_fzero_mod2(v(i),N1(i),pitch,r,setting,blade1,fluid);
74     T1(i) = dataR1.T; Q1(i) = dataR1.Q; P1(i) = dataR1.P;
75     fprintf("U = %.2f, Power = %.2f [kW], Thrust = %.2f [N], Torque = %.2f [Nm]\n",
       v(i),P1(i)*1e-3,T1(i),Q1(i))
76     [T1corr(i),Q1corr(i),P1corr(i)] = dPcorr2(dH1(i),fluid,dataR1,setting);
77     fprintf("U = %.2f, Power = %.2f [kW], Thrust = %.2f [N], Torque = %.2f [Nm]\n",
       v(i),P1corr(i)*1e-3,T1corr(i),Q1corr(i))
78     fprintf("Reference:\n")
79     fprintf("U = %.2f, Power = %.2f [kW], Thrust = %.2f [N], Torque = %.2f [Nm],
       Eff = %.2f [%%] \n",v(i),P1val(i)*1e-3,T1val(i),Q1val(i),eff1(i))
80
81     dataR1.Tcorr = T1corr(i);
82     dataR1.Qcorr = Q1corr(i);
83     dataR1.Pcorr = P1corr(i);
84     varName = sprintf("q_%.0f_N1_%.0f",Qflow(i)*1000,N1(i));
85     pR1.(varName) = dataR1;
86
87     setting.('downstream') = 'yes';
88     setting.('rot') = 2;
89     setting.('ap_up') = dataR1.ap;
90     [dataR2] = BEM_fzero_mod2(v(i),N2(i),pitch,r,setting,blade2,fluid);
91     T2(i) = dataR2.T; Q2(i) = dataR2.Q; P2(i) = dataR2.P;
92     fprintf("U = %.2f, Power = %.2f [kW], Thrust = %.2f [N], Torque = %.2f [Nm]\n",
       v(i),P2(i)*1e-3,T2(i),Q2(i))
93     [T2corr(i),Q2corr(i),P2corr(i)] = dPcorrDown(dH2(i),fluid,dataR1,dataR2,setting
       );
94     fprintf("U = %.2f, Power = %.2f [kW], Thrust = %.2f [N], Torque = %.2f [Nm]\n",
       v(i),P2corr(i)*1e-3,T2corr(i),Q2corr(i))
95     fprintf("Reference:\n")
96     fprintf("U = %.2f, Power = %.2f [kW], Thrust = %.2f [N], Torque = %.2f [Nm],
       Eff = %.2f [%%] \n",v(i),P2val(i)*1e-3,T2val(i),Q2val(i),eff2(i))
97
98     dataR2.Tcorr = T2corr(i);
99     dataR2.Qcorr = Q2corr(i);
100    dataR2.Pcorr = P2corr(i);
101    varName = sprintf("q_%.0f_N2_%.0f",Qflow(i)*1000,N2(i));
102    pR2.(varName) = dataR2;
103 end
104 % Uncomment to save
105 % save('variables/pR1_noF.mat','-struct','pR1');
106 % save('variables/pR2_down_noF.mat','-struct','pR2');
107
108 %% write into a file
109 % This is one way to write results into a txt file
110
111 % relative error [%]
112 e_T1 = (-(T1val - T1)./T1val).*100;
113 e_T1corr = (-(T1val - T1corr)./T1val).*100;
114
```

```
115 e_Q1 = (-(Q1val - Q1)./Q1val).*100;
116 e_Q1corr = (-(Q1val - Q1corr)./Q1val).*100;
117
118 e_P1 = (-(P1val - P1)./P1val).*100;
119 e_P1corr = (-(P1val - P1corr)./P1val).*100;
120
121 e_T2 = (-(T2val - T2)./T2val).*100;
122 e_T2corr = (-(T2val - T2corr)./T2val).*100;
123
124 e_Q2 = (-(Q2val - Q2)./Q2val).*100;
125 e_Q2corr = (-(Q2val - Q2corr)./Q2val).*100;
126
127 e_P2 = (-(P2val - P2)./P2val).*100;
128 e_P2corr = (-(P2val - P2corr)./P2val).*100;
129
130 % upstream
131 header = 'Q[m3/s] \t N1[RPM] \t dH1[m] \t Eff1[%] \t T1val[N] \t T1 \t error \t
        T1corr \t error \n';
132 values = [Qflow N1 dH1 eff1 T1val T1 e_T1 T1corr e_T1corr];
133 fid = fopen('results/pR1R2_noF_T1.txt', 'w');
134 fprintf(fid, header);
135 fprintf(fid, '%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\n', values');
136 fclose(fid);
137
138 header = 'Q[m3/s] \t N1[RPM] \t dH1[m] \t Eff1[%] \t Q1val[Nm] \t Q1 \t error \t
        Q1corr \t error \n';
139 values = [Qflow N1 dH1 eff1 Q1val Q1 e_Q1 Q1corr e_Q1corr];
140 fid = fopen('results/pR1R2_noF_Q1.txt', 'w');
141 fprintf(fid, header);
142 fprintf(fid, '%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\n', values');
143 fclose(fid);
144
145 header = 'Q[m3/s] \t N1[RPM] \t dH1[m] \t Eff1[%] \t P1val[W] \t P1 \t error \t
        P1corr \t error \n';
146 values = [Qflow N1 dH1 eff1 P1val P1 e_P1 P1corr e_P1corr];
147 fid = fopen('results/pR1R2_noF_P1.txt', 'w');
148 fprintf(fid, header);
149 fprintf(fid, '%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\n', values');
150 fclose(fid);
151
152 % downstream
153 header = 'Q[m3/s] \t N2[RPM] \t dH2[m] \t Eff2[%] \t T2val[N] \t T2 \t error \t
        T2corr \t error \n';
154 values = [Qflow N2 dH2 eff2 T2val T2 e_T2 T2corr e_T2corr];
155 fid = fopen('results/pR1R2_noF_T2.txt', 'w');
156 fprintf(fid, header);
157 fprintf(fid, '%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\n', values');
158 fclose(fid);
159
160 header = 'Q[m3/s] \t N2[RPM] \t dH2[m] \t Eff2[%] \t Q2val[Nm] \t Q2 \t error \t
        Q2corr \t error \n';
161 values = [Qflow N2 dH2 eff2 Q2val Q2 e_Q2 Q2corr e_Q2corr];
162 fid = fopen('results/pR1R2_noF_Q2.txt', 'w');
163 fprintf(fid, header);
164 fprintf(fid, '%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\n', values');
165 fclose(fid);
166
167 header = 'Q[m3/s] \t N2[RPM] \t dH2[m] \t Eff2[%] \t P2val[W] \t P2 \t error \t
        P2corr \t error \n';
168 values = [Qflow N2 dH2 eff2 P2val P2 e_P2 P2corr e_P2corr];
169 fid = fopen('results/pR1R2_noF_P2.txt', 'w');
170 fprintf(fid, header);
171 fprintf(fid, '%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\n', values');
172 fclose(fid);
```

A.2 BEM solver

The axial induction factor a is omitted in this script.

```
function [data] = BEM_fzero_CRPT(v,n,pitch,r,set,blade,fluid)
%   BEM function for the machine CRPT
%
%   inputs
%   v       [mx1 double]        = freestrem incoming velocity   [m/s]
%   n       [mx1 double]        = rotational speed rotor        [RPM]
%   r       [mx1 array]         = radial points to perform BEM   [m]
%   blade   [1x1 struct]        = rotor geometric data
%   fluid   [1x1 struct]        = fluid properties data
%
%   outputs
%   data    [1x1 struct]        = output data packed in a structure

%% Input data
% interpolate structural properties at radial points

% chord length at every radial position [m]
c = interp1(blade.r,blade.c,r); %array with length(r1)

% twist angle (pitch) for every profile [degrees]
beta = interp1(blade.r,blade.beta,r);
beta = beta + pitch;

% rotational speed [rad/s]
omega = n*pi/30;

% solidity [-]
sigma = blade.z*c./(2*pi*r);

% initialize variables
elem=length(r);

a=zeros(elem,1); ap=zeros(elem,1);
F=ones(elem,1); U=ones(elem,1); Re=ones(elem,1); cn=ones(elem,1); ct=ones(elem,1);
alpha=ones(elem,1);

% defining inputs
machine = set.machine;
flag = set.flag;

% define coefficient C depending on mode
if strcmp(flag,'turbine'); C=-1; elseif strcmp(flag,'pump'); C = 1;
else; error("flag must be 'turbine' or 'pump'"); end

%% rotor

for j=1:length(r)
    x0 = 50;
    phi = fzero(@(phi) func(phi,v,omega),x0);

    fprintf('Radius %.4f, F=%.2f, phi=%.2f, alpha=%.2f, Re=%.2g,[a,ap]=[%.4f,%.4f]\
    n',r(j),F(j),phi,alpha(j),Re(j),a(j),ap(j));
end

%% differential load calculations
% blade elements span dr [m]
if isrow(r); r=r'; sigma=sigma';end

dr = zeros(length(r),1);
aux=diff(r);

dr(1) = aux(1)/2 + (r(1) - blade.Rhub);
dr(end) = aux(end)/2 + (blade.Rtip - r(end));
for i=2:length(dr)-1
    dr(i) = aux(i-1)/2 + aux(i)/2;
end

%% Thrust and torque

```

```
% blade element theory
% thrust [N]
dT = sigma.*pi*fluid.rho.*U.^2.*cn.*r.*dr;
T = sum(dT);

% torque [Nm]
dQ = sigma.*pi*fluid.rho.*U.^2.*ct.*r.^2.*dr;
Q = sum(dQ);

% power [W]
P = omega.*Q;

% output into a structure
data.('T') = T;
data.('Q') = Q;
data.('P') = P;
data.('dT') = dT;
data.('dQ') = dQ;
data.('ap') = ap;
data.('r') = r;
data.('dr') = dr;
data.('beta') = beta;
data.('c') = c;
data.('omega') = omega;
data.('alpha') = alpha;
data.('Re') = Re;
data.('F') = F;
data.('Uaxial') = v;

%% Nested functions (used by fzero)

    function res = func(phi,v,omega)
        [ap(j)] = induc_factors(phi);
        res = sind(phi)-cosd(phi)*v/((1-C*ap(j))*(omega*r(j)));
    end

    function [ap] = induc_factors(phi)
        [cn(j),ct(j)] = force_coeff(phi);

        % manually uncomment and comment if no F factor
        F(j) = lossCorr(phi,r(j),blade);
%         F(j) = 1;

        % tangential induction factor a' and auxiliary term k'
        kp = 4*F(j)*sind(phi)*cosd(phi)./(sigma(j)*ct(j));
        ap = 1./(kp+C);

        % downstream runner
         if strcmp(set.downstream,'yes')
             if strcmp(flag,'pump')

             ap_up = set.('ap_up');
             kp = 4*F(j)*sind(phi)*cosd(phi)./(sigma(j)*ct(j));
             ratio = 0.75; %Omega2/Omega1
             ap = (2*ap_up(j)*kp./ratio - 1)./(-kp-C);

             elseif strcmp(flag,'turbine')

             ap_up = set.('ap_up');
             kp = 4*F(j)*sind(phi)*cosd(phi)./(sigma(j)*ct(j));
             ratio = 0.75; %Omega2/Omega1
             ap = (2*ap_up(j)*kp.*ratio - 1)./(-kp-C);

             end
        end
    end

    function [cn,ct] = force_coeff(phi)
        alpha(j) =  C*(beta(j) - phi);

```

```
        % Reynolds number [-]
        Ux = v;
        Uy = omega*r(j)*(1-C*ap(j));
        U(j) = sqrt(Ux^2 + Uy^2);
        Re(j) = U(j)*c(j)./fluid.nu;

%         [cl,cd] = tableLookup_oneRe(alpha(j), Re(j), r(j), flag, machine,set.rot)
    ;
        [cl,cd] = tableLookup_allRe(alpha(j), Re(j), r(j), flag, machine,set.rot);

        cn = cl*cosd(phi) - C*cd*sind(phi);
        ct = cl*sind(phi) + C*cd*cosd(phi);
    end

end
```

A.3 Interpolation

```
function [cl,cd] = tableLookup_allRe(alpha, Re, r, flag, machine,rot)
%   [cl,cd] = tableLookup(alpha, Re, r, flag, machine,rot)
%   For the CRPT this function computes cl and cd from the two closest Re numbers
    and radiuses
%   and returns the interpolation for the given Re and r. In this case it uses
%   the auxiliary function auxInterp for the radial interpolation.
%   For the other machines there is no Re interpolation.


if machine == "CRPT"
    if Re <= 0.5e6
        Re_file = '0.5e6';
        [cl,cd] = auxInterp(flag,rot,r,Re_file,alpha);

    elseif Re > 0.5e6 && Re <= 1e6
        Re_file1 = '0.5e6';
        Re_file2 = '1.0e6';

        [cl1,cd1] = auxInterp(flag,rot,r,Re_file1,alpha);
        [cl2,cd2] = auxInterp(flag,rot,r,Re_file2,alpha);

        cl = interp1([0.5e6,1e6], [cl1,cl2],Re);
        cd = interp1([0.5e6,1e6], [cd1,cd2],Re);

    elseif Re > 1e6 && Re <= 2e6
        Re_file1 = '1.0e6';
        Re_file2 = '2.0e6';

        [cl1,cd1] = auxInterp(flag,rot,r,Re_file1,alpha);
        [cl2,cd2] = auxInterp(flag,rot,r,Re_file2,alpha);

        cl = interp1([1e6,2e6], [cl1,cl2],Re);
        cd = interp1([1e6,2e6], [cd1,cd2],Re);

    elseif Re > 2e6
        Re_file = '2.0e6';
        [cl,cd] = auxInterp(flag,rot,r,Re_file,alpha);
    end

elseif machine == "NREL_5MW"
    %struct data
    array = readmatrix('structData/distBladeProp.txt');
    R = array(:,1);

    airfoil = cell(length(array),1);
    airfoil(1:3,1) = {'Cylinder1'};
    airfoil(4,1) = {'Cylinder2'};
    airfoil(5,1) = {'DU40'};
```

```
    airfoil(6:7,1) = {'DU35'};
    airfoil(8,1) = {'DU30'};
    airfoil(9:10,1) = {'DU25'};
    airfoil(11:12,1) = {'DU21'};
    airfoil(13:19,1) = {'NACA64'};

    for i=1:length(R)-1
        if r >= R(i) && r <= R(i+1)
            f1 = sprintf([airfoil{i,1} '%s'],'L.txt');
            f2 = sprintf([airfoil{i+1,1} '%s'],'L.txt');
            path_f1 = ['tables/NREL_5MW/' f1];
            path_f2 = ['tables/NREL_5MW/' f2];
            A1 = readmatrix(path_f1);
            A2 = readmatrix(path_f2);
            cl1 = interp1(A1(:,1),A1(:,2),alpha);
            cl2 = interp1(A2(:,1),A2(:,2),alpha);
            cl = interp1([R(i), R(i+1)],[cl1, cl2],r);

            f1 = sprintf([airfoil{i,1} '%s'],'D.txt');
            f2 = sprintf([airfoil{i+1,1} '%s'],'D.txt');
            path_f1 = ['tables/NREL_5MW/' f1];
            path_f2 = ['tables/NREL_5MW/' f2];
            A1 = readmatrix(path_f1);
            A2 = readmatrix(path_f2);
            cd1 = interp1(A1(:,1),A1(:,2),alpha);
            cd2 = interp1(A2(:,1),A2(:,2),alpha);
            cd = interp1([R(i), R(i+1)],[cd1, cd2],r);
        end
    end

elseif machine == "tidal"

    path_f = 'tables/Tidal/NACA_63815.dat';
    A = readmatrix(path_f);

    cl = interp1(A(:,1),A(:,2),alpha);
    cd = interp1(A(:,1),A(:,3),alpha);

elseif machine == "propeller"

    path_f = 'tables/Propeller/CLARKY.dat';
    A = readmatrix(path_f);
    A = [A(1:end-1,1) A(1:end-1,5) A(1:end-1,9)];
    A = [A; 180 A(end,2) A(end,3)];

    cl = interp1(A(:,1),A(:,2),alpha);
    cd = interp1(A(:,1),A(:,3),alpha);

else
    error("pick a valid machine: 'NREL_5MW', 'tidal', 'propeller' or 'CRPT'")
end

end
```

A.4 Pressure Correction

```
function [T,Q,P] = dPcorr(dH,fluid,data,setting)
% Extra thrust and torque due to the pressure difference.
% applied after BEM is implemeneted
%
%   inputs
%   dH       [1x1 double]       = Head drop across rotor   [m]
%   fluid    [1x1 struct]       = fluid properties
%   data     [1x1 struct]       = output structure from BEM function
%   setting  [1x1 struct]       = setting.flag -> "pump" or "turbine"
%
%   outputs
%   T       [1x1 double]    = Thrust after pressure load [N]
```

```
%   Q       [1x1 double]    = Torque after pressure load [Nm]
%   P       [1x1 double]    = Power after pressure load [W]

ap = data.ap;
r = data.r;
dr =data.dr;
beta = data.beta;
c = data.c;
omega = data.omega;

beta = beta';
c = c';

g = 9.81;
rho = fluid.rho;
u_theta = 2.*ap.*omega.*r; %tangential velocity

if strcmp(setting.flag,'pump')
    dp_static = dH*rho*g - 0.5*rho*(u_theta).^2; % [Pa]
elseif strcmp(setting.flag, 'turbine')
    dp_static = dH*rho*g + 0.5*rho*(u_theta).^2; % [Pa]
end

dT_pcorr = dp_static.*cosd(beta).*dr.*c;
dQ_pcorr = dp_static.*sind(beta).*dr.*c.*r;

Tcorr = sum(dT_pcorr);
Qcorr = sum(dQ_pcorr);

T = data.T;
Q = data.Q;

T = T + Tcorr;
Q = Q + Qcorr;

P = omega.*Q;

end
```

A.5 Pressure Correction Downstream

```
function [T,Q,P] = dPcorrDown(dH,fluid,dataUp,dataDown,setting)
% Extra thrust and torque due to the pressure difference.
% For downstream rotor applied after BEM is implemented
%
%   inputs
%   dH       [1x1 double]     = Head drop across rotor   [m]
%   fluid    [1x1 struct]     = fluid properties
%   dataUp   [1x1 struct]     = output structure from BEM function upstream
%   dataDown [1x1 struct]     = output structure from BEM function downstream
%   setting  [1x1 struct]     = setting.flag -> "pump" or "turbine"
%
%   outputs
%   T       [1x1 double]    = Thrust after pressure load [N]
%   Q       [1x1 double]    = Torque after pressure load [Nm]
%   P       [1x1 double]    = Power after pressure load [W]

ap_up = dataUp.ap;
omega_up = dataUp.omega;

ap_down = dataDown.ap;
omega_down = dataDown.omega;

r = dataDown.r; % same r upstream and downstream runner
dr =dataDown.dr;
beta = dataDown.beta;
c = dataDown.c;

```

```
28
29 beta = beta';
30 c = c';
31
32 g = 9.81;
33 rho = fluid.rho;
34 u_theta_up = 2.*ap_up.*omega_up.*r; %tangential velocity
35
36 du_theta = 4.*ap_up.*omega_up.*r + 2.*ap_down.*omega_down.*r;
37 if strcmp(setting.flag,'pump')
38     dp_static = dH*rho*g - rho*u_theta_up.*du_theta - 0.5*rho.*du_theta.^2; % [Pa]
39 elseif strcmp(setting.flag, 'turbine')
40     dp_static = dH*rho*g + rho*u_theta_up.*du_theta + 0.5*rho.*du_theta.^2; % [Pa]
41 end
42
43
44 dp_static(1) = 0; dp_static(end) = 0; %skip close to tip and hub
45
46 dT_pcorr = dp_static.*cosd(beta).*dr.*c;
47 dQ_pcorr = dp_static.*sind(beta).*dr.*c.*r;
48
49 Tcorr = sum(dT_pcorr);
50 Qcorr = sum(dQ_pcorr);
51
52 T = dataDown.T;
53 Q = dataDown.Q;
54
55 T = T + Tcorr;
56 Q = Q + Qcorr;
57
58 P = omega_down.*Q;
59
60 end
```

B
Lift and Drag Tables

B.1 Polar curves

Re = 0.5e6

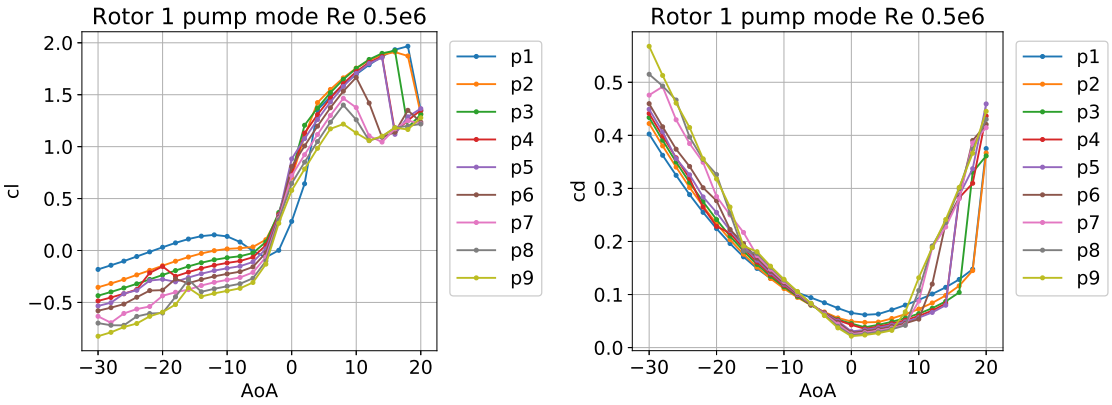


Figure B.1: Rotor 1 profile characteristics in pump mode $Re = 0.5e6$

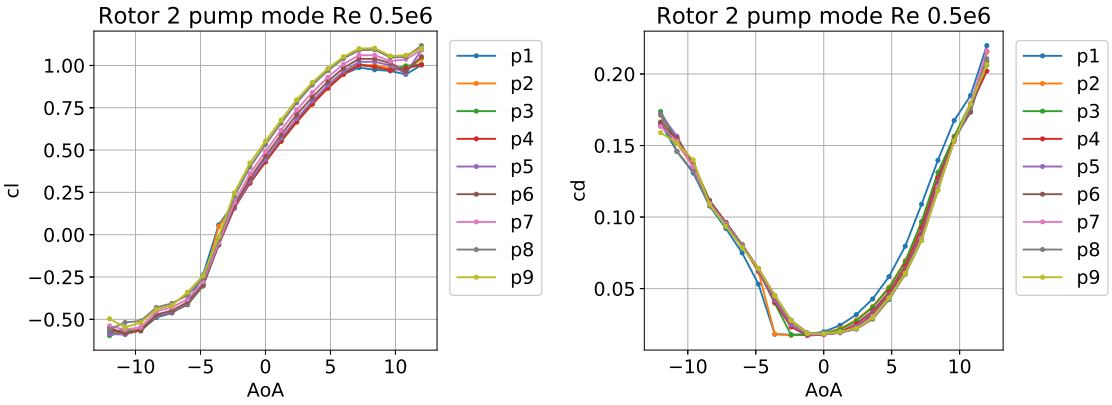


Figure B.2: Rotor 2 profile characteristics in pump mode $Re = 0.5e6$

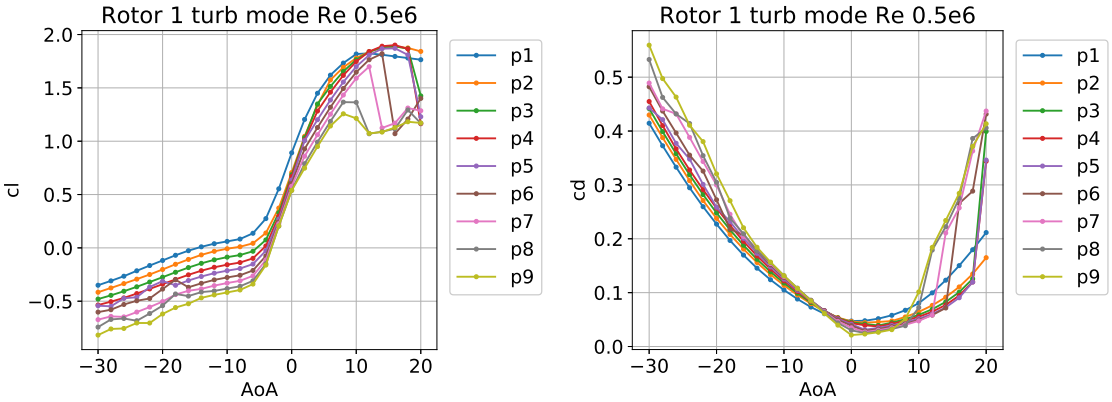


Figure B.3: Rotor 1 profile characteristics in turbine mode $Re = 0.5e6$

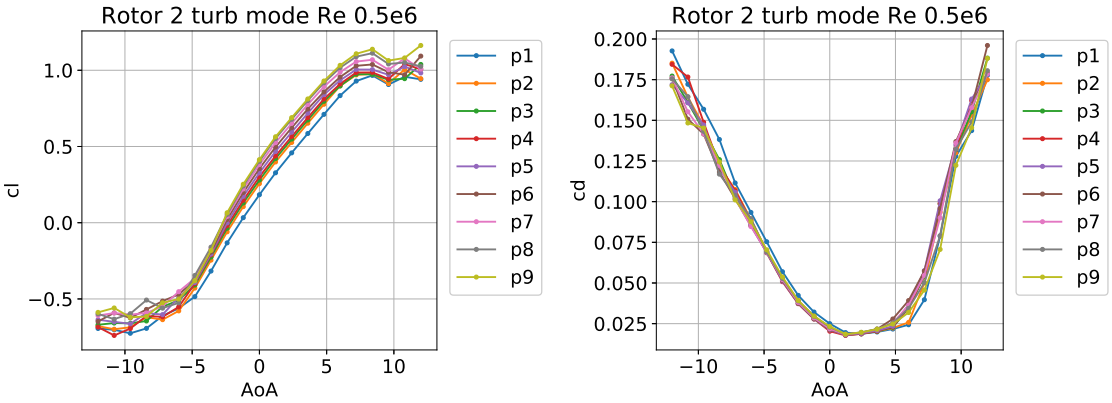


Figure B.4: Rotor 2 profile characteristics in turbine mode $Re = 0.5e6$

Re = 1.0e6

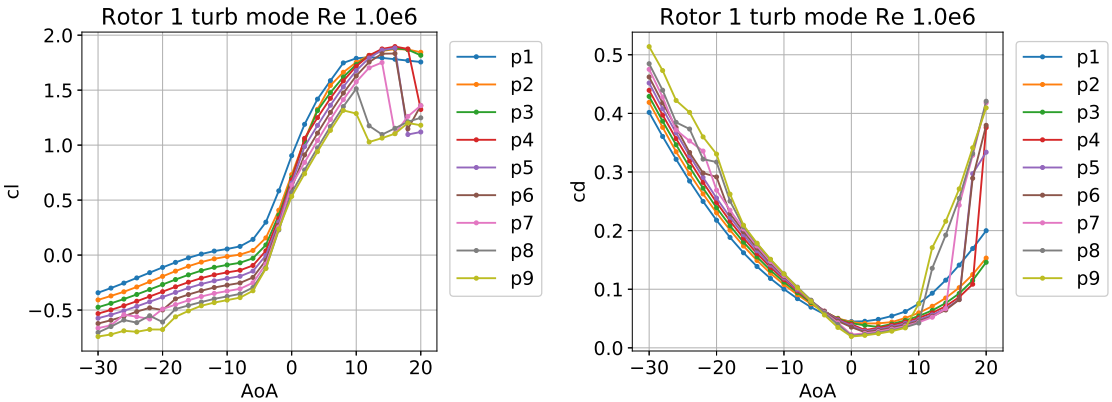


Figure B.5: Rotor 1 profile characteristics in turbine mode $Re = 1.0e6$

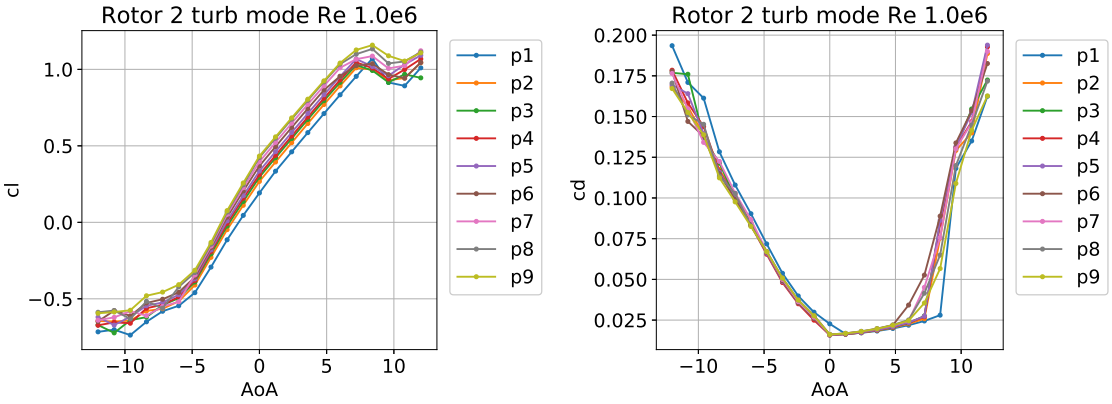


Figure B.6: Rotor 2 profile characteristics in turbine mode $Re = 1.0e6$

Re = 2.0e6

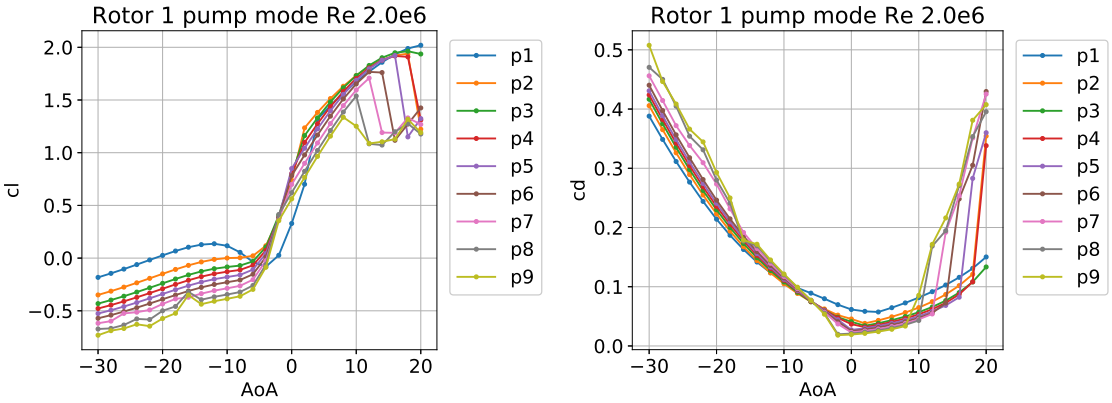


Figure B.7: Rotor 1 profile characteristics in pump mode $Re = 2.0e6$

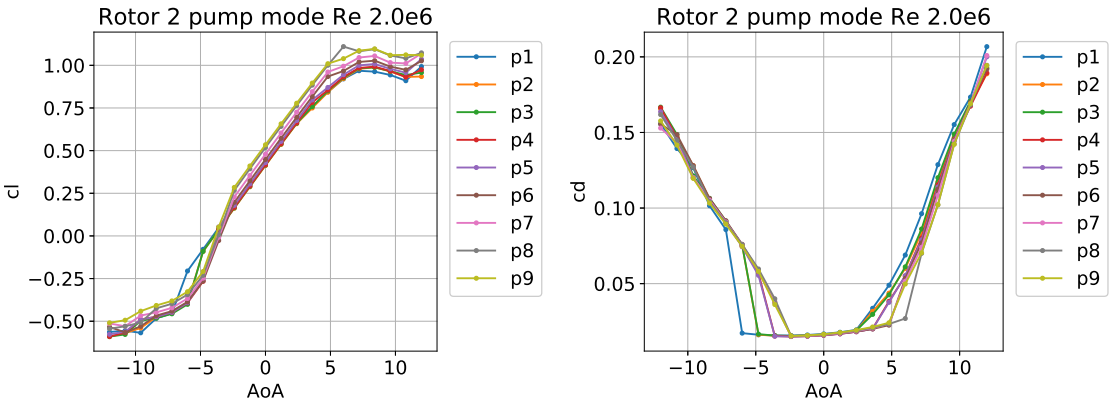


Figure B.8: Rotor 2 profile characteristics in pump mode $Re = 2.0e6$

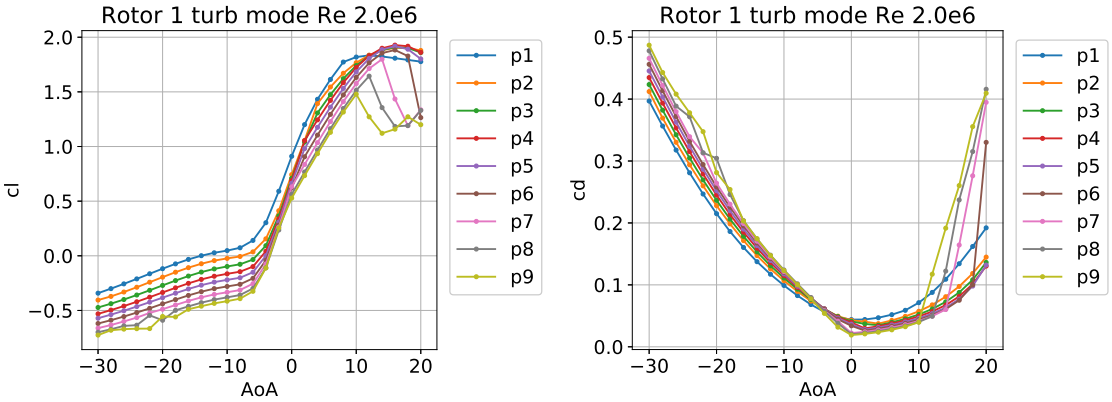


Figure B.9: Rotor 1 profile characteristics in turbine mode $Re = 2.0e6$

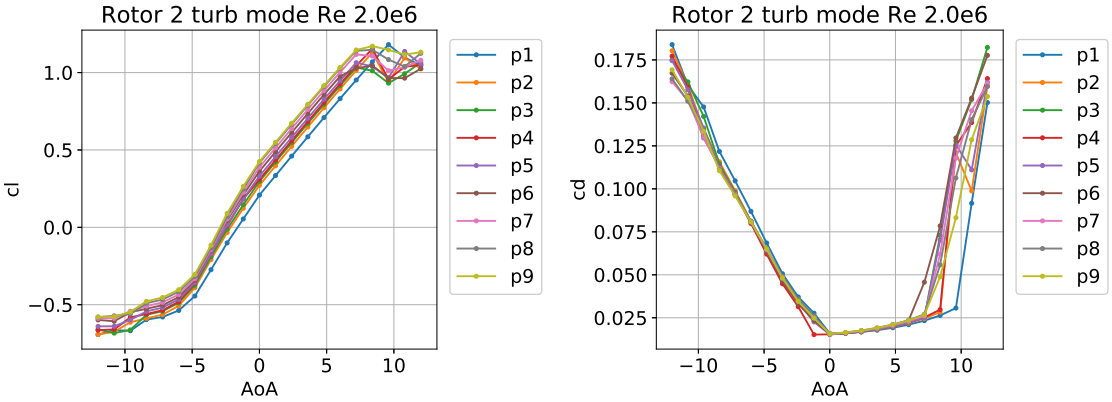


Figure B.10: Rotor 2 profile characteristics in turbine mode $Re = 2.0e6$

B.2 Polar curves extrapolated

Re = 0.5e6

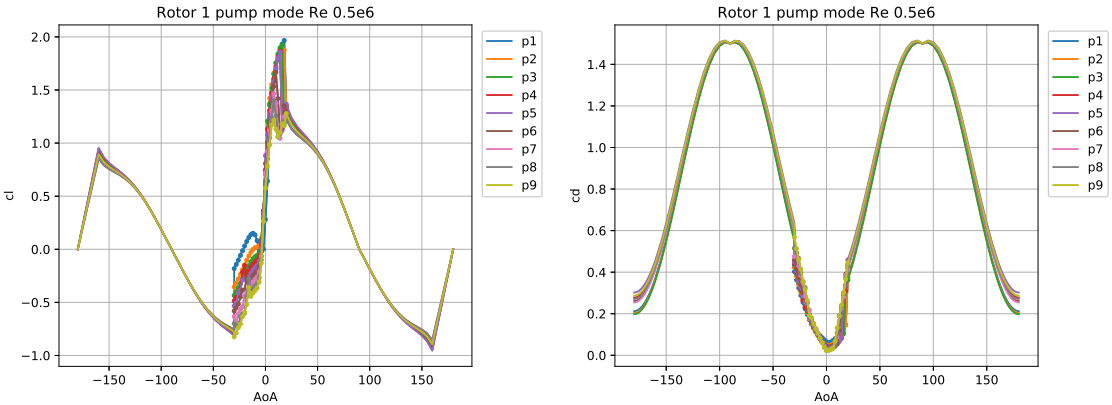


Figure B.11: Rotor 1 profile characteristics in pump mode extrapolated $Re = 0.5e6$

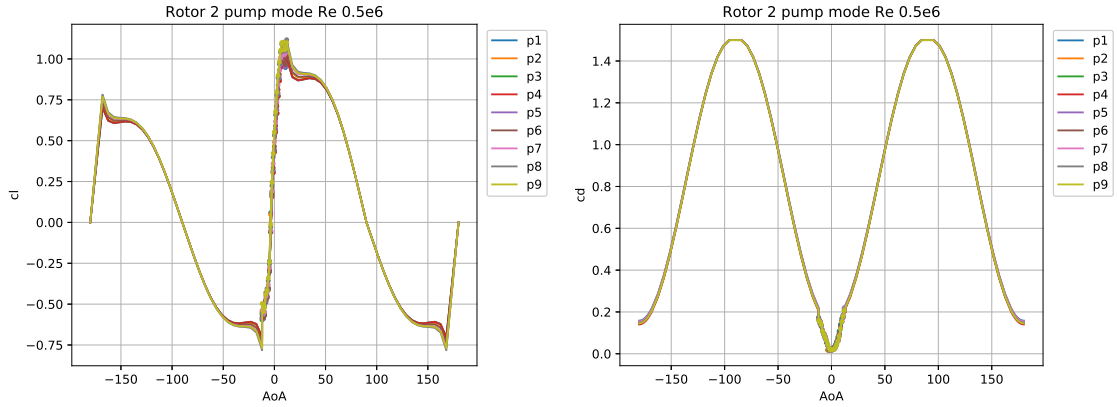


Figure B.12: Rotor 2 profile characteristics in pump mode extrapolated $Re = 0.5e6$

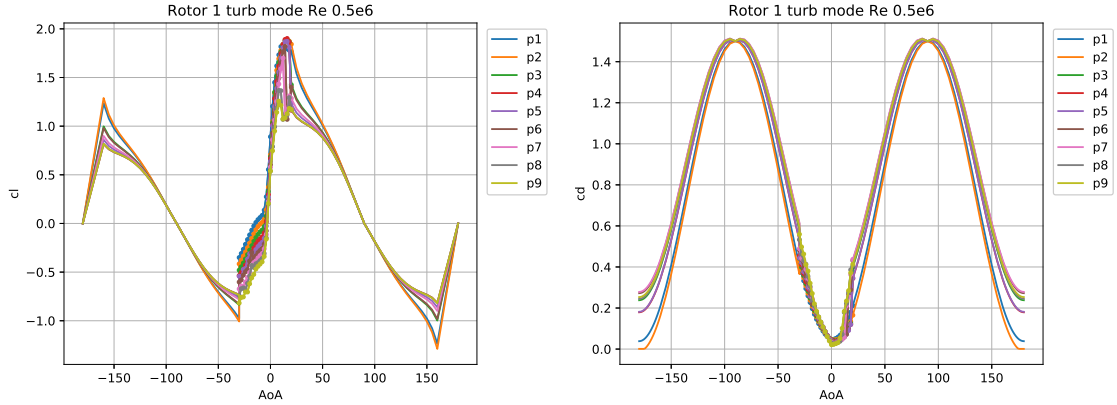


Figure B.13: Rotor 1 profile characteristics in turbine mode extrapolated $Re = 0.5e6$

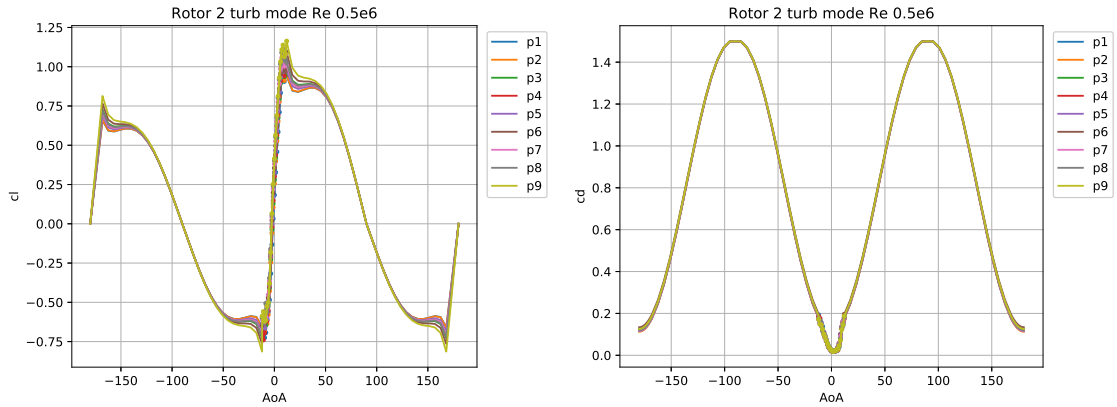


Figure B.14: Rotor 2 profile characteristics in turbine mode extrapolated $Re = 0.5e6$

Re = 1.0e6

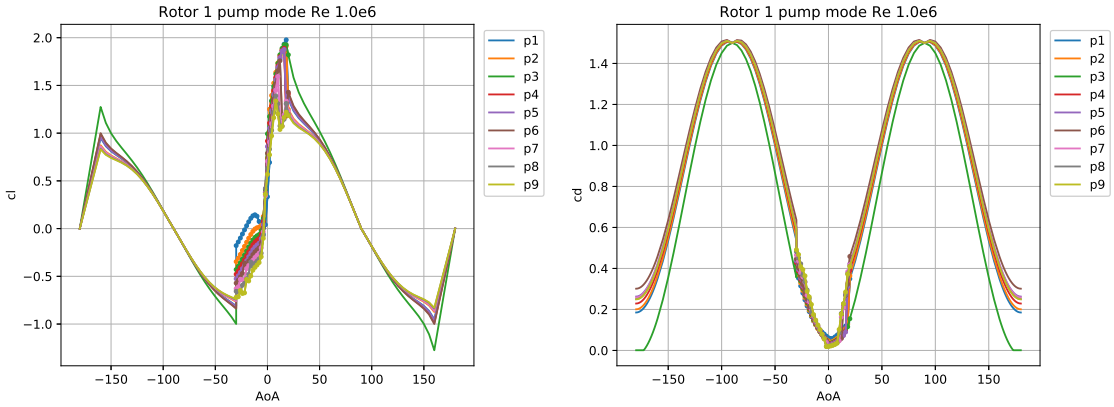


Figure B.15: Rotor 1 profile characteristics in pump mode extrapolated $Re = 1.0e6$

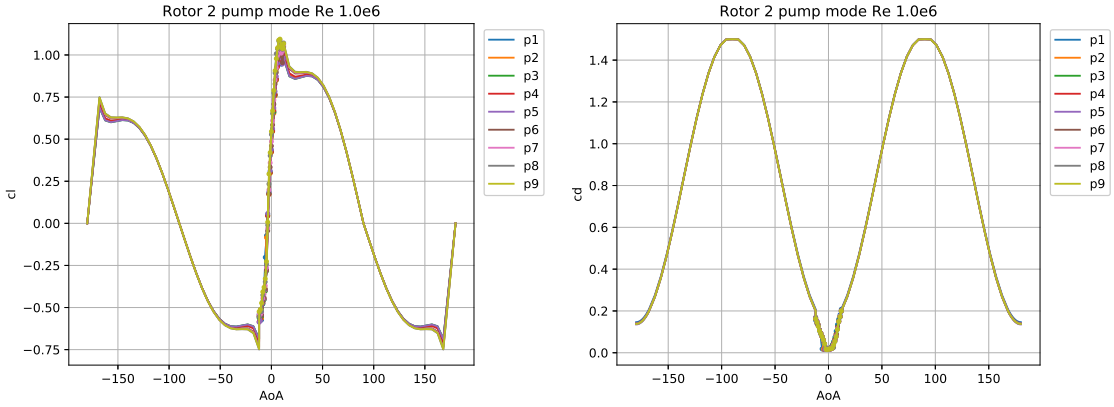


Figure B.16: Rotor 2 profile characteristics in pump mode extrapolated $Re = 1.0e6$

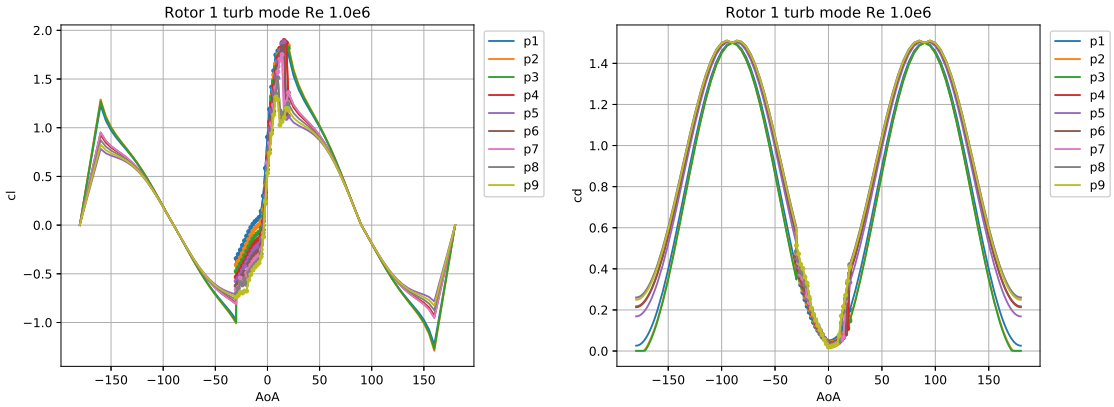


Figure B.17: Rotor 1 profile characteristics in turbine mode extrapolated $Re = 1.0e6$

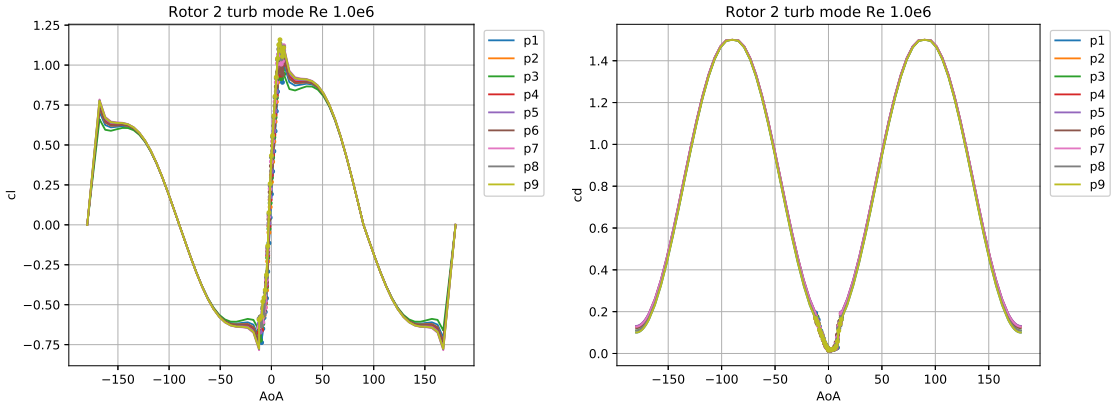


Figure B.18: Rotor 2 profile characteristics in turbine mode extrapolated $Re = 1.0e6$

Re = 2.0e6

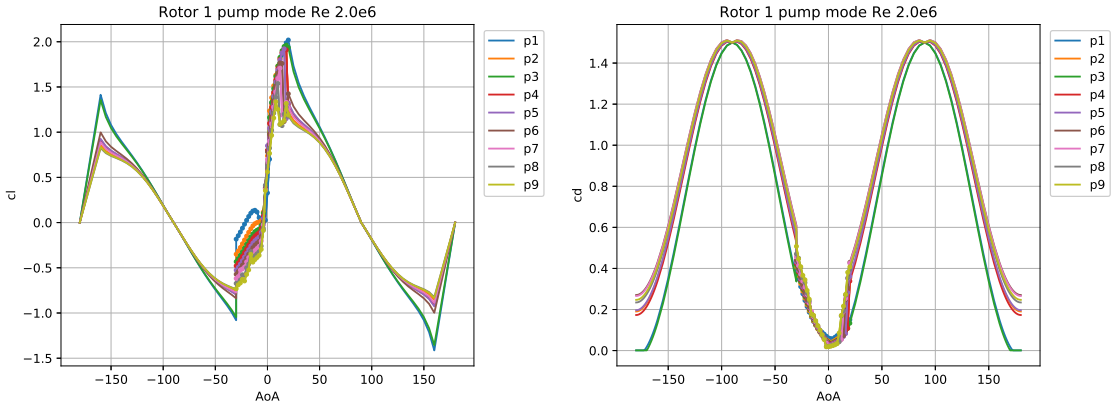


Figure B.19: Rotor 1 profile characteristics in pump mode extrapolated $Re = 2.0e6$

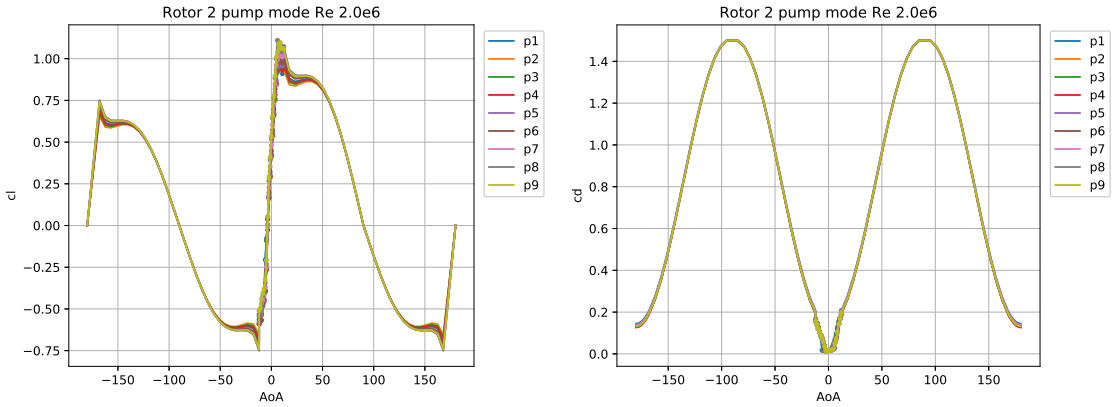


Figure B.20: Rotor 2 profile characteristics in pump mode extrapolated $Re = 2.0e6$

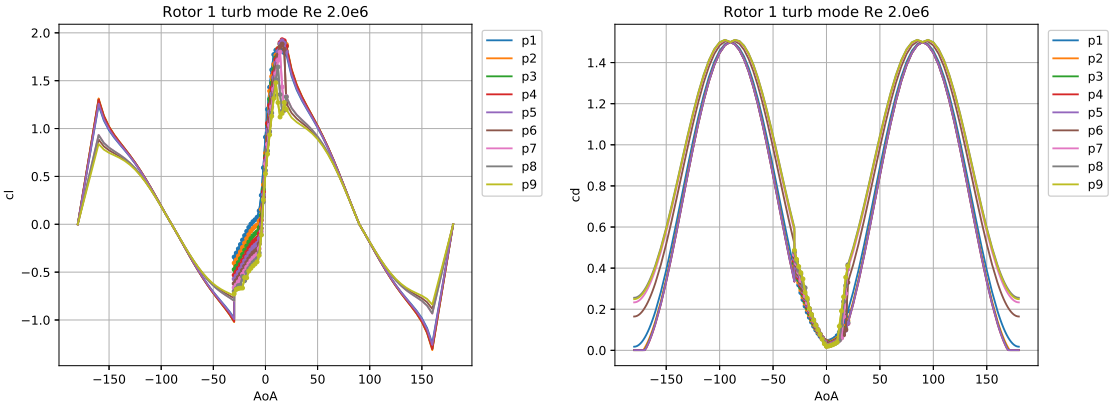


Figure B.21: Rotor 1 profile characteristics in turbine mode extrapolated $Re = 2.0e6$

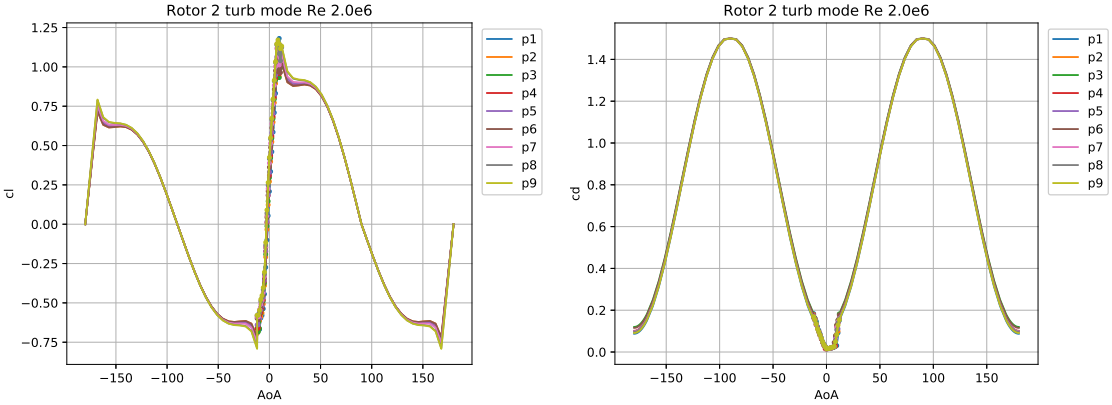


Figure B.22: Rotor 2 profile characteristics in turbine mode extrapolated $Re = 2.0e6$

C
OpenFOAM

C.1 createCases.py

This python script creates all the different individual simulations for a given *AoA*, *Re* number and operating mode. It changes the drag and lift directions respectively from the function object `forceCoeff` in `controlDict`, according to the *AoA* and therefore the flow direction.

```
import math
import numpy as np
import os
from shutil import copyfile

cwd = os.getcwd()

pi = math.pi
Re_vect = np.array([2e6]);
lRef = 1; nu = 1.0533e-6;

createdCases = "createdCases_"+ str(round(Re_vect[0]/1e6,1)) + "e6"
f = open(createdCases, "w")
f.close()

for r in range(2):
  if r+1==1:
    AoA_vect_pump = np.linspace(-30.0, 20.0, num=26) #pR1 [-30...20]
    AoA_vect_turb1 = np.linspace(-180.0, -150.0, num=16) #tR1 [-30...0]
    AoA_vect_turb2 = np.linspace(160.0, 180.0, num=10, endpoint=False) #tR1
    [0...20]
    AoA_vect = np.concatenate((AoA_vect_pump, AoA_vect_turb1, AoA_vect_turb2));
  elif r+1==2:
    AoA_vect_pump = np.linspace(-12.0, 12.0, num=21) #pR2 [-12...12]
    AoA_vect_turb1 = np.linspace(-180.0, -168.0, num=11) #tR2 [-12...0]
    AoA_vect_turb2 = np.linspace(168.0, 180.0, num=10, endpoint=False) #tR2
    [0...12]
    AoA_vect = np.concatenate((AoA_vect_pump, AoA_vect_turb1, AoA_vect_turb2));
  for p in range(9):
    RPname = "R"+str(r+1)+"P"+str(p+1)
    sourceFolder = RPname + "/cleanCase/"
    for k in range(len(Re_vect)):
      Re = Re_vect[k]
      runFolder = RPname + "/runFolder_" + str(round(Re/1e6,1)) + "e6/"
      Umag = Re*nu/lRef;
      if not os.path.exists(runFolder):
        os.makedirs(runFolder)
      for i in range(len(AoA_vect)):
        AoA_orig = AoA_vect[i]
        Ux = np.cos(AoA_orig*pi/180)*Umag
        Uy = np.sin(AoA_orig*pi/180)*Umag

        if AoA_orig < -90: #turbine angles
          AoA_eq = -180.0 - AoA_orig
          p_or_t = 't'
```

```
        elif AoA_orig > 90:
          AoA_eq = 180.0 - AoA_orig
          p_or_t = 't'
        else: #pump angles
          AoA_eq = AoA_orig
          p_or_t = 'p'

        folder = runFolder + p_or_t + "AoA_" + str(round(AoA_eq,1)) + "_Re_" + str(round(Re/1e6,1)) + "e6" + "/"

        os.system("cp -r " + sourceFolder + " "  + folder)

        os.chdir(cwd + '/' + folder)
        os.system("sed -i s/'Uinlet ('.*/'Uinlet (" +  str(Ux) + " " + str(Uy) + " 0);'/g 0/U")
        os.system("sed -i s/'dragDir ('.*/'dragDir (" +  str(Ux) + " " + str(Uy) + " 0);'/g system/controlDict")
        os.system("sed -i s/'magUInf '.*/'magUInf " + str(Umag) +  ";'/g system/controlDict")
        if np.abs(AoA_orig)<90:
          os.system("sed -i s/'liftDir ('.*/'liftDir (" +  str(-Uy) + " " + str(Ux) + " 0);'/g system/controlDict")
        elif np.abs(AoA_orig)>90:
          os.system("sed -i s/'liftDir ('.*/'liftDir (" +  str(Uy) + " " + str(-Ux) + " 0);'/g system/controlDict")

        os.system("touch ./"+p_or_t+RPname+"_"+str(round(AoA_eq,1))+".foam")

        os.system("echo case " + folder + " created")
        os.chdir(cwd)

        f = open(createdCases, "a")
        f.write(folder + "\n")
        f.close()

f = open(createdCases,"r")
lines = f.readlines()
numCases = len(lines)

os.system("echo "+ str(numCases) +" cases created")
"""
# submit job
os.system("sbatch --array=0-"+str(numCases-1)+"%60 batchFile2")
os.system("echo all "+ str(numCases) +" cases submitted")
"""
```

C.2 profile1D.csv

Example of `.csv` formate file located in `constant/` with radial profiles used as inlet for the downstream runner.

```
# volFlow = 370.0 l/s, upstream N1 = 1300.0 rpm, downstream N2 =975 rpm
# axial velocity = 7.773109243697478
#
[Data]
 R [ m ], Velocity Axial [ m s^-1 ], Velocity Radial [ m s^-1 ], Velocity Circumferential [ m s^-1 ], Pressure [ Pa ], Turbulence Kinetic Energy [ m^2 s^-2 ], Turbulence Eddy Dissipation [ m^2 s^-3 ], Turbulence Specific Dissipation Rate [ s^-1 ]
0.0623, 7.773109243697478, 0.0, 5.994472, 0.0, 0.0, 0.0, 0.0
0.064, 7.773109243697478, 0.0, 5.994472, 0.0, 0.0, 0.0, 0.0
0.0721, 7.773109243697478, 0.0, 4.725521, 0.0, 0.0, 0.0, 0.0
```

```
0.0802, 7.773109243697478, 0.0, 4.109743, 0.0, 0.0, 0.0, 0.0
0.0883, 7.773109243697478, 0.0, 3.711677999999999, 0.0, 0.0, 0.0, 0.0
0.0964, 7.773109243697478, 0.0, 3.5989599999999995, 0.0, 0.0, 0.0, 0.0
0.1046, 7.773109243697478, 0.0, 3.4128689999999997, 0.0, 0.0, 0.0, 0.0
0.1127, 7.773109243697478, 0.0, 3.336417, 0.0, 0.0, 0.0, 0.0
0.1208, 7.773109243697478, 0.0, 3.156816, 0.0, 0.0, 0.0, 0.0
0.1289, 7.773109243697478, 0.0, 3.0693310000000005, 0.0, 0.0, 0.0, 0.0
0.137, 7.773109243697478, 0.0, 4.280380999999999, 0.0, 0.0, 0.0, 0.0
0.138, 7.773109243697478, 0.0, 4.280380999999999, 0.0, 0.0, 0.0, 0.0
```

C.3 `profile1DinletValue` at `0/U`

How the boundary condition is called from in `0/U`

```
boundaryField
{
    lowPressure
    {
        type            profile1DFixedValue;
        fileName        "profile1D.csv";
        fileFormat      "turboCSV";
        interpolateCoord "R";
        fieldName       "Velocity";
        fieldScaleFactor 1;
    }
```

DEPARTMENT OF MECHANICAL AND MARITIME SCIENCES
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY