Spring 2017

Development of an Analysis and Design Optimization Framework for Marine Propellers

Ashish C. Tamhane
Old Dominion University

DEVELOPMENT OF AN ANALYSIS AND DESIGN OPTIMIZATION FRAMEWORK FOR MARINE PROPELLERS

by

Ashish C. Tamhane
B.E. July 2004, Father Conceicao Rodrigues Institute of Technology, India
M.S. December 2008, Old Dominion University

A Dissertation Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

MECHANICAL ENGINEERING

OLD DOMINION UNIVERSITY
May 2017

Approved by:

Miltiadis Kotinis (Director)

Sushil Chaturvedi (Member)

Ayodeji Demuren (Member)

Resit Unal (Member)

ABSTRACT

DEVELOPMENT OF AN ANALYSIS AND DESIGN OPTIMIZATION FRAMEWORK FOR MARINE PROPELLERS

Ashish C. Tamhane
Old Dominion University, 2017
Director: Dr. Miltiadis Kotinis

In this thesis, a framework for the analysis and design optimization of ship propellers is developed. This framework can be utilized as an efficient synthesis tool in order to determine the main geometric characteristics of the propeller but also to provide the designer with the capability to optimize the shape of the blade sections based on their specific criteria.

A hybrid lifting-line method with lifting-surface corrections to account for the three-dimensional flow effects has been developed. The prediction of the correction factors is achieved using Artificial Neural Networks and Support Vector Regression. This approach results in increased approximation accuracy compared to existing methods and allows for extrapolation of the correction factor values. The effect of viscosity is implemented in the framework via the coupling of the lifting line method with the open-source RANSE solver OpenFOAM for the calculation of lift, drag and pressure distribution on the blade sections using a transition κ-ω SST turbulence model.

Case studies of benchmark high-speed propulsors are utilized in order to validate the proposed framework for propeller operation in open-water conditions but also in a ship's wake.

ACKNOWLEDGEMENTS

I would like to thank all the people who have contributed to this endeavor. First of all, I would like to extend my gratitude to my advisor, Dr. Miltiadis Kotinis, without whose guidance and support this dissertation was simply not possible. I would also like to thank my committee members, Dr. Sushil Chaturvedi, Dr. Resit Unal, and Dr. Ayodeji Demuren for their time in serving on my dissertation committee. A special mention goes to Dr. Sebastian Bawab, our Department Chair, and Ms. Diane Mitchell.

On a personal note, I would like to thank my parents and my wife, Neha Niphadkar, for their support and encouragement throughout this endeavor. I would like to take this opportunity to thank all my friends who have made my stay at ODU such a memorable time of my life.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Ship propellers operate in the wake of the ship hull and, thus, in a flow field that is spatially non-uniform and turbulent. The main design goal for a wake-adapted propeller is the maximization of its efficiency, which has been the subject of many theoretical and experimental investigations in the past. In recent years, increased global transportation demands have necessitated the design of high-speed cargo ships with speeds in excess of 25 knots, which typically poses an additional challenge to the propeller designer: Suppress or delay cavitation inception for the widest range of operating conditions.

Theoretically, cavitation inception occurs when the local fluid pressure reaches the vapor pressure. The spatially non-uniform propeller inflow causes periodic formation, growth, and collapse of vapor-filled cavities; the latter phenomenon occurs when a cavity attached to a rotating propeller blade moves into a higher pressure region, which causes a sudden condensation of the vapor. The collapse of the cavities can produce very high pressures that can lead to surface pitting of the propeller blades. Additional undesirable effects of propeller cavitation include an increase in the propeller-induced pressure on the stern of the ship hull and a substantial increase in the noise emitted by the propeller, which in turn can have a detrimental effect on the signature of a naval vessel.

The aforementioned ship propeller design goals, i.e. propeller efficiency and cavitation-free operation, or delayed inception, are often conflicting. The latter goal necessitates blades with large

area, which increases the frictional losses and, thus, reduces the propeller efficiency. Therefore, developing blade sections that address both objectives is of paramount importance for the design of a high-speed ship propeller.

1.1 Literature Survey

The traditional approach for a propeller blade section design is to use the thickness and camber distribution of blade sections with known good cavitation performance to find the appropriate angle of attack, maximum-thickness-to-chord ratio, and maximum-camber-to-chord ratio to satisfy the ship thrust requirements and obtain sections capable of operating cavitation-free over a broad range of angles [1].

A different approach, which was originally proposed by Shen and Eppler [2, 3], is to design blade sections capable of operating cavitation-free over the widest possible range of inflow angles using a conformal mapping process where the section profile is designed based on a prescribed pressure distribution [4]. This approach has been applied by Kuiper and Jessup to the design of a wake-adapted propeller for a frigate [5], and more recently by Yamaguchi et al. for the design of the propeller blade sections of a high-speed merchant vessel [6].

The utilization of a parent (baseline) blade section as a starting design point, and a single-objective optimizer based on Lagrange multipliers combined with propeller flow analysis software have been successfully employed for the design of propeller blade sections [7, 8] and, more recently, using a multi-objective optimizer [9]. The inherent problem with classical constraint optimization methods, i.e. using gradient-based search methods, is that the foil performance characteristics, i.e.

lift and drag, need to be approximated as functions of the design variables. This issue can be overcome by employing optimizers based on evolutionary algorithms as demonstrated by Ouyang et al. [10].

Kawada [11] applied lifting line theory to a propeller by modeling each blade as a line of constant bound vorticity with a free vortex shed at the tip of the blade and an axial hub vortex with a strength equal to the sum of the strengths of the tip vortices. Lerbs' seminal work presented in [12] was the first attempt to analyze a moderately loaded propeller using lifting line theory by considering bound vorticity radially varying along the lifting line. In this way, the axial and tangential induced velocity components can be evaluated independently, thus, removing the assumption that the vortex sheets emanating from each blade are of true helical form; an assumption applicable only to lightly loaded propellers.

The lifting surface propeller analysis approach developed by Pien [13] models the propeller as an infinitely thin surface with a continuous distribution of vorticity along the spanwise direction but also along the chordal directions to account for the blade section camber. Later lifting surface models, like the one developed by Brockett [14], introduced a distribution of sources and sinks in the chordal directions at various radial positions to account for the blade section thickness.

A subclass of the lifting surface method is the vortex lattice approach proposed by Kerwin [15]. The characteristic difference being that the vortex lattice method employs a set of discrete straight-line segments of vortices of constant strength whose end points are located along the blade camber surface in order to form a system of vortex panels rather than the continuous distribution of

singularities used in the classical lifting surface approach. The method was developed further by Greeley and Kerwin [16] to account for viscous effects and cavitation inception.

Griffin and Kinnas [17] coupled the aforementioned vortex lattice cavitating propeller analysis method with the nonlinear blade section optimization method developed in [7, 8] and utilized a cubic B-spline polygon net to represent the propeller blade in order to develop a design method for high-speed propulsor blades. More recently, boundary element methods (BEM) have been utilized to solve the unsteady cavitating flow around propellers subject to non-uniform inflow conditions [18].

A hybrid lifting-line lifting-surface method was introduced by van Oossanen [19], where Lerbs' method [12] is supplemented by an additional iteration that takes into account the blade section geometry. The effects of a non-uniform inflow are incorporated into the model by determining the advance angle at each blade position within the propeller disc. The effect of viscosity is also implemented through the calculation of the boundary layer flow along the blade section, which increases the prediction accuracy at off-design conditions. Van Oossanen also utilized the lifting surface correction factors calculated by Morgan et al. [20] for propellers with moderate skew and by Cumming et al. [21] for highly-skewed propellers. These correction factors account for the impact of the three-dimensional flow to the camber and ideal angle of the blade section. They also include the effect of the blade thickness. Van Oossanen developed a polynomial representation of these correction factors, which is also available in [19]. A lifting line method combined with a Lagrange multiplier optimization method has recently been proposed as an efficient propeller design and analysis tool [22].

Choi and Kinnas [23] attempted to calculate the effective wake of a ship, i.e. the ship wake with an operating propeller; a task that requires the modeling of the ship-propeller interaction. They used a vortex lattice solver for the propeller analysis coupled with a solver for the Euler flow equations (the effect of the fluid viscosity is neglected). The coupling is achieved by representing the propeller impact on the ship wake via body forces computed by the vortex lattice solver and included in the Euler flow equations. A similar investigation, but using the vortex lattice method developed in [15] for the propeller calculations, coupled with a Reynolds-averaged Navier-Stokes equations (RANSE) solver was performed by Stern et al. [24, 25]. A detailed discussion regarding theoretical developments of ship propeller analysis and design methods can be found in [26].

In recent years, the coupling of cavitation prediction models with RANSE solvers has allowed researchers to obtain more accurate results regarding the cavitation inception of propeller blades. Singhal et al. [27] developed such a cavitation prediction model, named the "full cavitation model," which is based on the bubble dynamics equation derived from the generalized Rayleigh-Plesset equation [28]. The model has been used to predict the cavitation of a marine propeller by Arazgaldi et al. [29]. Huang et al. [30] have implemented the "full cavitation model" in the commercial Computational Fluid Dynamics (CFD) software FLUENT to solve the 2-D hydrofoil cavitating problem. Delgosha et al. [31] have developed cavitation prediction models based on the barotropic state law that links the fluid density to the local static pressure [32].

Scaling effects on cavitation inception and development have been investigated by Astolfi et al. [33] and Amromin [34]. Astolfi et al. also examined the impact of boundary layer separation on cavitation inception on a 2-D hydrofoil.

1.2 Summary and Objectives of the Proposed Research

The main objectives of this dissertation are, first, to develop an efficient methodology for the analysis and design of propeller blades, and second, to develop a method for the design of blade sections with low drag and a wide range of cavitation-free operation.

Regarding the first objective, the hybrid lifting-line lifting-surface method of van Oossanen is utilized as the baseline model and a number of improvements are implemented in order to increase its accuracy. Specifically, the lifting-surface correction factors were represented in [19] by polynomials derived via multiple regression analysis. In this work, artificial neural networks (ANNs) and support vector regression (SVR) are utilized to derive new surrogate models for the lifting-surface correction factors. It is demonstrated that these new models have superior performance compared to the polynomial representation and could be utilized not only for interpolation but also for extrapolation purposes.

The effect of viscosity is implemented in the hybrid lifting-line lifting-surface method by calculating the lift, drag, and pressure coefficients of the blade sections using the open-source RANSE solver OpenFOAM [35]. In this way, an accurate prediction of the performance of the blade sections even at off-design operating conditions can be obtained. The proposed propeller analysis and design framework has been coded using the scientific programming language GNU Octave [36].

As already mentioned, the design goals for a high-speed ship propeller, i.e. propeller efficiency and cavitation-free operation, are often conflicting. The typical solution of increasing the blade

area to limit the extent of cavitation results in increased frictional losses and, thus, reduces the propeller efficiency.

In this work, a design method for blade sections with low drag and wide range of cavitation free operation is proposed. The blade section is modeled using cubic B-splines and the corresponding control points are used as the design variables of a multi-objective constrained optimization problem. The optimizer is based on swarm intelligence principles; the objectives and constraints of the problem that correspond to the performance of the blade sections are calculated using OpenFOAM and modeled using ANNs. It is demonstrated that this approach provides the designer with a set of solutions that correspond to trade-offs between the design objectives. A selection can then be made based on the designer's preferences.

In Chapter 2, the development of the hybrid lifting-line lifting-surface model is described, followed by the derivation and validation of new models for the lifting-surface correction factors in Chapter 3. The calculation of the blade section lift, drag, and pressure distribution is performed using a RANSE solver in OpenFOAM. The solver, the utilized turbulence model, and the corresponding computational grid are discussed in detail in Chapter 4. In that chapter, a grid verification study is included, as well as validation case studies, and a performance analysis of the baseline blade sections utilized in the blade design method. The latter is described in Chapter 5, including a brief description of the multi-objective optimizer and the utilization of cubic B-splines for the representation of the blade sections. The optimization results and the performance analysis of a selected optimal blade section are also discussed in that chapter. Validation of the propeller analysis method for a number of ship propellers operating in open-water conditions is provided in

Chapter 6 along with the results of a case study for operation of the propeller in the wake of a high-speed container ship. The effectiveness of the optimal blade sections designed in Chapter 5 is also demonstrated as part of the case study. Conclusions and directions for future research are provided in Chapter 7.

CHAPTER 2

A HYBRID LIFTING-LINE LIFTING-SURFACE MODEL FOR A MODERATELY

LOADED PROPELLER

2.1 Introduction

The foundation of most mathematical models of propeller action is Prandtl's lifting line theory. This theory states that the lift of a wing, or of a propeller blade, is due to the development of circulation around the section in accordance with the Kutta-Joukowski law for lifting surfaces. Therefore, the blade can be represented as a single line vortex whose strength varies in the radial direction. This lifting line is considered to pass through the aerodynamic centers of the blade sections on which the circulation is distributed and to coincide with the radial coordinate *r* of a cylindrical coordinate system with the axial coordinate being along the direction of the propulsion shaft. As the circulation is varied along this line, a free vortex is shed between radial stations *r* and *r+dr*. In order to satisfy Helmholtz' vortex theorem, the shed vorticity is related to the bound vorticity of the lifting line as follows:

$$\bar{\Gamma}(r) = (\frac{\mathrm{d}\Gamma}{\mathrm{d}r})\mathrm{d}r \tag{2.1}$$

where $\bar{\Gamma}$ is the circulation of the free vortex line and Γ is the bound circulation. This free vortex line has a general helical shape.

In this work, the lifting line theory is applied in the inverse sense, i.e. to determine the loading on the propeller when the geometric parameters of the propeller and the velocity field in which the propeller operates are given. The velocity field, which corresponds to the ship wake, can vary both

radially and circumferentially. The proposed method is based on the quasi-linear approach for a moderately loaded propeller introduced by van Oossanen [19].

2.2 Calculation of the Induced Velocities

The utilization of Prandtl's lifting line theory for the design and analysis of moderately loaded propellers was originally introduced by Lerbs [12]. A concise description of Lerbs' proposed method is provided in this section. A GNU Octave script has been developed in order to implement the method and is available in Appendix A.

The starting point of the method is Laplace's equation, which can be utilized to obtain the velocity potential of the flow outside the vortex space. Kawada [11] considered a propeller with g symmetrically placed blades and a horseshoe vortex model. Each blade is represented by a line of constant bound vorticity from the hub to the tip. At the tip of each blade a free helical vortex emanates. Kawada also considered an axial hub vortex with strength equal to the sum of the tip vortex strengths. For the description of the flow, a cylindrical coordinate system is utilized with z: axial direction, θ: tangential direction, and r: radial direction, such that at the propeller disk, $z = 0$ and the z-axis is positive in the direction of the flow. The tangential coordinate θ is zero at the first lifting line and is positive in the direction of rotation of the propeller. The velocity potential for an infinitely long symmetrical system of g helical vortices with constant radius r_0 and pitch angle β_{i0}, can be expressed as follows:

For internal radial points ($r < r_0$),

$$\varphi_i = \frac{g}{2}\,\bar{\Gamma} + \frac{g}{2\pi}\,\bar{\Gamma}\left\{\frac{z}{\Bbbk_o} + 2\frac{r_{\mathfrak{d}}}{\Bbbk_o}\sum_{n=1}^{\infty} I_{ng}\left(\frac{ng}{\Bbbk_o}\,r\right)K_{ng}^{\iota}\left(\frac{ng}{\Bbbk_o}\,r_{\mathfrak{d}}\right)\sin(ng\vartheta)\right\}$$

(2.2a)

and for external points ($r > r_0$),

$$\varphi_e = \frac{\bar{\Gamma}}{2\pi}\left\{g\theta - 2\pi\sum_{m=1}^{g}\frac{m-1}{g} + 2g\frac{r_{\mathfrak{d}}}{\Bbbk_o}\sum_{n=1}^{\infty}K_{ng}\left(\frac{ng}{\Bbbk_o}\,r\right)I_{ng}^{\iota}\left(\frac{ng}{\Bbbk_o}\,r_{\mathfrak{d}}\right)\sin(ng\vartheta)\right\}$$

(2.2b)

where I and K are the modified Bessel functions of the first and second kind, respectively, and the prime means derivative with respect to the argument. Also,

$$\Bbbk_o = r_{\mathfrak{d}}\tan\beta_{io}$$ (2.3)

$$\vartheta = \theta - \frac{z}{\Bbbk_o}$$ (2.4)

The axial and tangential induced velocities can be calculated by differentiating this potential function. For ease of evaluation, the modified Bessel functions and their derivatives are replaced by Nicholson's asymptotic functions [37]:

$$I_n(ny) = \left[\frac{1}{2\pi n\sqrt{1+y^2}}\right]e^{nY}$$ (2.5a)

$$K_n(ny) = \left[\frac{\pi}{2n\sqrt{1+y^2}}\right]e^{-nY}$$ (2.5b)

$$Y = \sqrt{1+y^2} - \frac{1}{2}\log.\frac{\sqrt{1+y^2}+1}{\sqrt{1+y^2}-1}$$ (2.6)

The potential function derived by Kawada satisfies Helmholtz'vortex theorem: the vortex filaments of the helical vortices must extend to infinity in both directions. However, in propeller

flows, the vortices start at $z = 0$ and extend to infinity only in the positive z-direction. Lerbs showed

that the induced velocity expressions derived by Kawada are valid for points at the propeller disc

($z = 0$) provided that the induced axial and tangential velocity components are divided by two.

Lerbs also removed Kawada's assumption of constant bound vorticity and considered radially

varying bound vorticity, which results in the production of a vortex sheet with varying strength

(see Equation 2.1).

Considering points on the lifting lines ($z = 0$), the axial and tangential velocity components

induced at a radial position r by a free helical vortex emanating at position r_0 can be found to be,

$$U_{ai} = \frac{1}{k_o}\frac{g\overline{\Gamma}}{4\pi}(1 + B_2) \qquad U_{ae} = -\frac{1}{k_o}\frac{g\overline{\Gamma}}{4\pi}B_2 \tag{2.7a}$$

$$U_{ti} = -\frac{1}{k_o}\frac{g\overline{\Gamma}}{4\pi}B_1 \qquad U_{te} = \frac{1}{k_o}\frac{g\overline{\Gamma}}{4\pi}(1 + B_1) \tag{2.7b}$$

where,

$$B_{1,2} = \left(\frac{1+y_0{}^2}{1+y^2}\right)^{0.25}\left[\frac{1}{e^{A_{1,2}}-1} \mp \frac{1}{2g}\frac{y_0{}^2}{(1+y_0{}^2)^{1.5}}\ln\left(1 + \frac{1}{e^{A_{1,2}}-1}\right)\right]$$

$$A_{1,2} = \pm\left(\sqrt{1+y^2} - \sqrt{1+y_0{}^2}\right) \mp \frac{1}{2}\ln\left[\frac{\left(\sqrt{1+y_0{}^2}-1\right)\left(\sqrt{1+y^2}+1\right)}{\left(\sqrt{1+y_0{}^2}+1\right)\left(\sqrt{1+y^2}-1\right)}\right]$$

$$y_0 = \frac{r_0}{k_o} = \frac{1}{\tan\beta_{i0}} \qquad\qquad y = \frac{r}{k_o} = \frac{x}{x_0\tan\beta_{i0}}$$

$$x = \frac{r}{R} \qquad\qquad x_0 = \frac{r_0}{R}$$

and x is the radial position r nondimensionalized by the propeller radius R.

When applying those equations for the calculation of the induced velocities at $r = r_0$, the velocity

tends to infinity. This severely impairs the numerical results; in order to resolve this issue, Lerbs

utilized the concept of the induction factor proposed by Kawada [38]. The induction factor is

defined as the ratio of velocity induced at a point r on the lifting line due to a helical vortex at r_0 and the velocity induced by a straight line vortex at r_0 parallel to the z axis:

$$i_a = U_a \Big/ \frac{\bar{\Gamma}}{4\pi(r-r_0)}, \qquad i_t = U_t \Big/ \frac{\bar{\Gamma}}{4\pi(r-r_0)} \tag{2.8}$$

where i_a and i_t are the induction factors for the induced axial and tangential velocity, respectively.

At $r = r_0$, the velocity induced by the straight line vortex also tends to infinity and thus the ratio, i.e. the induction factor, remains finite. By combining Equations 2.7 and 2.8, the mathematical relations for the induction factors are found to be,

$$i_{ai} = g \frac{x}{x_0 \tan \beta_{io}} \left(\frac{x_0}{x} - 1\right)(1 + B_2) \tag{2.9a}$$

$$i_{ae} = -g \frac{x}{x_0 \tan \beta_{io}} \left(\frac{x_0}{x} - 1\right) B_1 \tag{2.9b}$$

$$i_{ti} = g \left(\frac{x_0}{x} - 1\right) B_2 \tag{2.9c}$$

$$i_{te} = -g \left(\frac{x_0}{x} - 1\right)(1 + B_1) \tag{2.9d}$$

Since the propeller wake is a continuous sheet of helical vortices of radially varying strength emanating along the lifting line between the hub (with radial location x_h) and the blade tip ($x = 1$), the total velocity induced by such a sheet at a radial location r is,

$$\frac{U_{a,t}}{V_a} = \frac{1}{2} \int_{x_h}^{1} \left[\frac{dG}{dxo} \frac{1}{(x-x_0)} i_{a,t}\right] dx_0 \tag{2.10}$$

where V_a is the average axial inflow velocity along the lifting line, and G is the non-dimensional circulation given by the formula,

$$G = \frac{\Gamma}{\pi D V_a} \tag{2.11}$$

It is important to note that in a circumferentially varying inflow, $U_{a,t}$, V_a, G, and β_i are functions of r and θ. Therefore, the induction factors, being functions of the hydrodynamic pitch angle β_i, are also functions of θ.

Equation 2.10 is an improper integral and can be numerically evaluated using Glauert's method [39]. The radial variable x is replaced by a new variable φ such that,

$$x = \frac{1}{2}(1 + x_h) - \frac{1}{2}(1 - x_h)\cos\varphi \tag{2.12}$$

It is observed from Equation 2.12 that at $x = x_h$, $\varphi = 0$, and at $x = 1$, $\varphi = \pi$. Next, G is resolved into a Fourier series,

$$G(\varphi, \theta) = \sum_{m=1}^{\infty} G_m(\theta) \sin(m\varphi) \tag{2.13}$$

As equilization of pressures takes place at the hub and at the tip, it is observed that this Fourier series correctly predicts G at both positions to be equal to zero. The induction factors are represented by an even Fourier series,

$$i_{a,t}(\varphi, \varphi_0) = \sum_{n=0}^{\infty} I_n^{a,t}(\varphi) \cos(n\varphi_0) \tag{2.14}$$

Substituting Equations 2.13 and 2.14 into Equation 2.10 we get,

$$\frac{U_{a,t}(\varphi,\theta)}{V_a(\theta)} = \frac{1}{1 - x_h} \sum_{m=1}^{\infty} m G_m(\theta) h_m^{a,t}(\varphi) \tag{2.15}$$

where,

$$h_m^{a,t}(\varphi) = \int_0^\pi \frac{i_t(\varphi,\varphi_0)\cos m\,\varphi_0}{\cos\varphi_0 - \cos\varphi} \, d\varphi_0 =$$

$$\frac{1}{2}\sum_{n=0}^{\infty} I_n^{a,t}(\varphi) \left\{ \int_0^\pi \frac{\cos(m+n)\varphi_0}{\cos\varphi_0 - \cos\varphi} d\varphi_0 + \int_0^\pi \frac{\cos(m-n)\varphi_0}{\cos\varphi_0 - \cos\varphi} d\varphi_0 \right\} \tag{2.16}$$

The integrals in this expression are called the Glauert integrals with the known principal value of,

$$h_m^{a,t} = \frac{\pi}{\sin \varphi} \left[\sin(m\varphi) \sum_{n=0}^{\infty} I_n^{a,t}(\varphi) \cos(n\varphi) + \cos(m\varphi) \sum_{n=m+1}^{\infty} I_n^{a,t} \sin(n\varphi) \right]$$

$$(2.17)$$

Equation 2.17 becomes indefinite at the end points, i.e. $\varphi = 0$ and $\varphi = \pi$, and hence L'Hospital's rule is used to evaluate $h_m^{a,t}$ at these points,

$$h_m^{a,t}(0) = \pi \left[m \sum_{m=1}^{\infty} I_n^{a,t}(0) + \sum_{n=m+1}^{\infty} n I_n^{a,t}(0) \right] \qquad (2.18a)$$

$$h_m^{a,t}(\pi) = -\pi \cos(m\ 180°) \Big[m \sum_{n=0}^{m} I_n^{a,t}(\pi) \cos(n\pi) +$$

$$\sum_{n=m+1}^{\infty} n I_n^{a,t}(\pi) \cos(n\pi) \Big] \qquad (2.18b)$$

Figure 2.1: Velocities and Forces on Propeller Blade Section.

The hydrodynamic pitch angle β_i is calculated as follows (see Figure 2.1):

$$\tan \beta_i(x, \theta) = \frac{\bar{V}_a(x,\theta) + U_a(x,\theta)}{x\pi nD - \bar{V}_t(x,\theta) - U_t(x,\theta)} \qquad (2.19)$$

where $\bar{V}_a$ and $\bar{V}_t$ are the average axial and tangential velocity components between the leading edge and the trailing edge of the blade section at (x, θ).

The angle of attack with respect to the chord line of the blade section is defined as:

$$a_{NT} = \tan^{-1}\left[\frac{P(x)}{x\pi D}\right] - \beta_i(x,\theta) \tag{2.20}$$

A first value of β_i can be estimated by Sontvedt's empirical relation [40] :

$$\beta_i(x,\theta) = \gamma_0(x,\theta) - [\gamma_0(x,\theta) - \beta(x,\theta)]\left[0.135 + \frac{0.053}{1.093-x}\right] \tag{2.21}$$

where,

$$\beta(x,\theta) = \tan^{-1}\frac{\bar{V}_a(x,\theta)}{x\pi nD - \bar{V}_t(x,\theta)} \tag{2.22}$$

is the advance angle,

$$\gamma_0(x,\theta) = \tan^{-1}\left[\frac{P(x)}{x\pi D}\right] + a_0(x,\theta) \tag{2.23}$$

is the effective pitch angle, defined schematically in Figure 2.2,

$P(x)$: pitch of the propeller at x,

D : diameter of the propeller, and

α_0 : the value of the angle of zero lift.
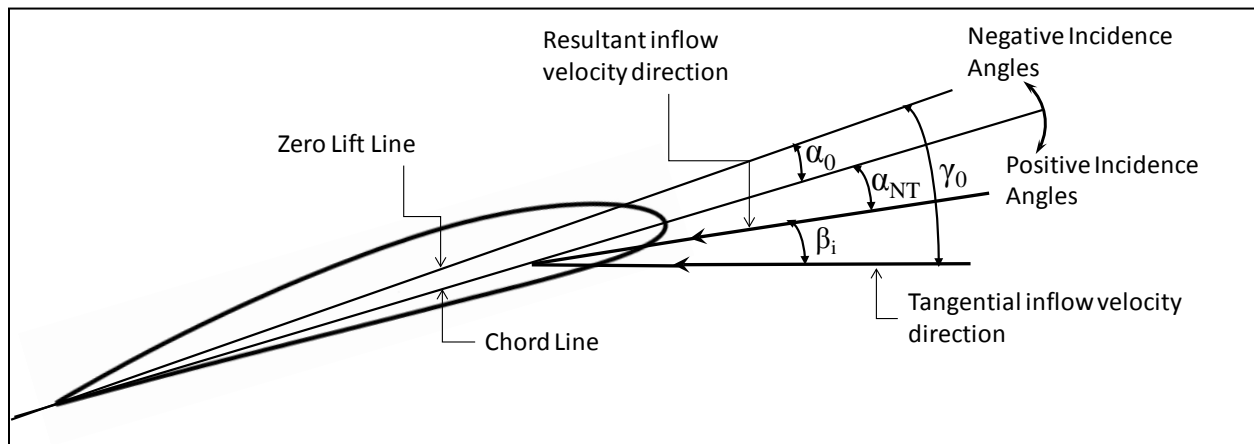
Figure 2.2: Schematic Representation of the Inflow Angles.

With the initial values of β_i calculated from Equation 2.21, the induction factors can then be calculated from Equation 2.9. These values are substituted into Equation 2.14, which produces a system of linear algebraic equations with the Fourier series coefficients as the unknown variables. The number of Fourier coefficients utilized depends on the number of radial stations considered in the analysis. Here, nine stations are considered and so the Fourier series expansion is truncated at $n = 9$. The solution of the aforementioned system of equations produces the values of the Fourier coefficients which are then substituted into Equation 2.18 to obtain the values of h_m; those are then utilized in Equation 2.15. This equation still has two unknowns: the induced velocity and the non-dimensional circulation, and thus an additional equation is required.

Lerbs suggested using Equation 2.19 as the second equation in the following form:

$$\frac{V_a(\theta)}{\overline{V_a}(x,\theta)}\frac{U_a(x,\theta)}{V_a(\theta)} + \frac{V_a(\theta)}{\overline{V_a}(x,\theta)}\frac{U_t(x,\theta)}{V_a(\theta)}\tan\beta_i(x,\theta) = \frac{\tan\beta_i(x,\theta)}{\tan\beta(x,\theta)} - 1 \qquad (2.24)$$

which is combined with Equation 2.15 to provide the following system of equations:

$$\frac{1}{1-x_h}\sum_{m=1}^{\infty} mG_m(\theta)[h_m^A(\phi) + \tan\beta_i(\phi,\theta)\,h_m^T(\varphi)]\frac{V_a(\theta)}{\overline{V_a}(x,\theta)} =$$

$$\frac{\tan\beta_i(x,\theta)}{\tan\beta(x,\theta)} - 1 \qquad (2.25)$$

from which the value of G_m at the nine radial stations can be calculated. The calculated values are then substituted into Equation 2.15 in order to obtain the induced velocities.

2.3 Calculation of the Propeller Thrust and Torque

Considering that the induced velocities were obtained with an estimated initial value from Equation 2.21, an iterative process is required in order to obtain the final values. Van Oossanen

suggested using an equation that incorporates the geometric parameters of the propeller blade sections. Starting with the Kutta-Joukowski law,

$$\Gamma(x,\theta) = \frac{L(x,\theta)}{\rho V(x,\theta)} \tag{2.26}$$

where $L(x, \theta)$: lift force of the blade section given by,

$$L(x,\theta) = C_L(x,\theta)\frac{1}{2}\rho V^2(x,\theta)c(x) \tag{2.27}$$

$V(x, \theta)$: resultant velocity at the blade section (see Figure 2.1), and

ρ : density of water.

Equation 2.27 is substituted into Equation 2.11,

$$G(x,\theta) = \frac{C_L(x,\theta)V(x,\theta)c(x)}{2\pi DV_a(\theta)} \tag{2.28}$$

The resultant inflow velocity at each considered location can be expressed in terms of its components as,

$$V(x,\theta) = \frac{\overline{V_a}(x,\theta)+U_a(x,\theta)}{\sin\beta_i(x,\theta)} \tag{2.29}$$

and hence Equation 2.28 becomes,

$$G(x,\theta) = \frac{C_L(x,\theta)c(x)\left[\frac{\overline{V_a}(x,\theta)}{V_a(\theta)}+\frac{U_a(x,\theta)}{V_a(\theta)}\right]}{2\pi D \sin\beta_i(x,\theta)} \tag{2.30}$$

where C_L is the three-dimensional lift coefficient value and is related to the two-dimensional lift coefficient by means of the corresponding lift-curve slopes,

$$\frac{dC_L}{d\alpha} = \frac{dC_L}{d\alpha_2}\frac{1}{K_c} \tag{2.31}$$

where,

$dC_L/d\alpha$: three-dimensional lift-curve slope, which can be calculated as,

$$\frac{dC_L}{d\alpha}(x,\theta) = \frac{C_L(x,\theta)}{\gamma_0(x,\theta)-\beta_i(x,\theta)} \tag{2.32}$$

$dC_L/d\alpha_2$: two-dimensional lift curve slope, and

K_c : lifting surface correction factor for camber (obtained using the method described in Chapter III). In this work, the value of $dC_L/d\alpha_2$ is calculated using the CFD method described in Chapter IV. Combining Equations 2.31 and 2.32,

$$C_L(x,\theta) = \frac{dC_L}{d\alpha_2}\frac{1}{K_c}\left(\gamma_0(x,\theta) - \beta_i(x,\theta)\right) \qquad (2.33)$$

The value of the three-dimensional zero lift angle α_0 needed to calculate the effective pitch angle (see Equation 2.23 and Figure 2.2) can be calculated as:

$$\alpha_0(x,\theta) = \alpha_{0_2}(x,\theta) + \alpha_i(x,\theta) - \alpha_{i_2}(x,\theta) \qquad (2.34)$$

where,

α_{0_2}: two-dimensional angle of zero lift, which is obtained using the CFD method described in Chapter 4,

α_i: three-dimensional ideal angle of incidence, which is evaluated by employing the lifting surface correction factors (provided in Chapter 3) for the two-dimensional ideal angle of incidence and the correction factor for blade thickness (K_α and K_t, respectively) via the following equation,

$$\alpha_i(x,\theta) = K_\alpha\alpha_{i_2}(x,\theta) + K_t BTF_x \qquad (2.35)$$

α_{i_2}: two-dimensional ideal angle of incidence, and

BTF_x : blade thickness fraction which is given by,

$$BTF_x = \frac{\frac{[t_{max}]x}{D}-0.003}{1-x} + 0.003 \qquad (2.36)$$

in which, $(t_{max})_x$: maximum thickness of the blade at section x.

The calculation of the zero-lift angle, the ideal angle of incidence, and the lift-curve slope for various blade sections has been implemented in a GNU Octave function, which is available in Appendix C.

Substituting Equation 2.33 into Equation 2.30,

$$G(x,\theta) = \frac{\frac{dC_L}{d\alpha_2}(x,\theta)\frac{1}{K_C}[\gamma_0(x,\theta)-\beta_i(x,\theta)]c(x)\left[\frac{\overline{V_a}(x,\theta)}{V_a(\theta)}+\frac{U_a(x,\theta)}{V_a(\theta)}\right]}{2\pi D \sin\beta_i(x,\theta)}$$

(2.37)

Equation 2.37 can now be used to calculate a new value for the pitch angle and the iteration process is continued until convergence has been achieved. The iterative process is carried out at each radial position x and at each angular position θ of the propeller blade. At convergence, the resultant velocity is calculated as,

$$V(x,\theta) = \sqrt{[\overline{V_a}(x,\theta) + U_a(x,\theta)]^2 + [x\pi nD - \overline{V_t}(x,\theta) - U_t(x,\theta)]^2}$$

(2.38)

The thrust and torque at (x, θ) will then be,

$$\frac{dT}{dx}(x,\theta) = \frac{1}{2}\rho c(x)V^2(x,\theta)[C_L(x,\theta)\cos\beta_i(x,\theta) - C_D(x,\theta)\sin\beta_i(x,\theta)]$$

(2.39a)

$$\frac{dQ}{dx}(x,\theta) = \frac{1}{2}\rho x R c(x)V^2(x,\theta)[C_L(x,\theta)\sin\beta_i(x,\theta) + C_D(x,\theta)\cos\beta_i(x,\theta)]$$

(2.39b)

where C_D is the value of the drag coefficient, which is calculated using the CFD method described in Chapter IV. The code of the GNU Octave function that is used to pass the values of C_D to the lifting line script is available in Appendix G.

The thrust and torque acting on a blade at each angular position θ can be calculated as follows,

$$T(\theta) = \int_{x_H}^{1} \frac{1}{2}\rho c(x)V^2(x,\theta)[C_L(x,\theta)\cos\beta_i(x,\theta) - C_D(x,\theta)\sin\beta_i(x,\theta)]dx \quad (2.40a)$$

$$Q(\theta) = \int_{x_H}^{1} \frac{1}{2}\rho x R c(x)V^2(x,\theta)[C_L(x,\theta)\sin\beta_i(x,\theta) + C_D(x,\theta)\cos\beta_i(x,\theta)]dx \quad (2.40b)$$

The total thrust and torque on the propeller are then calculated as,

$$T = \frac{Z}{2\pi}\int_0^{2\pi} T(\theta)d\theta \quad\quad (2.41a)$$

$$Q = \frac{Z}{2\pi}\int_0^{2\pi} Q(\theta)d\theta \quad\quad (2.41b)$$

where Z is the number of propeller blades.

CHAPTER 3

LIFTING SURFACE CORRECTION FACTORS

3.1 Introduction

The lifting line theory described in Chapter 2 can address in a satisfactory manner the three-dimensional character of the flow around propeller blades only along the spanwise direction. Unlike airplane propellers, ship propellers have broad blades with low aspect ratios, which are often skewed in order to reduce propeller-induced unsteady forces and moments, and also have fairly large maximum thickness-to-chord ratios in order to address strength requirements.

All these geometric characteristics of ship propeller blades have a significant impact on the flow in the chordwise direction. The utilization of correction factors derived from systematic lifting surface calculations performed for a number of propellers by Morgan et al. [20] and Cumming et al. [21] can lead to significant improvements in a lifting-line-based propeller analysis method as has been demonstrated in [19, 22]. Three lifting surface correction factors are typically utilized:

- The camber correction factor K_c accounts for the loss in lift due to the curvature of the induced velocity along the chord.

- The ideal angle of incidence correction factor K_α accounts for the change in angle of incidence (angle that corresponds to shock-free entry) from its two-dimensional value.

- The correction factor for thickness K_t accounts for the main effect of blade thickness.

The lifting surface correction factors for camber and ideal angle of attack are defined as the ratio of three-dimensional maximum camber and ideal angle to the two-dimensional section values. Morgan et al. [20] derived the following expressions for these factors:

$$K_c(r) = \frac{f_p(x,x_{max})}{f_{max}} = \frac{1}{C_{max}C_L} \times \left\{ \int_0^{x_{max}} \left[\frac{U_n(x,x_c)}{V} - \frac{U(r)}{V} \right] dx_c - x_{max}\alpha_i(x) \right\} \tag{3.1}$$

$$K_\alpha(x) = \frac{\alpha_i}{\alpha_{i1.0}C_L} = \frac{1}{\alpha_{i1.0}C_L} \times \int_0^1 \left[\frac{U_n}{V_r}(x,x_c) - \frac{U(r)}{V_r} \right] dx_c \tag{3.2}$$

where,

C_{max} is the two-dimensional maximum mean line ordinate,

x_c is the chordwise section abscissa non-dimensionalized by the chord length c,

x_{max} is the chordwise position of the maximum camber, and

U_n is the resultant induced velocity normal to the chord.

The thickness correction factor is the ratio of the change in the ideal angle due to thickness (α_t) and the blade thickness fraction (BTF),

$$K_t(r) = \frac{\alpha_t(x)}{BTF} = \frac{1}{BTF} \int_0^1 \frac{U_{nt}}{V_r}(x,x_c) dx_c \tag{3.3}$$

where U_{nt} is the velocity induced by thickness normal to the blade section chord line.

The correction factors are independent of loading but depend on propeller parameters like the number of blades, the radial position, the hydrodynamic pitch angle, the expanded blade area ratio, and the skew angle. The definition of the propeller skew angle is shown in Figure 3.1.
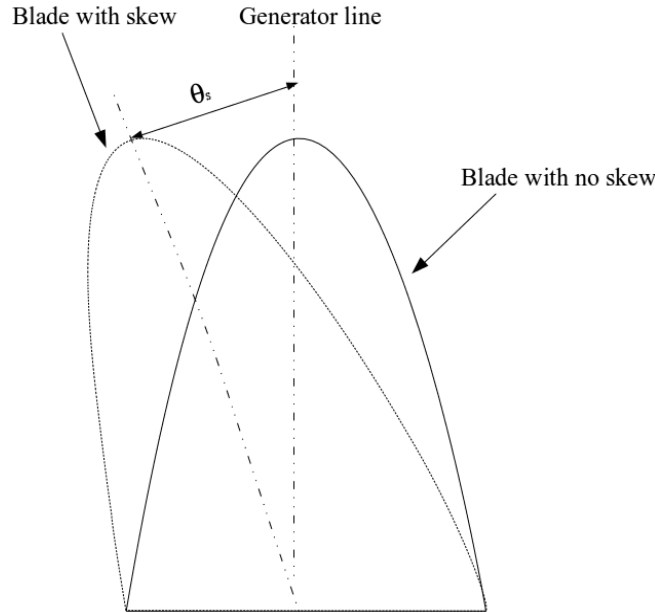
Figure 3.1 Definition of Skew Angle.

Morgan et al. [20] evaluated these integrals for specific values of the parameters: four, five and six

blades, radial positions x from 0.3 to 0.9, expanded blade area ratios from 0.35 to 1.15,

hydrodynamic pitch ratios from 0.4 to 2.0, and tip skew angles from 0 to 21 degrees. The tip skew

angle θ_s is defined as the angle between two lines in the projected plane; one straight line from the

shaft centerline through the mid-chord of the root section, which coincides with the blade generator

line, and the other from the shaft centerline through the mid-chord of the blade tip section as shown

in Figure 3.1. The two lines coincide, and thus $\theta_s = 0$, for a blade with no skew (also shown in

Figure 3.1). The radial distribution of skew used in [20] is given by,

$$\frac{skew}{R} = R_s - \sqrt{R_s^2 - (r - 0.2)^2} \tag{3.4}$$

$$R_s = \frac{0.32}{skew_{tip}} + \frac{skew_{tip}}{2}$$

where

$$skew_{tip} = \frac{\theta_s}{\cos \beta_i}$$

The thickness distribution used for the sections is that of a NACA 66 (mod) hydrofoil section [1] with a chordwise load distribution corresponding to a NACA a = 0.8 camber line. The radial thickness distribution is given by,

$$\frac{t_{max}(x)}{D} = (BTF - 0.003)(1 - x) + 0.003 \tag{3.5}$$

and the radial chord length distribution is given by,

$$\frac{c(x)}{D} = \frac{C(x)}{Z}\left(\frac{A_e}{A_0}\right) \tag{3.6}$$

where the values of the coefficient $C(x)$ [20] are listed in Table 3.1.

x	$C(x)$
0.2	1.6338
0.3	1.8082
0.4	1.9648
0.5	2.0967
0.6	2.1926
0.7	2.232
0.8	2.1719
0.9	1.8931

Table 3.1 Values of the Coefficient $C(x)$ at Different Radial Stations.

Cumming et al. [21] expanded the range of skew to include highly-skewed propellers. They represented the skew as a percentage of the tip skew angle θ_s,

$$\theta_{s,percent} = \frac{\theta_s}{360/Z} \times 100 \tag{3.7}$$

where the distribution of the skew angle in the radial direction is given by,

$$\theta_s(r) = \theta_s \left(\frac{r - 0.2R}{0.8R} \right) \tag{3.8}$$

The tip skew values that they tested were 0%, 50%, 100%, and 150% with an expanded area ratio of 0.75, and hydrodynamic pitch ratios of 0.8 and 1.2 for propellers with four, five, and six blades.

Van Oossanen used multiple regression analysis to derive polynomials to represent the lifting surface correction factors [19]. In this work, artificial neural networks (ANNs) and support vector regression (SVR) are utilized to derive surrogate models for the approximation of the lifting surface correction factors using the data sets from [20, 21]. For this purpose, the University of Waikato's open source data mining software WEKA [41] was utilized. For the case of a propeller with zero skew, the input variables correspond to the number of propeller blades Z, the expanded area ratio of the propeller blades A_e/A_0, the radial coordinate x, and the induced advance coefficient λ multiplied by π. The induced advance coefficient at each radial station r is defined as,

$$\lambda_i = r \cdot \tan(\beta_i) \tag{3.9}$$

The skew angle is also utilized as an input variable for propellers with moderate or high skew. Considering that the available data from [21] are given for a single value of the expanded area ratio A_e/A_0, the latter is not utilized as an input variable for propellers with high skew. It is demonstrated later in this chapter that the derived surrogate models have superior performance compared to the polynomial representation and could be utilized not only for interpolation but also for extrapolation purposes.

3.2 Deriving Surrogate Models for the Lifting Surface Correction Factors

Artificial neural networks have been utilized in various fields of scientific research as they represent a robust technique of statistical learning. A typical feedforward fully-connected ANN with two hidden layers is depicted in Figure 3.2. The input nodes provide a weighed input to the hidden layer nodes. That input is filtered using a sigmoid function and multiplied by a scaling coefficient (weight); the algebraic sum of the scaled output of the hidden nodes is then passed to the output node; that sum corresponds to the network prediction of the approximated function value.

The goal is to determine the network parameters, i.e. number of hidden layers and number of nodes in each hidden layer that provide the optimal network architecture. A detailed description of the theoretical aspect of ANNs, along with application examples, is provided in [42]. As was shown by Cybenko [43], an artificial neural network consisting of a single hidden layer of sigmoid neurons (hidden nodes) and linear output nodes is a universal approximator given an adequate number of hidden nodes.

The training of the ANN is performed using the backpropagation algorithm, which was originally proposed by Rumelhart et al. [44]. The network is trained by propagating forward the training dataset of points, calculating the prediction error, and then backpropagating the adjustments that need to be made to the network weights. This process is repeated a number of times (epochs) until there is no further improvement to the cumulative network prediction error evaluated on a validation data set.
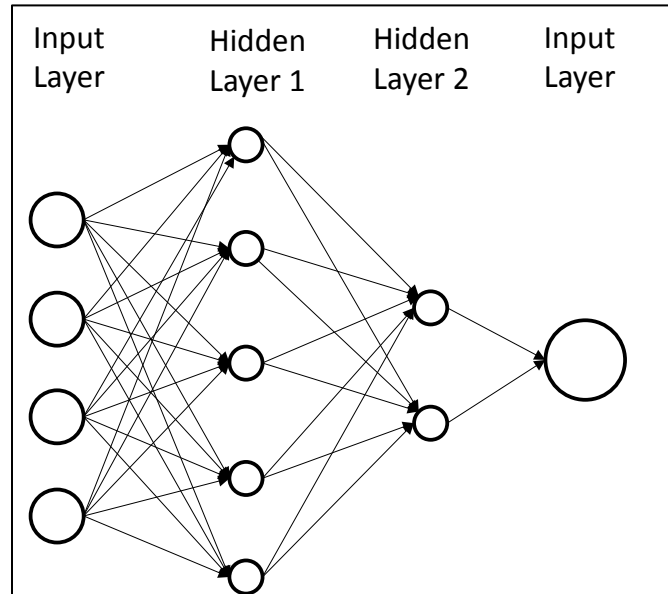
Figure 3.2 Feedforward Artificial Neural Network with Two Hidden Layers.

Support vector regression (SVR) is a machine learning technique based on the support vector machine (SVM) algorithm [45, 46]. Given a set of training points (features) SVR uses a nonlinear mapping function to map the data from the input variable space to the feature space. The algorithm attempts to find a regression function that predicts the training points within a specified distance from the actual values and also the function is as flat as possible. The trade-off between accuracy and flatness is controlled through a parameter C. The SVR training algorithm used in WEKA was proposed by Shevade et al. [47]. Two nonlinear mapping functions (kernels) were considered:

- The radial basis function (RBF) kernel:

$$k(\mathbf{x}, \mathbf{x}_i) = exp(\gamma \|\mathbf{x} - \mathbf{x}_i\|^2) \tag{3.10}$$

- and the Pearson VII function-based universal kernel (PUK) [48]:

$$k(\mathbf{x}, \mathbf{x}_i) = \frac{1}{\left[1 + \left(\frac{2\sqrt{\|\mathbf{x}-\mathbf{x}_i\|^2 \sqrt{2^{1/\omega}-1}}}{\sigma}\right)^2\right]^\omega} \tag{3.11}$$

where, γ, σ, and ω are input parameters that control the shape of the function, $\mathbf{x}$ is the input variable vector, and $\mathbf{x}_i$ is a training point.

It was decided to generate a surrogate model for each of the aforementioned lifting surface correction factors for propellers with zero skew, moderate skew, and high skew. ANNs with one, two, and three hidden layers and with varying number of nodes in each layer were systematically tested. The parameters varied for SVR were the complexity constant, C, and the kernel parameters γ, σ, and ω. The result for the best performer for each ANN architecture and SVR kernel are reported.

The learning effectiveness of each surrogate model was assessed by calculating the following statistical performance indicators: the root mean squared error (RMSE), the mean absolute error (MAE), and the correlation coefficient (R). The smaller the values of the RMSE and the MAE, the higher the accuracy of the surrogate model. The correlation coefficient can vary between zero and one. High R values, i.e. above 0.95, indicate that the function approximator is capable of explaining the variation in the training data set.

Five-fold cross validation was utilized for the training/validation of each surrogate model: the training data are split into five sets and the model is evaluated five times using each set in turn for validation and the other four for training; the performance indicators were calculated as the average

of all five runs. In this way, the performance of the network on unseen data could be evaluated over the entire range of the available data.

Even though cross validation is a good indicator of the surrogate model's performance on unseen data, i.e. extrapolation capability, considering the fact that the available data sets are relatively small, and particularly the ones for highly-skewed propellers, it was deemed appropriate to use a second training/validation method. The 67-33% split method was utilized where 67% of the dataset is used for training and 33% is used for validation of the surrogate model.

Tables 3.2 through 3.10 list the calculated statistical performance indicator values. The architecture of the network and the number of epochs used for its training are provided for the ANNs. The values of the C, γ, σ, and ω parameters are listed for the SVR surrogate models. The best values obtained for each performance indicator and for each of the training/validation methods are shown in bold font. The surrogate model selected to be utilized as the function approximator for each skew condition and lifting surface correction factor is also shown in bold font. Those surrogate models were coded in GNU Octave and the corresponding functions for K_c, K_a, and K_t are provided in Appendices D, E, and F, respectively.

	Five-fold cross validation		
	RMSE	MAE	*R*
ANN-11 (N = 10000)	0.0427	0.0299	0.9958
ANN-11-4 (N = 12000)	**0.0238**	**0.0164**	**0.9987**
ANN-11-3-2 (N = 20000)	0.0251	0.0176	0.9986
SMO-PUK (C = 70.0, ω = 2.0, σ = 2.0)	0.0293	0.0195	0.9980
SMO-RBF (C = 55.0, γ = 1.8)	0.0280	0.0193	0.9982
	67-33% split		
	RMSE	MAE	*R*
ANN-11 (N=10000)	0.0506	0.0380	0.9939
ANN-11-4 (N=12000)	0.0231	**0.0131**	**0.9985**
ANN-11-3-2 (N=20000)	**0.0230**	0.0135	**0.9985**
SMO-PUK (C=70.0, ω = 2.0, σ = 2.0)	0.0270	0.0191	0.9979
SMO-RBF (C = 55.0, γ = 1.8)	0.0272	0.0202	0.9979

Table 3.2. Comparison of K_c Surrogate Models for Zero Skew.

	Five-fold cross validation		
	RMSE	MAE	*R*
ANN-12 (N = 10000)	0.0273	0.0203	0.9985
ANN-10-3 (N = 15000)	**0.0257**	**0.0195**	**0.9987**
ANN-11-3-2 (N = 15000)	0.0291	0.0219	0.9984
SMO-PUK (C = 100.0, ω = 29.0, σ = 1.6)	0.0301	0.0205	0.9982
SMO-RBF (C = 70.0, γ = 1.3)	0.0313	0.0212	0.9981
	67-33% split		
	RMSE	MAE	*R*
ANN-12 (N = 10000)	0.0268	0.0199	0.9985
ANN-10-3 (N = 15000)	**0.0233**	**0.0178**	**0.9990**
ANN-11-3-2 (N = 15000)	0.0246	0.0183	0.9988
SMO-PUK (C = 100.0, ω = 29.0, σ = 1.6)	0.0340	0.0221	0.9975
SMO-RBF (C = 70.0, γ = 1.3)	0.0354	0.0230	0.9973

Table 3.3. Comparison of K_c Surrogate Models for Moderate Skew.

	Five-fold cross validation		
	RMSE	MAE	*R*
ANN-14 (N = 40000)	0.0373	0.0264	0.9955
ANN-10-3 (N = 40000)	0.0343	0.0223	0.9962
ANN-10-4-2 (N = 35000)	**0.0404**	**0.0246**	**0.9946**
SMO-PUK (C = 300.0, ω = 50.0, σ = 1.9)	0.0581	0.0372	0.9913
SMO-RBF (C = 450.0, γ = 0.75)	0.0552	0.0358	0.9922
	67-33% split		
	RMSE	MAE	*R*
ANN-14 (N = 40000)	0.0378	0.0262	0.9953
ANN-10-3 (N = 40000)	0.0391	0.0250	0.9952
ANN-10-4-2 (N = 35000)	**0.0333**	**0.0240**	**0.9963**
SMO-PUK (C = 300.0, ω = 50.0, σ = 1.9)	0.0617	0.0394	0.9887
SMO-RBF (C = 450.0, γ = 0.75)	0.0634	0.0399	0.9884

Table 3.4. Comparison of K_c Surrogate Models for High Skew.

	Five-fold cross validation		
	RMSE	MAE	*R*
ANN-12 (N = 10000)	0.0404	0.0291	0.9966
ANN-11-3 (N = 10000)	0.0284	0.0209	0.9983
ANN-11-3-2 (N = 15000)	**0.0245**	**0.0183**	**0.9988**
SMO-PUK (C = 30.0, ω = 2.0, σ = 1.4)	0.0392	0.0241	0.9966
SMO-RBF (C = 50.0, γ = 2.5)	0.0386	0.0241	0.9967
	67-33% split		
	RMSE	MAE	*R*
ANN-12 (N = 10000)	0.0287	0.0224	0.9978
ANN-11-3 (N = 10000)	0.0233	0.0188	0.9986
ANN-11-3-2 (N = 15000)	**0.0222**	**0.0166**	**0.9989**
SMO-PUK (C = 30.0, ω = 2.0, σ = 1.4)	0.035	0.0212	0.9967
SMO-RBF (C = 50.0, γ = 2.5)	0.0342	0.0214	0.9969

Table 3.5. Comparison of K_α Surrogate Models for Zero Skew.

	Five-fold cross validation		
	RMSE	MAE	*R*
ANN-12 (N = 10000)	0.1102	0.0834	0.9934
ANN-12-3 (N = 15000)	0.1062	0.0804	0.9939
ANN-12-3-2 (N = 16000)	**0.0975**	**0.0705**	**0.9957**
SMO-PUK (C = 90.0, ω = 3.0, σ = 1.8)	0.1164	0.0823	0.9926
SMO-RBF (C = 80.0, γ = 1.2)	0.1179	0.0794	0.9924
	67-33% split		
	RMSE	MAE	*R*
ANN-12 (N = 10000)	0.1284	0.0949	0.9924
ANN-12-3 (N = 15000)	0.1081	0.0832	0.9958
ANN-12-3-2 (N = 16000)	**0.0889**	**0.0605**	**0.9962**
SMO-PUK (C = 90.0, ω = 3.0, σ = 1.8)	0.1338	0.0972	0.9915
SMO-RBF (C = 80.0, γ = 1.2)	0.1432	0.0993	0.9906

Table 3.6. Comparison of K_α Surrogate Models for Moderate Skew.

	Five-fold cross validation		
	RMSE	MAE	*R*
ANN-14 (N = 30000)	**0.2243**	**0.1725**	**0.9984**
ANN-11-4 (N = 30000)	0.2585	0.2016	0.9979
ANN-10-4-2 (N = 30000)	0.3434	0.2339	0.9962
SMO-PUK (C = 1200.0, ω = 35.0, σ = 2.5)	1.3386	0.9199	0.9448
SMO-RBF (C = 1200.0, γ = 1.55)	2.0430	1.2394	0.8563
	67-33% split		
	RMSE	MAE	*R*
ANN-14 (N = 30000)	**0.2407**	**0.1806**	**0.9984**
ANN-11-4 (N = 30000)	0.4313	0.3374	0.9965
ANN-10-4-2 (N = 30000)	0.4909	0.3907	0.9933
SMO-PUK (C = 1200.0, ω = 35.0, σ = 2.5)	1.5884	1.0343	0.9312
SMO-RBF (C = 1200.0, γ = 1.55)	1.9197	1.247	0.8958

Table 3.7. Comparison of K_α Surrogate Models for High Skew.

	Five-fold cross validation		
	RMSE	MAE	*R*
ANN-11 (N = 4000)	0.0451	0.0315	0.9848
ANN-11-4 (N = 8000)	0.0213	0.0114	0.9966
ANN-11-3-2 (N = 12000)	0.0209	0.0114	0.9967
SMO-PUK (C = 10.0, ω = 2.0, σ = 1.7)	**0.0192**	**0.0084**	**0.9973**
SMO-RBF (C = 35.0, γ = 1.1)	0.0199	**0.0084**	0.9971
	67-33% split		
	RMSE	MAE	*R*
ANN-11 (N = 4000)	0.0208	0.0146	0.9969
ANN-11-4 (N = 8000)	0.0216	0.0137	0.9964
ANN-11-3-2 (N = 12000)	0.0213	0.0130	0.997
SMO-PUK (C = 10.0, ω = 2.0, σ =1.7)	**0.0136**	**0.0070**	**0.9984**
SMO-RBF (C = 35.0, γ = 1.1)	0.0139	0.0071	**0.9984**

Table 3.8. Comparison of K_t Surrogate Models for Zero Skew.

	Five-fold cross validation		
	RMSE	MAE	*R*
ANN-15 (N = 6000)	0.0199	0.0140	0.9973
ANN-13-3 (N = 15000)	0.0174	0.0126	0.9980
ANN-13-3-1 (N = 18000)	0.0175	0.0124	0.9979
SMO-PUK (C = 7.0, ω = 8.0, σ = 1.2)	**0.0146**	**0.0088**	**0.9986**
SMO-RBF (C = 8.0, γ = 2.1)	0.0148	**0.0088**	0.9985
	67-33% split		
	RMSE	MAE	*R*
ANN-15 (N = 6000)	0.0172	0.0118	0.9982
ANN-13-3 (N = 15000)	0.0154	0.0104	**0.9985**
ANN-13-3-1 (N = 18000)	**0.0153**	0.0102	**0.9985**
SMO-PUK (C = 7.0, ω = 8.0, σ = 1.2)	0.0158	**0.0100**	**0.9985**
SMO-RBF (C = 8.0, γ = 2.1)	0.0161	0.0101	**0.9985**

Table 3.9. Comparison of K_t Surrogate Models for Moderate Skew.

	Five-fold cross validation		
	RMSE	MAE	*R*
ANN-13 (N = 35000)	0.0066	0.0049	0.9996
ANN-11-4 (N = 30000)	0.0069	0.0053	0.9995
ANN-11-3-2 (N = 25000)	**0.0067**	**0.0051**	**0.9996**
SMO-PUK (C = 100.0, ω = 10.0, σ = 1.8)	0.0128	0.0077	0.9984
SMO-RBF (C = 100.0, γ = 0.95)	0.0127	0.0076	0.9984
	67-33% split		
	RMSE	MAE	*R*
ANN-13 (N = 35000)	0.0134	0.0093	0.9992
ANN-11-4 (N = 30000)	0.0122	0.0082	0.9987
ANN-11-3-2 (N = 25000)	**0.0073**	**0.0057**	**0.9995**
SMO-PUK (C = 100.0, ω = 10.0, σ = 1.8)	0.0143	0.0097	0.9981
SMO-RBF (C = 100.0, γ = 0.95)	0.0153	0.0094	0.9979

Table 3.10. Comparison of K_t Surrogate Models for High Skew.

In all cases, the surrogate models were able to explain the variation in the dataset, which is demonstrated by the fact that the computed correlation coefficient values are greater than 0.99. The main selection criterion was the RMSE value in the five-fold cross validation results as long as the MAE value was also low. RMSE is an indicator of the overall approximation capability of the model, whereas a high MAE value indicates large prediction errors for certain points in the dataset.

Figures 3.3 through 3.11 present a comparison between the correction factor values computed using the derived surrogate models ('ANN'), the corresponding values computed using Van Oossanen's polynomials ('VAN'), and the values reported in [20] 'Morgan' and in [21] 'Cumming'.
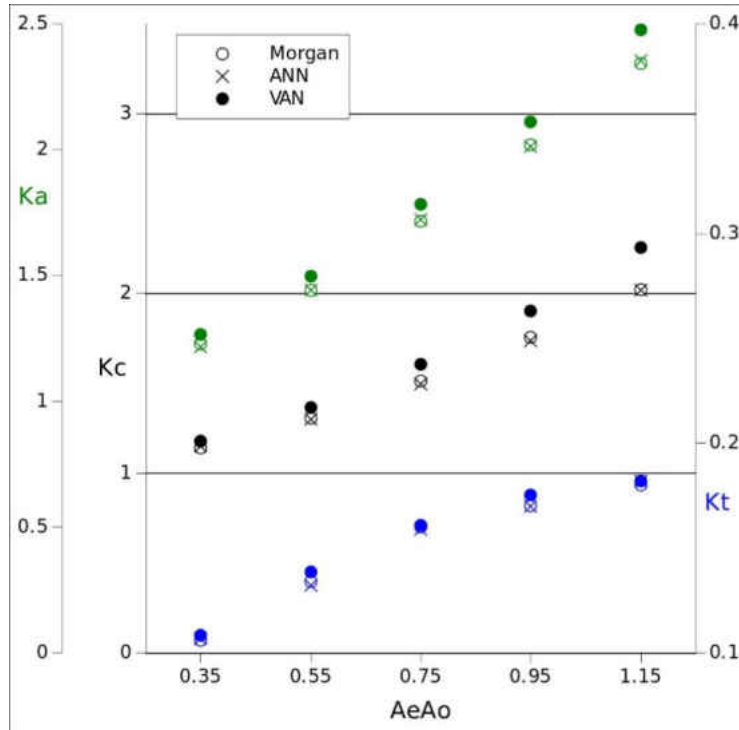
Figure 3.3 Four-Bladed Propeller with Zero Skew at $x = 0.6$ and $\pi \cdot \lambda = 2.0$
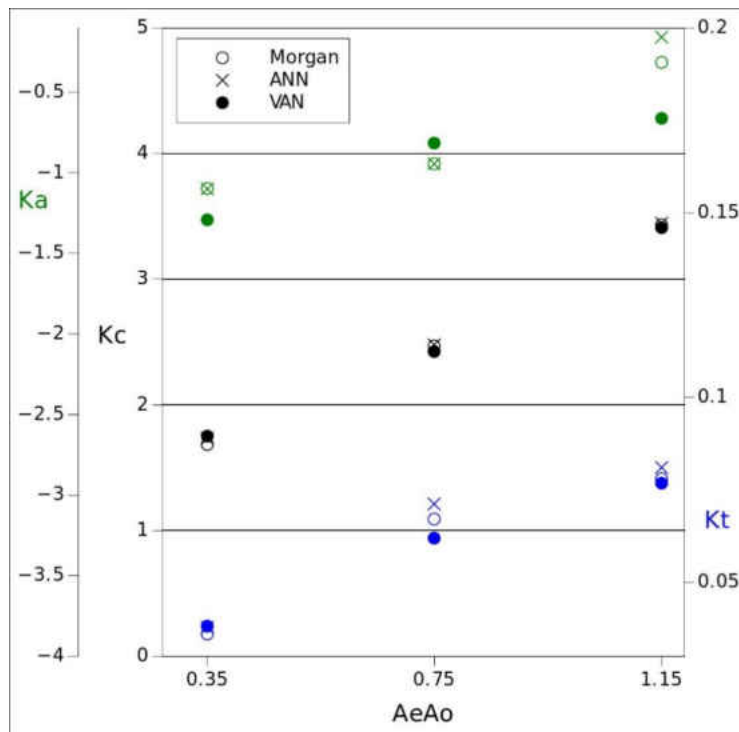
Figure 3.4 Four-Bladed Propeller with Moderate Skew at $x = 0.9$ and $\pi \cdot \lambda = 2.0$.

Figures 3.3 and 3.4 demonstrate the fact that the derived surrogate models are capable of approximating the K_c, K_a, and K_t correction factors over the entire range of propeller aspect-ratio values for four-bladed propellers. The ANN surrogate model for moderate skew used for K_c (see Table 3.3) shows excellent performance considering the wide range of K_c values; this is very important because the latter is used to scale the lift-curve slope (see Equation 2.31). Figure 3.5 reveals that the surrogate models for all three correction factors remain very effective even when utilized for radial stations with high skew values. Similar conclusions can be drawn from Figures 3.6 through 3.8 for five-bladed propellers and from Figures 3.9 through 3.11 for six-bladed propellers.
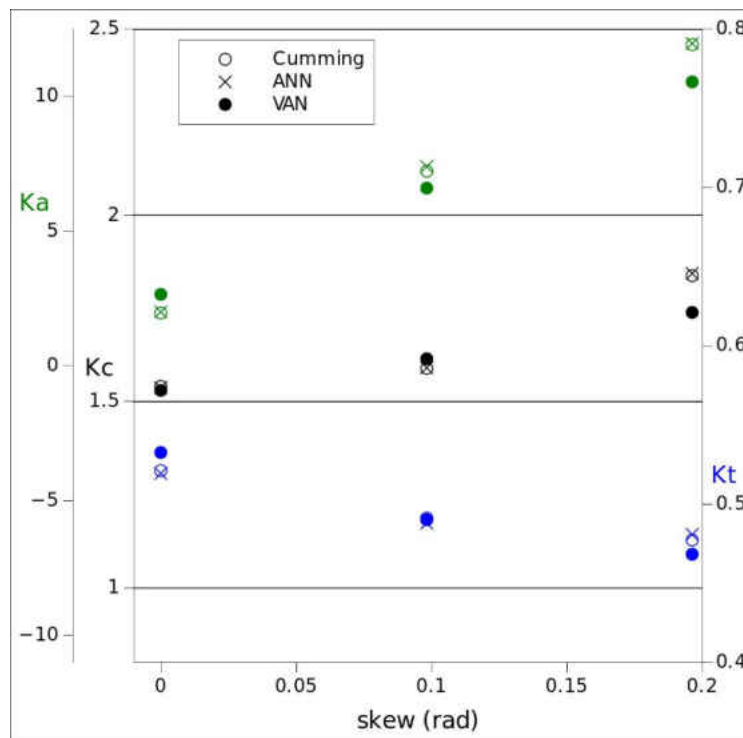
Figure 3.5 Four-Bladed Propeller with High Skew at $x = 0.3$ and $\pi \cdot \lambda = 0.8$.
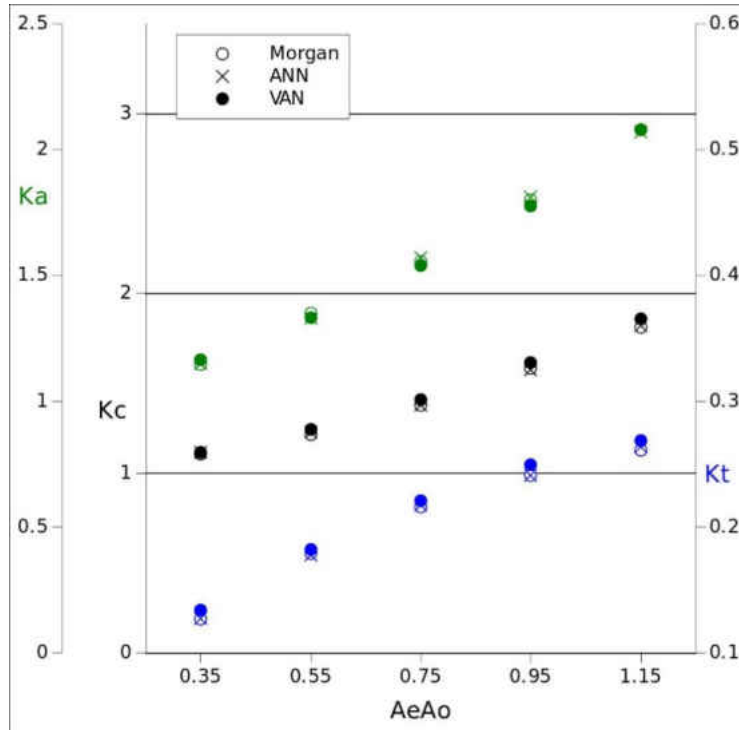
Figure 3.6 Five-Bladed Propeller with Zero Skew at $x = 0.6$ and $\pi \cdot \lambda = 2.0$.
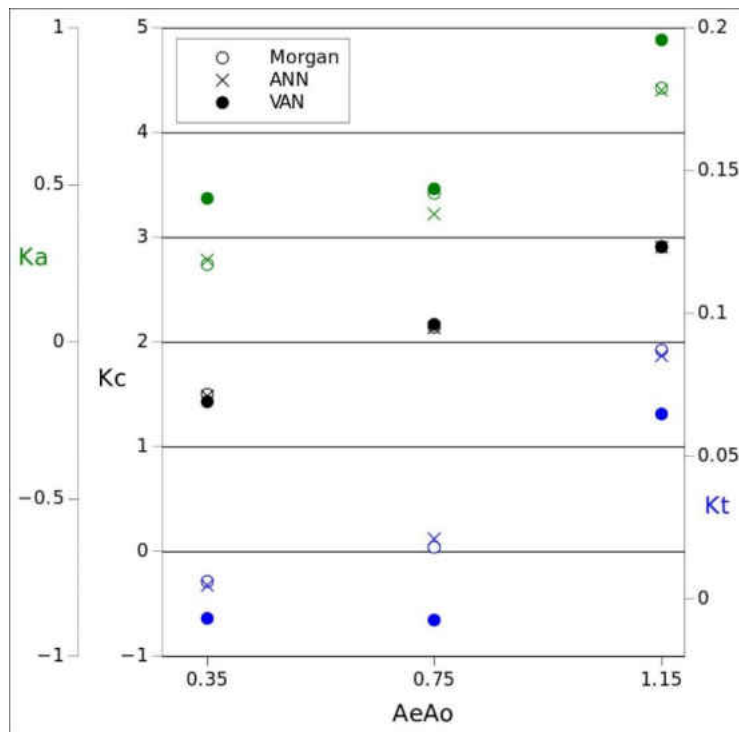
Figure 3.7 Five-Bladed Propeller with Moderate Skew at $x = 0.9$ and $\pi \cdot \lambda = 0.4$.
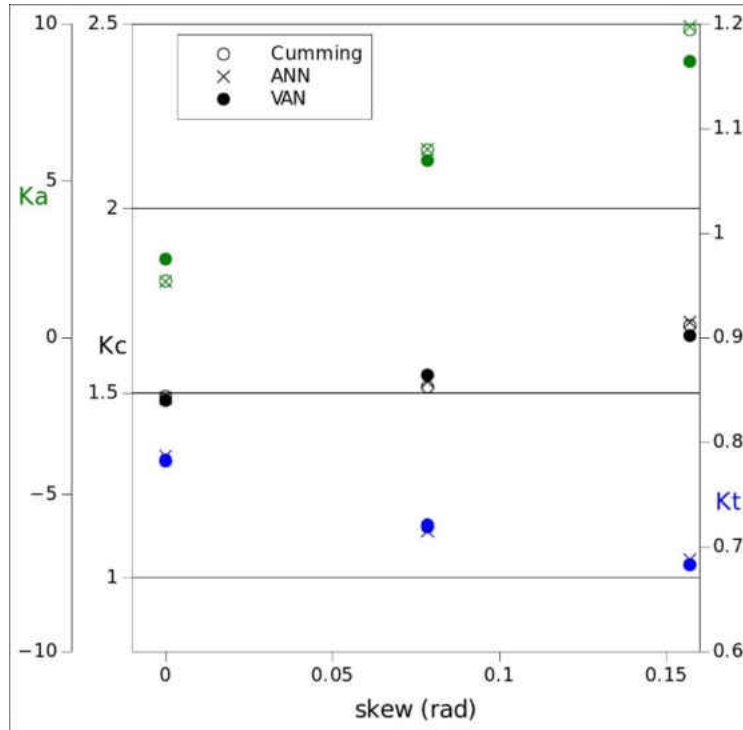
Figure 3.8 Five-Bladed Propeller with High Skew at $x = 0.3$ and $\pi \cdot \lambda = 0.8$.
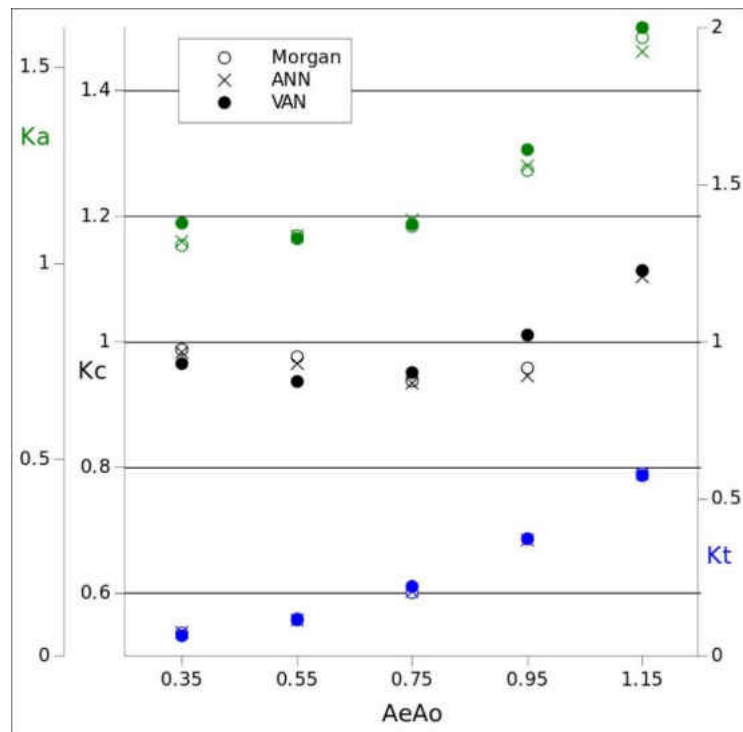
Figure 3.9 Six-Bladed Propeller with Zero Skew at $x = 0.6$ and $\pi \cdot \lambda = 0.4$.
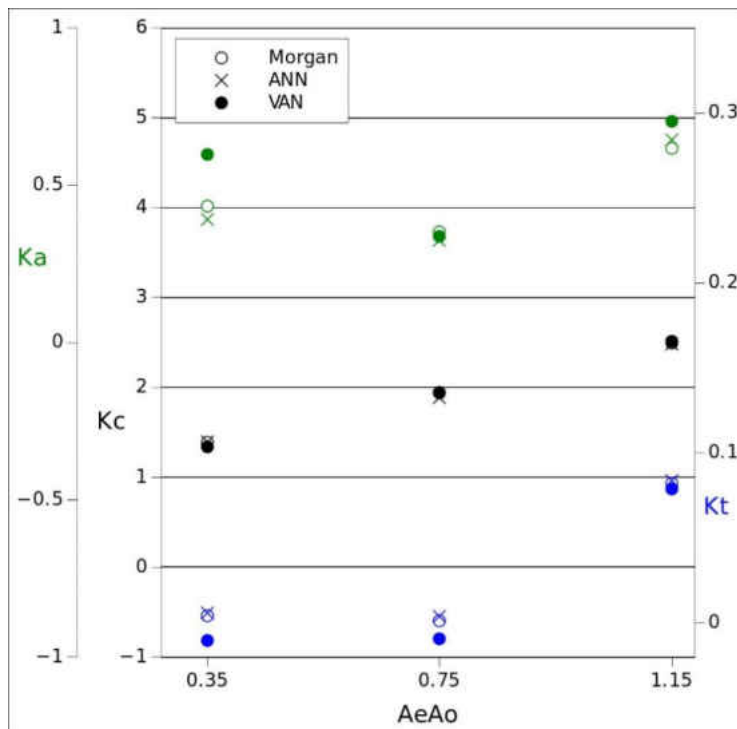
Figure 3.10 Six-Bladed Propeller with Moderate Skew, $x = 0.9$, and $\pi \cdot \lambda = 0.4$.
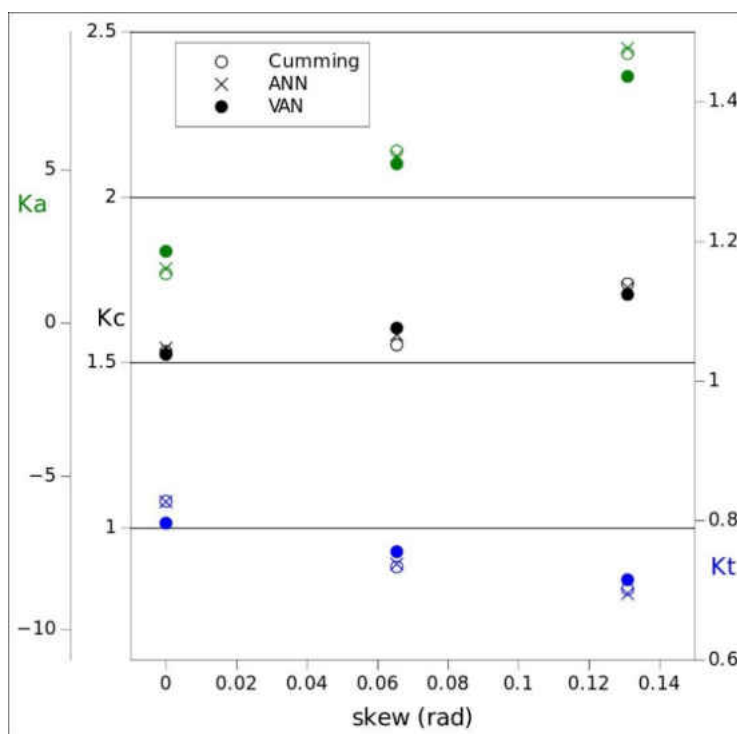
Figure 3.11 Six-Bladed Propeller with High Skew at $x = 0.3$ and $\pi \cdot \lambda = 1.2$.

As is amply demonstrated in Figure 3.7, the utilized SMO-PUK surrogate model (see Table 3.9) is capable of capturing the nonlinearities present in the K_t values when the latter are plotted as a function of the expanded area ratio, unlike the multiple-regression-based polynomials. Figure 3.10 shows that the ANN model (see Table 3.6) can provide very accurate values for K_a despite the nonlinearities present. The K_a and K_t correction factors have an impact on the calculation of the three-dimensional values of the zero-lift angle and the ideal angle of incidence through their implementation in Equation 2.35.

CHAPTER 4

PERFORMANCE EVALUATION OF BLADE SECTIONS USING CFD

4.1 Introduction

The original lifting line model assumes inviscid flow around the propeller. Van Oossanen in [19] showed the significance of incorporating the effect of viscosity into the propeller analysis and utilized boundary layer flow solvers and empirical correlations with experimental data. Viscous effects on hydrodynamic loading and cavitation have also been investigated by Shen and Dimotakis [49]. In this work, the impact of viscosity is incorporated into the propeller analysis and design method by utilizing CFD to compute the lift, drag, and pressure distribution of the two-dimensional blade sections. The computed values are then utilized in the thrust and torque calculation equations listed in Chapter 2, in addition to determining the cavitation-free range of the blade section based on the negative minimum pressure coefficient value.

The RANS equations and the turbulence model utilized are described in Section 4.2. The flow domain discretization is discussed in Section 4.3; the boundary conditions that are applied to the flow domain are discussed in Section 4.4, in addition to a brief description of the RANSE solver used in OpenFOAM. The results of a grid verification study are reported in Section 4.5, and experimental validation results are provided in Section 4.6, followed by the presentation, in Section 4.7, of the hydrodynamic characteristics of the baseline blade sections utilized in the shape optimization application described in Chapter 5.

4.2 RANS Equations and Turbulence Modeling

One of the most popular techniques employed in fluid flow computations involves the solution of the Reynolds-averaged Navier-Stokes (RANS) equations. The basic premise is that the fluid velocity and pressure can be expressed as the sum of mean and fluctuating parts as,

$$u_i = U_i + u'_i, \qquad p = P + p' \tag{4.1}$$

where the time-averaged mean and fluctuating parts satisfy

$$\bar{u}_i = U_i, \quad \overline{u'_i} = 0 \quad \text{and} \quad \bar{p} = P, \quad \overline{p'} = 0 \tag{4.2}$$

Inserting Equation 4.1 into the Navier-Stokes equations for steady fluid flow and taking a time average yields,

$$\frac{\partial U_i}{\partial x_i} = 0 \tag{4.3}$$

$$\rho \frac{\partial}{\partial x_j}\left(U_i U_j\right) = -\frac{\partial P}{\partial x_i} + \frac{\partial}{\partial x_j}\left(2\mu S_{ij} - \rho \overline{u'_i u'_j}\right) \tag{4.4}$$

where S_{ij} is the mean strain-rate tensor

$$S_{ij} = \frac{1}{2}\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) \tag{4.5}$$

and $\overline{u'_i u'_j} = -\tau_{ij}$ is the Reynolds stress tensor.

Combining Equations 4.3 and 4.4 we get,

$$U_j \frac{\partial U_i}{\partial x_j} = -\frac{\partial P}{\partial x_i} + \frac{\partial^2 U_i}{\partial x_i \partial x_j} - \frac{\partial \overline{u'_i u'_j}}{\partial x_j} \tag{4.6}$$

The Reynolds stresses are modeled using the Boussinesq hypothesis,

$$-\overline{u'_i u'_j} = 2\nu_T S_{ij} - \frac{2}{3}\kappa\delta_{ij} \tag{4.7}$$

where ν_T is the kinetic eddy viscosity and κ is the turbulence kinetic energy defined as,

$$\kappa = \frac{1}{2}\overline{u'_\iota u'_\iota} \tag{4.8}$$

A transport equation is typically employed to compute the turbulence kinetic energy. One of the most popular turbulence models is the κ-ω SST turbulence model. A detailed description of the variants of the κ-ω SST model is presented in [50]. The standard model that is available in OpenFOAM involves solving two transport equations; one for the turbulence kinetic energy κ and a second one for the specific turbulence dissipation rate ω as follows:

$$\rho\frac{\partial(U_j\kappa)}{\partial x_j} = \tilde{P}_\kappa - \beta^*\rho\kappa\omega + \frac{\partial}{\partial x_j}\left[(\mu + \sigma_\kappa\mu_t)\frac{\partial\kappa}{\partial x_j}\right] \tag{4.9}$$

$$\rho\frac{\partial(U_j\omega)}{\partial x_j} = \alpha\rho S^2 - \beta\rho\omega^2 + \frac{\partial}{\partial x_j}\left[(\mu + \sigma_\omega\mu_t)\frac{\partial\omega}{\partial x_j}\right] + 2(1 - F_1)\rho\sigma_{\omega 2}\frac{1}{\omega}\frac{\partial\kappa}{\partial x_j}\frac{\partial\omega}{\partial x_j}$$

$$\tag{4.10}$$

The blending function F_1 determines what model is used as it is equal to zero away from the surface (where the κ-ε turbulence model is utilized) and is equal to one inside the boundary layer (where the κ-ω model is employed). It is defined as:

$$F_1 = \tanh\left\{\left\{min\left[max\left(\frac{\sqrt{\kappa}}{\beta^*\omega y}, \frac{500\nu}{y^2\omega}\right), \frac{4\rho\sigma_{\omega 2}\kappa}{CD_{\kappa\omega}y^2}\right]\right\}^4\right\} \tag{4.11}$$

where,

$CD_{\kappa\omega} = max\left(2\rho\sigma_{\omega 2}\frac{1}{\omega}\frac{\partial\kappa}{\partial x_j}\frac{\partial\omega}{\partial x_j}, 10^{-10}\right)$ and y is the distance from the nearest wall.

The turbulent eddy viscosity is defined as,

$$\nu_t = \frac{\alpha_1}{max(\alpha_1\omega, \ |S|F_2)} \tag{4.12}$$

where $|S|$ is the invariant measure of the strain rate and F_2 is another blending function defined as,

$$F_2 = \tanh\left\{\left[max\left(\frac{\sqrt{\kappa}}{\beta^*\omega y}, \frac{500\nu}{y^2\omega}\right)\right]^2\right\}$$ (4.13)

For the production term in the κ transport equation (Equation 4.9), a limiter is used so as to avoid the build-up of turbulence in the stagnation regions,

$$P_\kappa = \mu_t \frac{\partial U_i}{\partial x_j}\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) \longrightarrow \tilde{P}_\kappa = min(P_\kappa, 10\beta^*\rho\kappa\omega)$$ (4.14)

The constants used in the equations are calculated via blending of the corresponding constants of the two turbulence models, for example,

$$\alpha = \alpha_1 F + \alpha_2(1 - F)$$ (4.15)

The values of these constants are [50],

$\beta^* = 0.09$, $\alpha_1 = 5/9$, $\beta_1 = 3/40$, $\alpha_{\kappa 1} = 0.85$, $\alpha_{\omega 1} = 0.5$, $\alpha_2 = 0.44$, $\beta_2 = 0.0828$, $\alpha_{\kappa 2} = 1$, and $\alpha_{\omega 2} = 0.856$.

In was decided to utilize the κ-ω SST turbulence model developed by Langtry and Menter [51, 52], which has the capability to predict boundary layer flow transition from laminar to turbulent using two additional transport equations and empirical correlations. The version of the model utilized in this work corresponds to the one modified by Langtry and Menter in order to improve the prediction of natural transition [53]. As will be shown in later sections the utilized turbulence model increases the prediction accuracy of integral variables, like the lift and drag coefficients, but also of flow field properties, like the pressure distribution on the blade section, compared to the standard κ-ω SST turbulence model.

In order to predict the flow transition, two additional transport equations are solved: one for the intermittency γ, and a second one for the transition momentum-thickness Reynolds number $\widetilde{Re}_{\theta t}$. According to the formulation presented in [53], the γ transport equation is given as,

$$\rho \frac{\partial(U_j \gamma)}{\partial x_j} = P_\gamma - E_\gamma + \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_t}{\sigma_f}\right)\frac{\partial \gamma}{\partial x_j}\right] \tag{4.16}$$

where the source term is defined as

$$P_\gamma = F_{length} c_{a1} \rho S [\gamma F_{onset}]^{0.5}(1 - c_{e1}\gamma) \tag{4.17}$$

F_{length} is an empirical correlation that controls the length of the transition region and F_{onset} controls the transition onset location. The destruction term is defined as

$$E_\gamma = c_{a2}\rho \Omega \gamma F_{turb}(c_{e2}\gamma - 1) \tag{4.18}$$

where Ω is the vorticity magnitude. The transition onset is controlled by the correlation between the vorticity Reynolds number (Re_v) and the momentum thickness Reynolds number (Re_θ) as follows:

$$Re_v = \frac{\rho y^2 S}{\mu} \tag{4.19}$$

$$F_{onset1} = \frac{Re_v}{2.193 Re_{\theta c}} \tag{4.20}$$

$$F_{onset2} = min(max(F_{onset1}, F_{onset1}^4), 2.0) \tag{4.21}$$

$$Re_T = \frac{\rho \kappa}{\mu \omega} \tag{4.22}$$

$$F_{onset3} = max\left(1 - \left(\frac{R_T}{2.5}\right)^3, 0\right) \tag{4.23}$$

$$F_{onset} = max(F_{onset2} - F_{onset3}, 0) \tag{4.24}$$

where $Re_{\theta C}$ is the critical Reynolds number where the intermittency first starts to increase in the boundary layer.

The transport equation for $\widetilde{Re}_{\theta t}$ is as follows:

$$\rho \frac{\partial(U_j \widetilde{Re}_{\theta t})}{\partial x_j} = P_{\theta t} + \frac{\partial}{\partial x_j}\left[\sigma_{\theta t}(\mu + \mu_t)\frac{\partial \widetilde{Re}_{\theta t}}{\partial x_j}\right] \tag{4.25}$$

where,

$$P_{\theta t} = c_{\theta t}\frac{\rho}{t}\left(Re_{\theta t} - \widetilde{Re}_{\theta t}\right)(1.0 - F_{\theta t}) \tag{4.26}$$

$$Re_{\theta t} = \frac{\rho \theta_t U_0}{\mu} \tag{4.27}$$

$$t = \frac{500\mu}{\rho U^2} \tag{4.28}$$

t is a time scale which is present for dimensional reasons and $Re_{\theta t}$ is the transition Reynolds number based on the freestream conditions. Also,

$$F_{\theta t} = min\left(max\left(F_{wake}e^{-\left(\frac{y}{\delta}\right)^4}, 1.0 - \frac{\gamma - 1/c_{e2}}{1 - 1/c_{e2}}\right), 1\right) \tag{4.29}$$

$$\theta_{BL} = \frac{\widetilde{Re}_{\theta t}\mu}{\rho U}; \quad \delta_{BL} = \frac{15}{2}\theta_{BL}; \quad \delta = \frac{50\Omega y}{U}\delta_{BL} \tag{4.30}$$

$$Re_\omega = \frac{\rho\omega y^2}{\mu}; \quad F_{wake} = e^{-\left(\frac{Re_\omega}{1E+5}\right)^2} \tag{4.31}$$

The values of the various constants in the model are [53],

$$c_{e1} = 1.0; \quad c_{a1} = 2.0; \quad c_{e2} = 50; \quad c_{a2} = 0.06; \quad \sigma_f = 1.0; \quad c_{\theta t} = 0.03; \quad \sigma_{\theta t} = 2.0$$

The two additional transport equations are integrated into the standard κ-ω SST formulation by modifying the source and decay terms in the κ transport equation as,

$$\tilde{P}_k = \gamma_{eff}P_k; \quad \tilde{D}_k = min(max(\gamma_{eff}, 0.1), 1)D_k \tag{4.32}$$

and the blending function in the ω transport equation as,

$$R_y = \frac{\rho y\sqrt{\kappa}}{\mu}; \quad F_3 = e^{e^{-\left(\frac{R_y}{120}\right)^8}}; \quad F_1 = max(F_{1orig}, F_3) \tag{4.33}$$

where P_k, D_k, and F_{1orig} are the original source term, the decay term, and the blending function, respectively. The transition κ-ω SST model is available in the OpenFOAM-dev, the current development line of OpenFOAM.

4.3 Discretization of the Flow Domain for 2-D Blade Sections

The domain constructed for the evaluation of the blade sections is illustrated in Figure 4.1. It consists of a semicircular arc of radius $15 \cdot c$ (where c is the section chord length) at the inlet and a rectangular block extending up to $50 \cdot c$ in the downstream direction. In order to to allow for better control of the grid spacing in the boundary layer region, the domain was subdivided into a smaller semicircular arc of radius $2 \cdot c$ around the blade section. The portion of this subdivision extending in the downstream direction aligns itself with the wake based on the angle-of-attack value as illustrated in Figures 4.2 and 4.3. The foil is divided into two sections near the mid-chord so as to have a denser distribution of cells in the vicinity of the leading and trailing edges. At least ten cells in the direction normal to the blade section surface are placed within the boundary layer in order to provide adequate flow resolution in that region, as shown in Figure 4.4. The number of cells in each segment and the corresponding expansion ratio are listed in Table 4.1. The total number of cells is 120,000; that number was decided upon after performing a grid verification analysis, which is discussed in Section 4.5. The GNU Octave script used to generate the OpenFOAM grid is available in Appendix H.

Segments	Cells	Expansion Ratio
Foil (Front)	150	5
Foil (Back)	100	0.2
Downstream	250	600
Transverse (Inner)	85	30000
Transverse (Outer)	25	3

Table 4.1: Number of Cells and Expansion Ratio in Each Segment of the Grid.
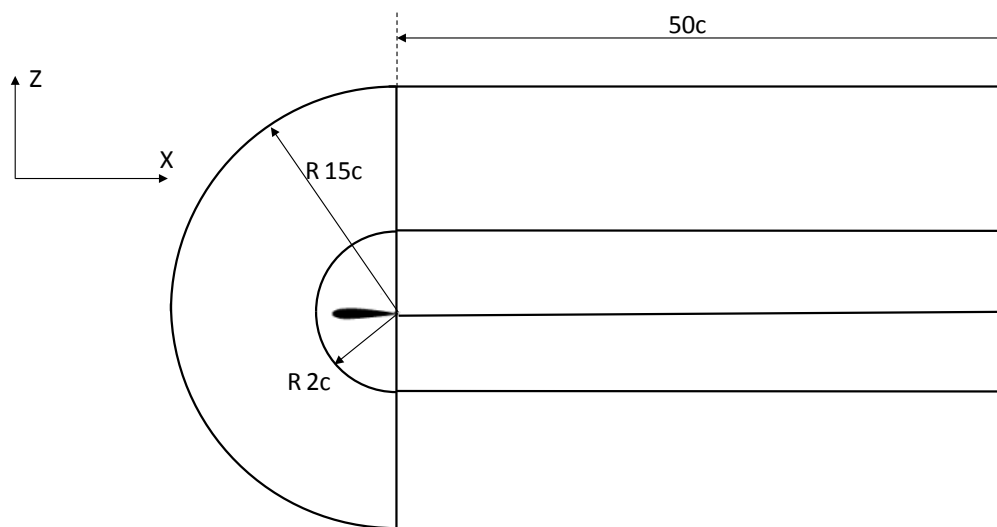
Figure 4.1 Fluid Flow Domain Used in CFD Simulations.
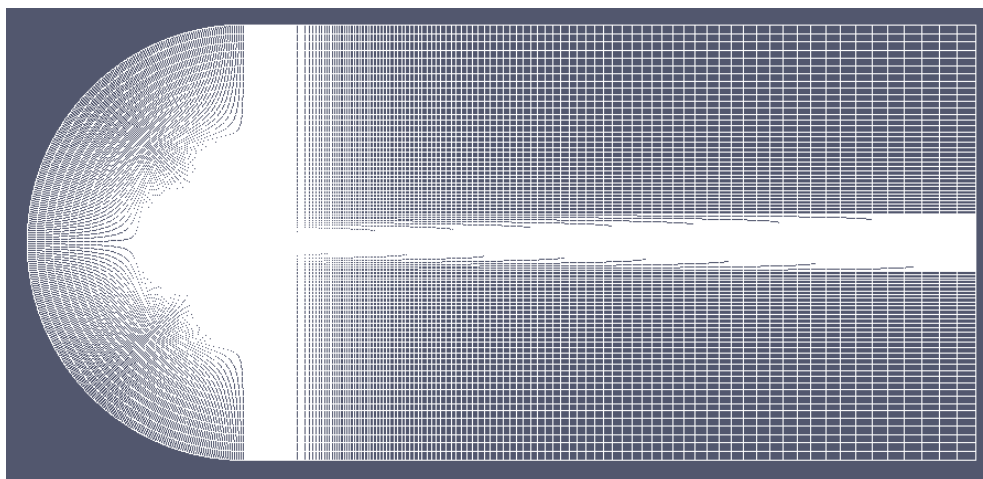
Figure 4.2 Grid with the Blade Section at 0º Angle of Attack.
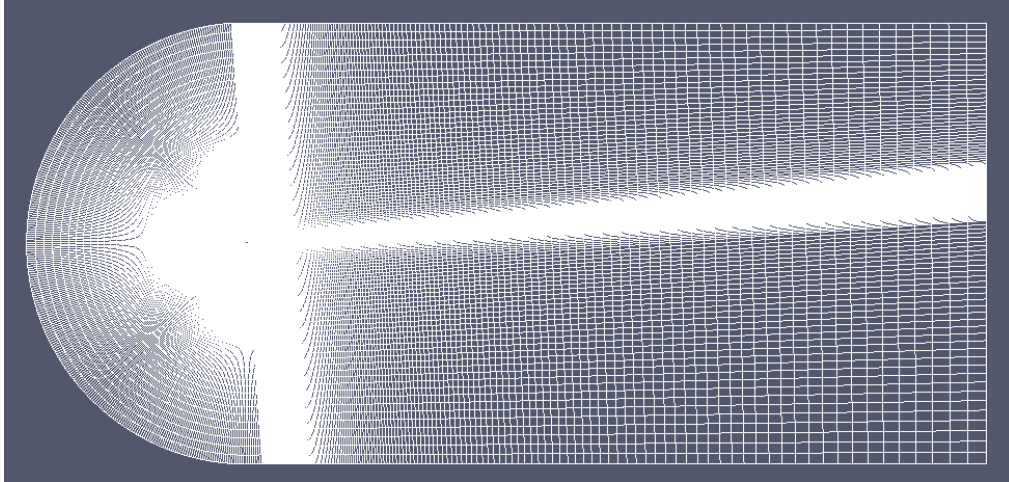
Figure 4.3 Grid with the Blade Section at -4º Angle of Attack.
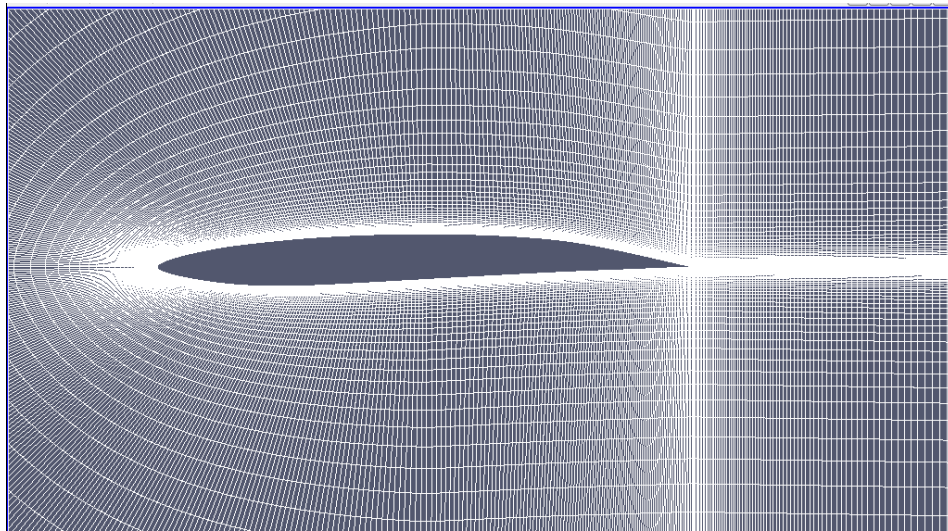
Figure 4.4 Mesh Details in the Vicinity of the Blade Section.

4.4 Description of the CFD Simulation Settings

The RANS equations and the four transport equations of the turbulence model are solved in OpenFOAM using the geometric-algebraic multigrid (GAMG) solver, the Gauss-Seidel smoother, and the semi-implicit method for pressure-linked equations (SIMPLE). The relaxation factor was set equal to 0.3 for the pressure and to 0.7 for all the other transported flow properties. OpenFOAM

is based on finite volume discretization with Gaussian integration, which requires the interpolation of the cell center values to the cell face centers. For the advection terms, the advective field needs to be interpolated to the cell faces; it was decided to use a second order, upwind-biased scheme for the velocity, the intermittency γ, and for the transition momentum-thickness Reynolds number $\widetilde{Re}_{\theta t}$, and the 'limitedLinear' scheme for κ and ω, which limits towards the first-order upwind scheme in regions of rapidly changing gradient and towards the second-order linear scheme in all others. The diffusion terms are discretized using a central-differencing scheme.

A fixed-velocity boundary condition was enforced at the inlet of the domain and a zero-gradient boundary condition was enforced at the outlet. For the pressure, the boundary conditions are zero-gradient at the inlet and a fixed value of zero at the outlet. At the inlet, the value of γ was set equal to one and the value of $Re_{\theta t}$ was set following the guidelines provided in [53]. The inlet value of ω is set equal to five times the value of the ratio of the freestream velocity U_∞ and the chord length c; the value of κ is set equal to $10^{-6} \cdot U_\infty^2$, which corresponds to a freestream turbulence intensity level of 0.08165% [54]. The boundary condition at the wall is zero-gradient for all transported properties. The blade sections were tested at a chord Reynolds number of 20 million with an average value of the dimensionless wall distance $y+ \simeq 2$ in order to resolve the flow all the way to the viscous sublayer without having to use wall functions. The CFD simulations were terminated when the scaled residuals of all the governing equations have been reduced by at least five orders of magnitude, provided that the values of the lift and drag coefficients had also converged.

4.5 Grid Convergence Study

A grid convergence study was carried out for the YS920 foil section [3] as per the guidelines

prescribed in [55]. The transition κ-ω SST model was used at a chord Reynolds number of 20

million. Three systematically refined grids with 30,000, 60,000, and 120,000 cells, respectively,

were tested according to the procedure described in the following paragraphs.

A representative cell size is first calculated using the formula,

$$h = \left[\frac{1}{N}\sum_{i=1}^{N} \Delta A_i\right]^{\frac{1}{2}} \tag{4.34}$$

where ΔA_i is the area of the ith cell and N is the number of cells. The grid refinement factor r is

then calculated. The summation of areas of individual cells yields the same value of the total

domain area and so the r value is,

$$r = \frac{h_{fine}}{h_{medium}} = \frac{h_{medium}}{h_{coarse}} = \sqrt{\frac{N_{fine}}{N_{medium}}} = \sqrt{\frac{N_{medium}}{N_{coarse}}} \approx 1.414 \tag{4.35}$$

The variables φ chosen for this study are the drag coefficient of drag C_D and the negative minimum

pressure coefficient $-C_{Pmin}$. Their apparent order is calculated by,

$$p = \frac{1}{\ln r}\ln|\varepsilon_{32}/\varepsilon_{21}| \tag{4.36}$$

where $\varepsilon_{32} = \varphi_3 - \varphi_2$ and $\varepsilon_{21} = \varphi_2 - \varphi_1$, with φ_k denoting the solution on the kth grid.

The extrapolated values of the variables are calculated from the formula,

$$\varphi_{ext}^{21} = \frac{r^p \varphi_1 - \varphi_1}{r^p - 1} \tag{4.37}$$

The approximate relative error is,

$$e_a^{21} = \left| \frac{\varphi_1 - \varphi_2}{\varphi_1} \right| \tag{4.38}$$

the extrapolated relative error is,

$$e_{ext}^{21} = \left| \frac{\varphi_{ext}^{12} - \varphi_1}{\varphi_{ext}^{12}} \right| \tag{4.39}$$

and the grid convergence index is,

$$GCI^{21} = \frac{1.25 e_a^{21}}{r^p - 1} \tag{4.40}$$

The results of the grid convergence study are tabulated in Table 4.2.

		φ			ε	
	Coarse	Medium	Fine	21	32
# of Cells	30000	60000	120000	-	-
C_D	$6.94 \cdot 10^{-3}$	$6.59 \cdot 10^{-3}$	$6.60 \cdot 10^{-3}$	$3.54 \cdot 10^{-4}$	$-1.41 \cdot 10^{-5}$
$-C_{Pmin}$	0.3982	0.3995	0.3997	$-1.30 \cdot 10^{-3}$	$-2.00 \cdot 10^{-4}$

	21-32
pC_D	9.30
pC_{Pmin}	5.40

	φ_{ext}		$e_a\%$		$e_{ext}\%$		GCI %	
	21	32	21	32	21	32	21	32
C_D	$7.0 \cdot 10^{-3}$	$6.6 \cdot 10^{-3}$	5.0948	0.2138	0.2109	0.0089	0.2642	0.0111
$-C_{Pmin}$	0.3980	0.3995	0.3265	0.0501	0.0594	0.0091	0.0742	0.0114

Table 4.2 Grid Convergence Index Calculation Results.

The negative value of $\varepsilon_{32} / \varepsilon_{21}$ for C_D is an indication of oscillatory convergence, however, considering the very low value of ε_{32}, we can presume that the 'exact' solution has been attained. The GCI values, which quantify the numerical uncertainty on the fine grid for C_D and $-C_{Pmin}$ are

0.0111% and 0.0114%, respectively, which imply that the solutions obtained with the fine grid are grid-independent. It was therefore decided to use the fine grid with 120,000 cells in all the CFD simulations that were performed.

4.6 Experimental Validation

The CFD solver was tested with both the standard ('SST') and the transition ('SST-LM') κ-ω SST models in order to validate the predictions with experimental data for three foil profiles.

4.6.1 NACA 2410 Profile

The NACA 2410 airfoil section was tested first for lift and drag calculations at angles of attack of 2 and 4 degrees. The simulations were performed at a chord Reynolds number of 9 million in order to compare the CFD results with the experimental data provided in [56]. The results listed in Table 4.3 show very good agreement for both models regarding the computation of the lift coefficient. However, the transition κ-ω SST model performs significantly better than the standard model in predicting the drag coefficient, mainly due to the inherent assumption in the standard κ-ω SST model that the flow is fully turbulent over the entire foil section.

		C_L	C_D	C_M	Iterations	$y+$
	Experiment	0.665	0.0070	-0.0485	-	
4 degrees	SST	0.660	0.0092	-0.0481	1590	1.11
	SST-LM	0.661	0.0079	-0.0476	2044	1.01
	Experiment	0.450	0.0060	-0.0485	-	
2 degrees	SST	0.445	0.0080	-0.0492	1381	1.10
	SST-LM	0.445	0.0073	-0.0489	1884	1.03

Table 4.3 CFD Results and Experimental Data for the NACA 2410 Profile.

4.6.2 G1Profile

The validation process was continued with the G1 foil section, which has been tested

experimentally by Bourgoyne et al. [57]. The G1 profile airfoil has a nearly flat pressure side and

a NACA 16 suction side which is modified with a rounded trailing edge bevel of apex angle 44°

(see Figure 4.5). The simulations were performed at a chord Reynolds number of 8 million. The

results reported in Table 4.4 show a much better agreement between the transition κ-ω SST model

results and the experimental data for both lift and drag. The convergence of C_L and C_D is shown

in Figures 4.6 and 4.7, respectively.
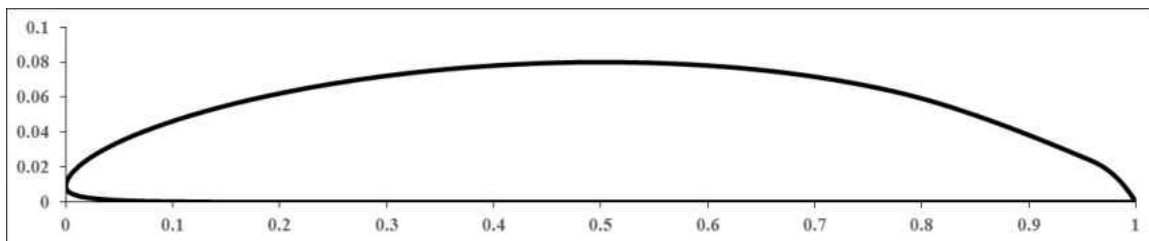
Figure 4.5 The G1 Profile.

		C_L	C_D	C_M	Iterations	y+
	Experiment	0.52	0.0060	-	-	-
0 degrees	SST	0.47	0.0083	-0.1125	2072	1.56
	SST-LM	0.51	0.0064	-0.1202	2097	1.29

Table 4.4 CFD Results and Experimental Data for the G1 Profile.
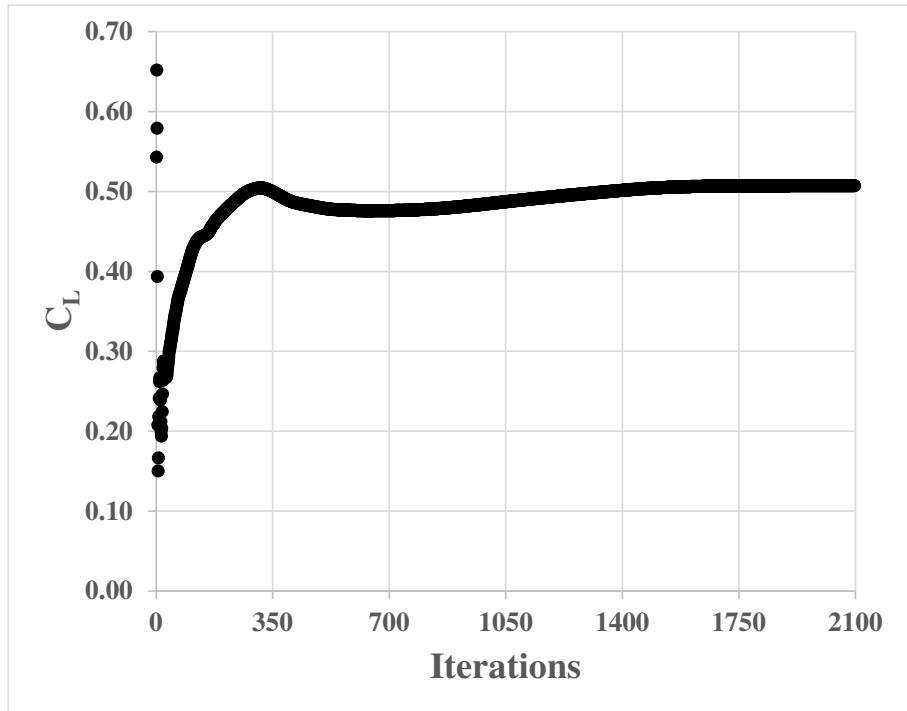
Figure 4.6 Convergence of C_L with the Transition κ-ω SST Model for the G1 Profile.

Figure 4.7 Convergence of C_D with the Transition κ-ω SST Model for the G1 Profile.

4.6.3 E817 Profile

The E817 foil section [58] was tested at a chord Reynolds number of 500,000 to check the validity of the computed negative minimum pressure coefficient values but also of the pressure distribution along the foil section. The experimental data reported here were obtained by Astolfi et al. [59]. The results listed in Table 4.5 demonstrate that the transition κ-ω SST model provides a more accurate prediction of the pressure coefficient on the suction side at higher values of the angle of attack. Similar to the conclusions drawn in Section 4.6.1, the predicted drag coefficient values are significantly lower than the values computed using the standard κ-ω SST model.

Astolfi et al. [59] also performed a comparison between the experimental values of $-C_{Pmin}$, the minimum negative pressure coefficient, and σ, the vapor cavitation number. The latter is defined as:

$$\sigma = \frac{p_\infty - p_v}{\frac{1}{2}\rho V^2} \tag{4.41}$$

where p_∞ is the ambient pressure, p_v is the fluid vapor pressure, and the term in the denominator is the dynamic stagnation pressure. The minimum pressure coefficient for the flow along a blade section can be written as:

$$-|C_{Pmin}| = \frac{p_{min} - p_\infty}{\frac{1}{2}\rho V^2} \tag{4.42}$$

where p_{min} is the minimum pressure along the section. When the resultant local fluid velocity V becomes sufficiently great so that $p_{min} = p_v$, cavitation inception will occur at that location along the section. For a propeller blade section, the velocity V can be calculated from equation 2.38. As shown in Table 4.5, there is fairly good agreement between the measured $-C_{Pmin}$ values and the σ values at which cavitation inception was observed.

		C_L	C_D	$-C_{Pmin}$	σ	# of Iterations	$y+$
0 degrees	Measured	-	-	0.52	0.51	-	-
	SST	0.401	0.0121	0.50	-	1038	0.18
	SST-LM	0.404	0.0083	0.50	-	3000	0.13
1 degree	Measured	-	-	0.53	0.56	-	-
	SST	0.507	0.0126	0.55	-	1122	0.18
	SST-LM	0.512	0.0083	0.54	-	3000	0.13
3 degrees	Measured	-	-	1.18	1.03	-	-
	SST	0.644	0.0135	1.02	-	1027	0.18
	SST-LM	0.662	0.0082	1.13	-	3000	0.13

Table 4.5 Computed and Measured [59] Data for the E817 Hydrofoil Profile.

Figures 4.8 and 4.9 illustrate the plots of $-C_P$ over the suction side for 0° and 3°, respectively. As shown in Figure 4.8, the transition κ-ω SST model provides an accurate prediction of the characteristic rooftop-type pressure distribution on the suction side in the region between 0.7c and the trailing edge.
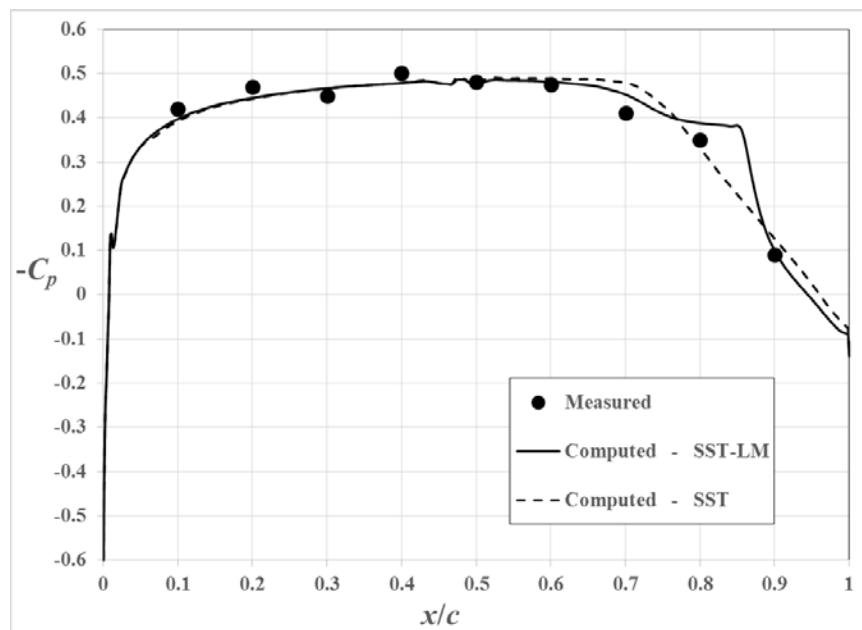
Figure 4.8 Pressure Coefficient Distribution on the Suction Side of E817 at 0°.

The transition $\kappa\text{-}\omega$ SST model also provides a more accurate prediction of the negative minimum

pressure coefficient value near the leading edge at the 3° angle of attack as shown in Table 4.5 and

in Figure 4.9. The standard $\kappa\text{-}\omega$ SST model significantly underpredicts the strong suction pressure

peak at the leading edge. This is of particular significance as the negative minimum pressure

coefficient is utilized as an indicator of cavitation inception in the investigation presented in

Chapter 5. Similar to the 0°-angle-of-attack case, the two models predict a different pressure

distribution in the region between 0.7c and the trailing edge.
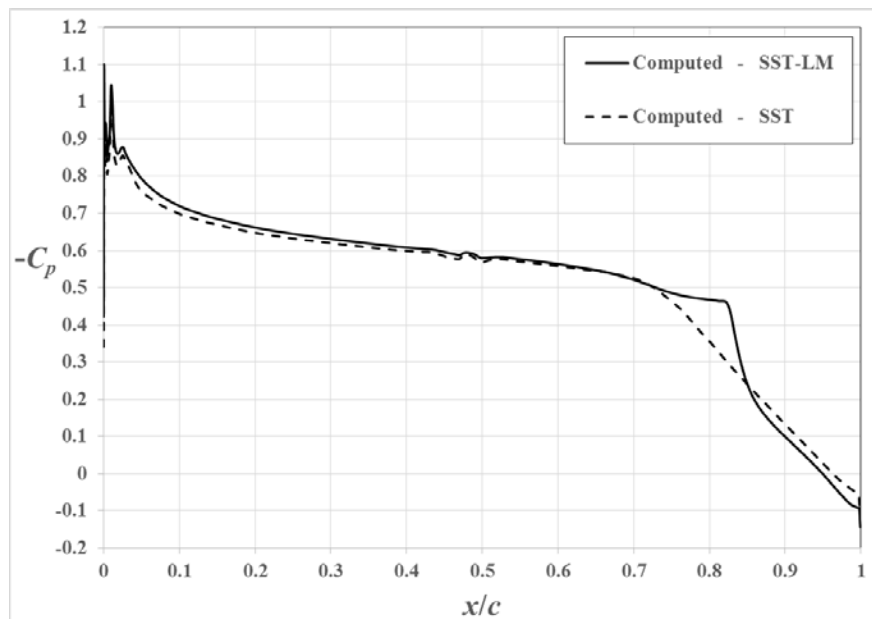
Figure 4.9 Pressure Coefficient Distribution on the Suction Side of E817 at 3°.

4.7 Evaluation of Baseline Propeller Blade Sections

The E817 profile [55], shown in Figure 4.10, and the YS920 profile [3, 60], shown in Figure 4.11,

were chosen as the baseline foil sections to be utilized in the optimization procedure presented in

Chapter 5 owing to their superior cavitation properties [34, 60]. Both profiles have a fairly thick

leading edge, which theoretically enables effective delay of the suction pressure peak [3]. The foils were tested at a chord Reynolds number of $2.0 \cdot 10^7$, a typical operating value of a full-scale ship propeller, using the transition κ-ω SST model. The E817 has a maximum-thickness-to-camber ratio of approximately 11% while the YS920 maximum-thickness-to-camber ratio is 9%.

Assuming a design lift coefficient $C_{Ld} = 0.3$ and a design cavitation number $\sigma_d = 0.6$, typical design values for propeller blade sections [1, 3], the angle of attack, the corresponding value of the drag coefficient, the value of $-C_{Pmin}$, and the cavitation-free angle range are listed in Table 4.6, in addition to the non-dimensional section modulus values (based on the chord length).

The negative minimum pressure coefficient values for various angles of attack (bucket diagram) for the E817 and the YS920 profiles are plotted in Figures 4.12 and 4.13, respectively. Based on the plot shown in Figure 4.12, the lift coefficient value of 0.3 corresponds to an angle near the midpoint of the bottom segment of the cavitation bucket.
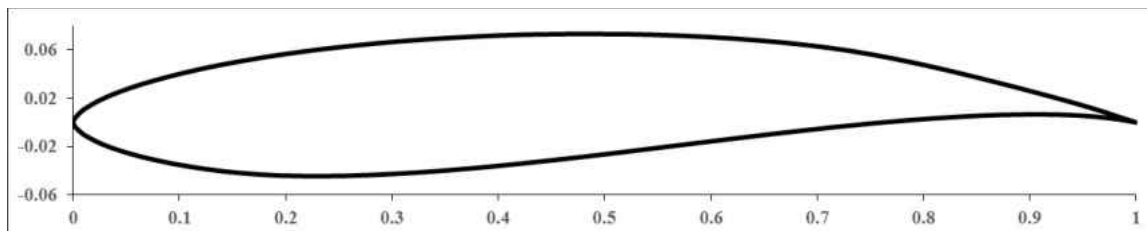
Figure 4.10 E817 Profile.

A comparison between the CFD results and the experimental data for the YS920 profile reported by Shen [60], albeit at a chord Reynolds number of $2.5 \cdot 10^6$, shows that the computed value of -

2.52° for the zero-lift angle is very close to the experimental value of 2.45°. The calculated lift-curve slope is 0.106 per degree, which is also in good agreement with the experimentally measured value of 0.103. The computed cavitation-free angle range at the cavitation number $\sigma = 0.45$ is 3.16° (see Figure 4.13), a value that is also very close to the experimental value of 3.20°.

	$(\alpha°)$	C_D	$-C_{Pmin}$	Bucket Width (°)	Section Modulus
E817	-1.40	$6.66 \cdot 10^{-3}$	0.477	3.91	$9.29 \cdot 10^{-4}$
YS920	0.16	$6.65 \cdot 10^{-3}$	0.405	3.85	$6.98 \cdot 10^{-4}$

Table 4.6 Analysis of Baseline Blade Sections.
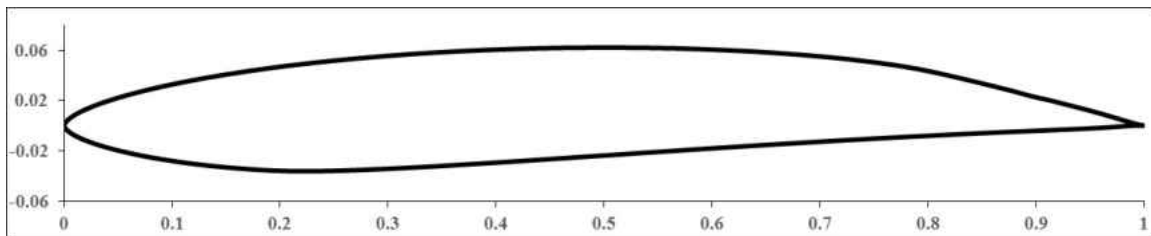
Figure 4.11 YS920 Profile.
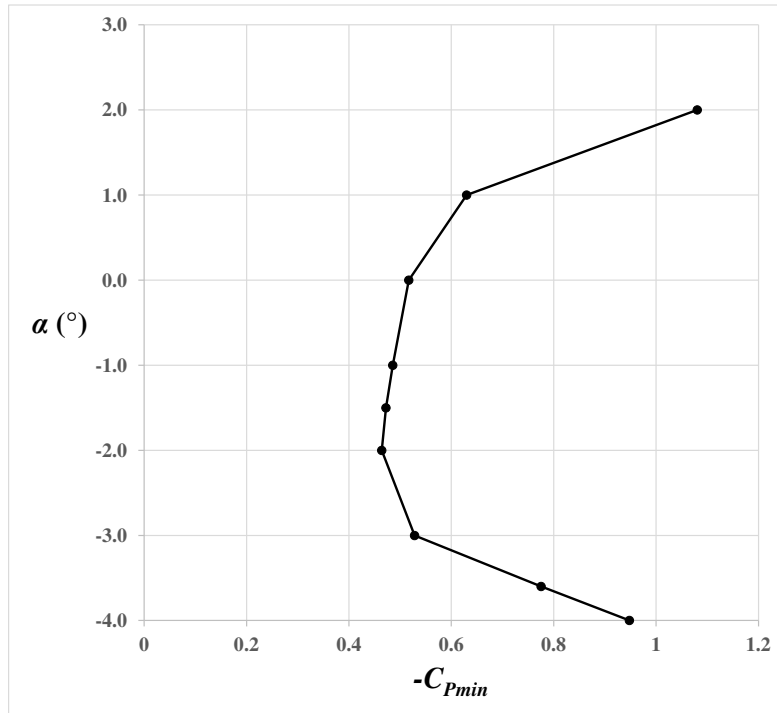
Figure 4.12: Cavitation-Free Bucket Diagram of E817 Profile.
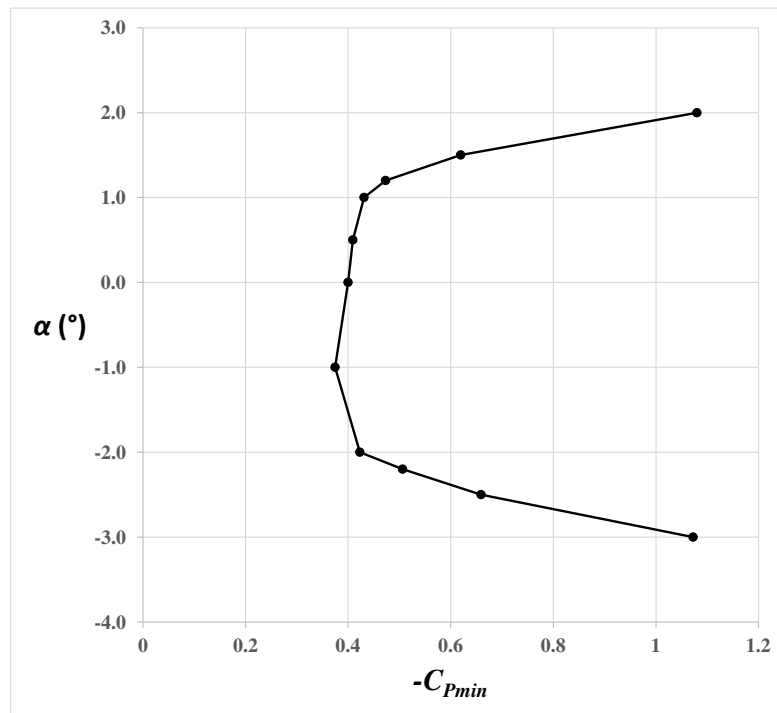
Figure 4.13 Cavitation-Free Bucket Diagram of YS920 Profile.

CHAPTER 5

DESIGN OF BLADE SECTIONS WITH GOOD CAVITATION PERFORMANCE AND LOW DRAG

5.1 Introduction

One of the most critical aspects of marine propeller design for high-speed applications corresponds to the design of the blade sections. The traditional formulation of the blade section design problem is to utilize a parent (baseline) foil section, e.g. the modified NACA 66 thickness form combined with the a = 0.8 mean line [1], and determine the angle of attack, the maximum-thickness-to-chord ratio and the maximum-camber-to-chord ratio that minimize the drag coefficient for a prescribed lift coefficient and cavitation number subject to constraints regarding the structural properties of the blade sections. For high-speed vessels, additional cavitation constraints can be imposed through the utilization of computational tools capable of predicting the characteristics of the cavity, i.e. volume and length. A successful application of this approach can be found in [7, 8].

A different approach, which was originally proposed by Shen and Eppler [2, 3] is to design blade sections with the main design goal being to suppress or delay cavitation inception over the widest possible range of angles of attack. The design problem in that case is formulated as follows: For a prescribed value of the blade section lift coefficient find the shape, i.e., thickness and camber distribution, that allows the largest variation in the angle of attack without occurrence of surface cavitation at a given blade section cavitation number. This range, which can be measured in the minimum pressure envelope of the profile, is also termed the bucket width (see Figure 5.1). The solution to this problem is obtained through a conformal mapping process where the section profile

is designed based on a prescribed pressure, or velocity, distribution in a way that allows the designer to control certain performance characteristics, e.g., the extent of the constant pressure region and the starting point of the pressure recovery region on both the pressure and suction sides of the blade section [4].

The YS920 [3] and E817 [58] profiles were designed using the aforementioned conformal mapping method. An additional successful application of this method for the design of a wake-adapted propeller for a frigate is demonstrated in [5]. The importance of considering two design goals, i.e., maximize the bucket width while maximizing the lift-to-drag ratio, was considered by Yamaguchi et al. [6] for the design of blade sections for the propeller of a high-speed vessel.

In this work, the shape optimization problem is formulated in the following way. For a given design lift coefficient C_{Ld} find the blade section that has the maximum cavitation-free range (CFR) at a given design cavitation number σ_d and minimum drag coefficient C_D at the corresponding design angle α_d while satisfying constraints regarding the section modulus (SM) and the minimum distance between the operating conditions and the bottom of the cavitation-free bucket. The design angle α_d is defined as the angle at which the design lift coefficient C_{Ld} is obtained. The inherent assumption in this approach is that cavitation occurs when the fluid pressure falls below the vapor pressure.

The section modulus of the blade section is utilized as a structural constraint in lieu of a blade section strength calculation. The minimum distance D_M between the operating conditions, i.e. the given design cavitation number σ_d, and the bottom of the bucket, as illustrated in Figure 5.1,

corresponds to the margin against bubble (mid-chord) cavitation. The latter occurs for angles of

attack in the vicinity of the ideal angle, i.e., the region at the bottom of the bucket.
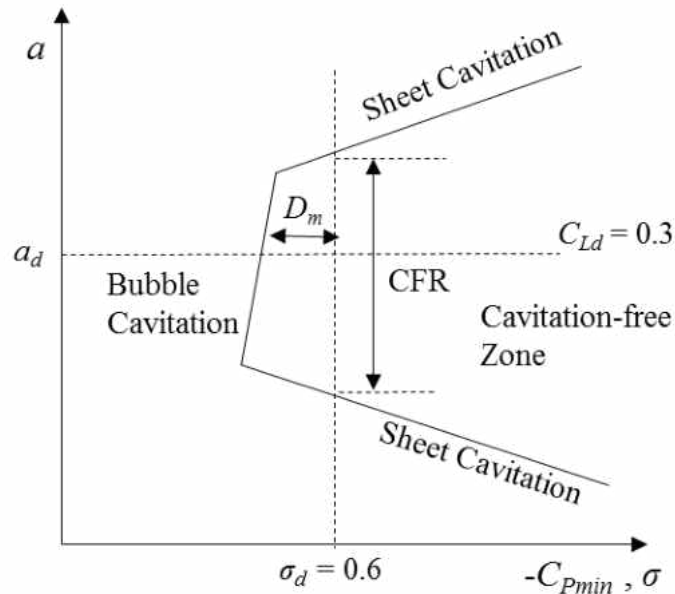
Figure 5.1 Schematic Illustration of Blade Section Bucket Diagram.

As will be demonstrated later in this chapter, the two design objectives are conflicting, i.e., there

is no single optimal solution but a set of optimal solutions that correspond to trade-offs between

the objectives.

5.2 Parameterization of the Blade Section Shape

The blade section is parameterized using two cubic B-splines; one for the upper and one for the

lower side. The B-splines are generated by utilizing eleven control points for each surface as shown

in Figure 5.2. Assuming that the chord length is equal to one, the first control point, which is

common to both sides, is placed on the leading edge with coordinates (0, 0). The last control point

is placed on the trailing edge with coordinates   (1, 0). Nine additional control points are placed

at intermediate positions on each side. The abscissa of each of those points is kept fixed: [0, 0.1,

0.2, 0.3, 0.5, 0.7, 0.8, 0.9, 0.95] and their ordinates are varied in order to generate different profiles.
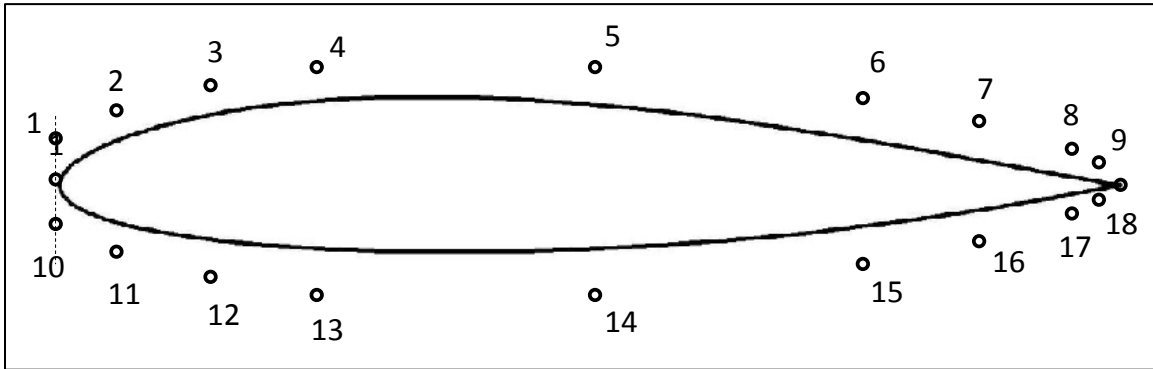
Figure 5.2 Distribution of Control Points of Cubic B-Splines.

The range of the ordinate of each control point is given in Table 5.1. The ordinates of the 18 control

points form the vector of design variables **x**; the ranges listed in Table 5.1 also correspond to the

bound constraints of the optimization problem. Those ranges were based on the values that were

used to generate the baseline designs (Profiles E817 and YS920).

5.3 Formulation of the Multi-Objective Shape Optimization Problem

The blade section shape optimization problem is defined as follows:

$$\text{Maximize:} \quad f_1(\mathbf{x}) = (\text{CFR}) @ \sigma_d \tag{5.1}$$

$$\text{Minimize} \quad f_2(\mathbf{x}) = C_D @ \alpha_d \tag{5.2}$$

$$\text{Subject to:} \quad g_1(\mathbf{x}) = D_M \geq 0.05 \tag{5.3}$$

$$g_2(\mathbf{x}) = SM \geq 8.5 \cdot 10^{-4} \tag{5.4}$$

where the design lift coefficient $C_{Ld} = 0.30$ and the design cavitation number $\sigma_d = 0.60$. The vector

of design variables $\mathbf{x}$ consists of the 18 control points, and the ranges listed in Table 5.1 are the

corresponding lower and upper bound values of each design variable j, x_{jl} and x_{ju}, respectively.

The vector $\mathbf{x}$ is defined as: $\mathbf{x} = (x_1, x_2, \ldots, x_{18})$, $x_{jl} \leq x_j \leq x_{ju}$, $j = 1, \ldots, 18$.

Control Point	Min Ordinate	Max Ordinate
1	0.0101	0.0133
2	0.0350	0.0442
3	0.0460	0.0565
4	0.0560	0.0690
5	0.0610	0.0762
6	0.0530	0.0665
7	0.0375	0.0505
8	0.0200	0.0271
9	0.0115	0.0142
10	-0.0135	-0.0105
11	-0.0400	-0.0309
12	-0.0464	-0.0370
13	-0.0454	-0.0355
14	-0.0325	-0.0235
15	-0.0195	-0.0047
16	-0.0130	0.0019
17	-0.0075	0.0070
18	-0.0028	0.0073

Table 5.1 Ordinate Range of each Control Point of the Cubic B-Splines.

5.4 Surrogate Models of Objective and Constraint Functions

The multi-objective shape optimization problem was solved using surrogate models of the

objective and constraint functions. The open source data mining software WEKA [41] was utilized

to obtain the surrogate models, which were based on ANNs. The ANNs were developed and

trained using a data set of points selected from within the design variable space according to an

experimental design methodology, Uniform Design [61, 62], which is also applicable to numerical simulations. Uniform Design is a statistical technique that generates a fractional factorial design and has been shown [63] to be a very effective sampling strategy even when small data sets are available. Therefore, it was decided to use a $U_{121}(11^{18})$ uniform design for 121 runs with 18 factors (design variables) each having 11 discrete levels. Considering the continuous nature of the design variables, each one of the ranges listed in Table 5.1 was discretized into 11 equispaced values. The 121 blade sections were analyzed using OpenFOAM at a chord Reynolds number of 20 million in order to obtain the corresponding values of $f_1(\mathbf{x})$, $f_2(\mathbf{x})$, and $g_1(\mathbf{x})$, and generate the data set of training points.

5.5 Description of the Multi-Objective Optimization Process

The multi-objective shape optimization problem formulated in Section 5.3 was solved using the Adaptive Coevolutionary Multi-Objective Particle Swarm Optimizer (ACMOPSO) [64, 65]. ACMOPSO explores the design variable space using the search mechanisms of particle swarm optimization [66]; a bio-inspired search algorithm that has been developed in the field of evolutionary computation [67]. As already mentioned in Section 5.1, when the optimization objectives are conflicting, there is no single optimal solution but a set of optimal solutions that correspond to trade-offs between the objectives. This set of feasible optimal solutions is termed the Pareto-optimal set, and its representation in the objective function space is termed the Pareto-optimal front.

The concept of Pareto optimality is briefly described in this section. In unconstrained optimization problems, solution vector $\mathbf{x}_A$ dominates solution vector $\mathbf{x}_B$, if and only if the following two conditions for Pareto dominance are both satisfied:

- The objective vector that corresponds to solution $\mathbf{x}_A$ is no worse than the objective vector that corresponds to solution $\mathbf{x}_B$ in all objectives.

- Solution $\mathbf{x}_A$ is strictly better than solution $\mathbf{x}_B$ in at least one objective.

If solution $\mathbf{x}_A$ is not dominated by any other solution, it is called a Pareto-optimal solution, and the corresponding objective function vector belongs to the set of vectors that comprise the Pareto-optimal front. In constrained problems, like the problem formulated in section 5.3, $\mathbf{x}_A$ constraint-dominates $\mathbf{x}_B$ if any of the following conditions are satisfied [68]:

- Both $\mathbf{x}_A$ and $\mathbf{x}_B$ are feasible and $\mathbf{x}_A$ dominates $\mathbf{x}_B$ based on the aforementioned conditions for Pareto dominance.

- Both $\mathbf{x}_A$ and $\mathbf{x}_B$ are infeasible but $\mathbf{x}_A$ has a smaller constraint violation.

- Solution $\mathbf{x}_A$ is feasible and solution $\mathbf{x}_B$ is not.

In the current work, a version of ACMOPSO with five swarms was employed for the solution of the optimization problem defined in Equations 5.1 through 5.4. It was shown in [64] that by applying ACMOPSO with five swarms to a number of benchmark problems, the search algorithm was able to converge to the true Pareto-optimal front in a very fast manner.

ACMOPSO was run for 500 iterations per optimization cycle using five swarms with twenty particles, i.e. solution vectors, in each. At the end of the first cycle, six solutions were selected from the computed Pareto-optimal front. The corresponding blade section shapes were generated

using cubic B-splines, and their performance was evaluated using OpenFOAM. The results were

added to the existing dataset of 121 blade sections and the surrogate models were reconstructed

in WEKA. This iterative process was repeated until the computed optimal set showed minimal

changes between two successive iterations. A total of three optimization cycles was required in

order to obtain the final Pareto-optimal front. A flowchart of the optimization process is shown in

Figure 5.3.
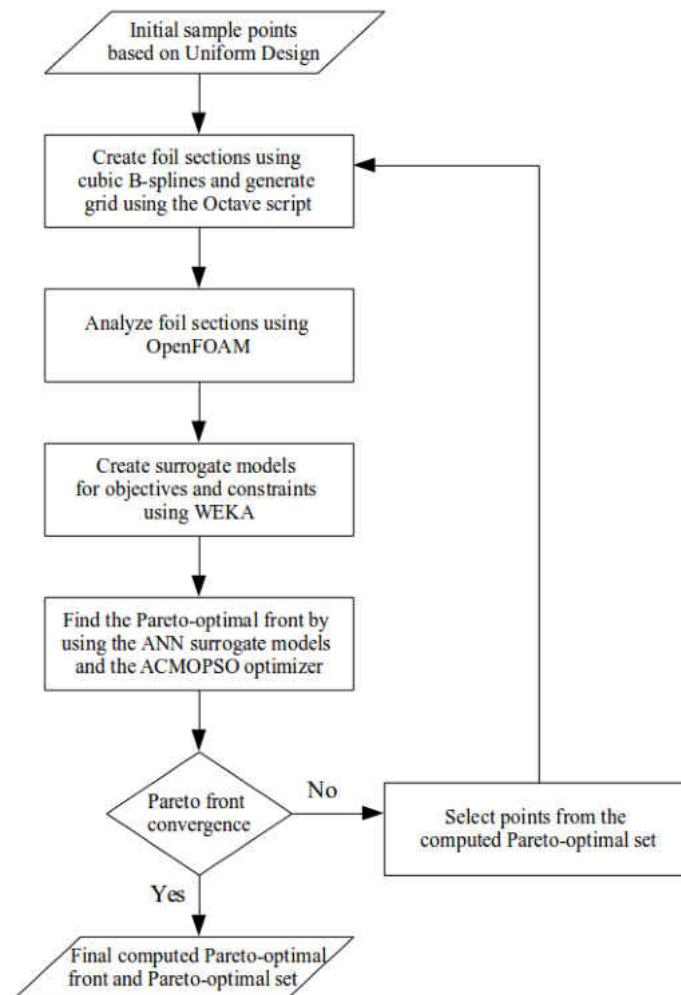
Figure 5.3 Flowchart of Multi-Objective Shape Optimization Process.

5.6 Multi-Objective Shape Optimization Results

The Pareto-optimal front that was obtained after performing three optimization cycles is shown in Figure 5.4. The trade-off between minimizing the drag coefficient at the design angle and maximizing the bucket width at the operating conditions is clearly illustrated. It needs to be noted that all the depicted solutions are feasible, i.e. satisfy the constraints listed in Equations 5.3 and 5.4. It was decided to select one Pareto-optimal solution and analyze it using OpenFOAM in order to verify the prediction accuracy of the surrogate models, but also to compare the Pareto-optimal solution with the E817 profile and validate the claim that the implemented design optimization procedure can produce solutions with superior capabilities compared to the baseline profiles.

Among the obtained Pareto-optimal solutions, a solution vector located in the 'knee' section of the front was selected; the 'knee' corresponds to the part of the front with solutions that provide a balanced trade-off between low drag and wide cavitation bucket width. The selected solution is highlighted in Figure 5.4 and the corresponding profile is shown in Figure 5.5. The coordinates of the profile are available in Appendix I. A comparison between the values predicted by the surrogate models and the actual values of the objectives and constraints is presented in Table 5.2. It can be observed that all the predicted values are fairly close to the actual values. The cavitation bucket diagram of the selected profile is shown in Figure 5.6.

	C_D	D_M	Bucket Width (°)	Section Modulus
Actual	$6.50 \cdot 10^{-3}$	0.108	4.25	$9.27 \cdot 10^{-4}$
Predicted	$6.43 \cdot 10^{-3}$	0.112	4.21	$9.23 \cdot 10^{-4}$

Table 5.2 Actual vs. Predicted Values of Objectives and Constraints.
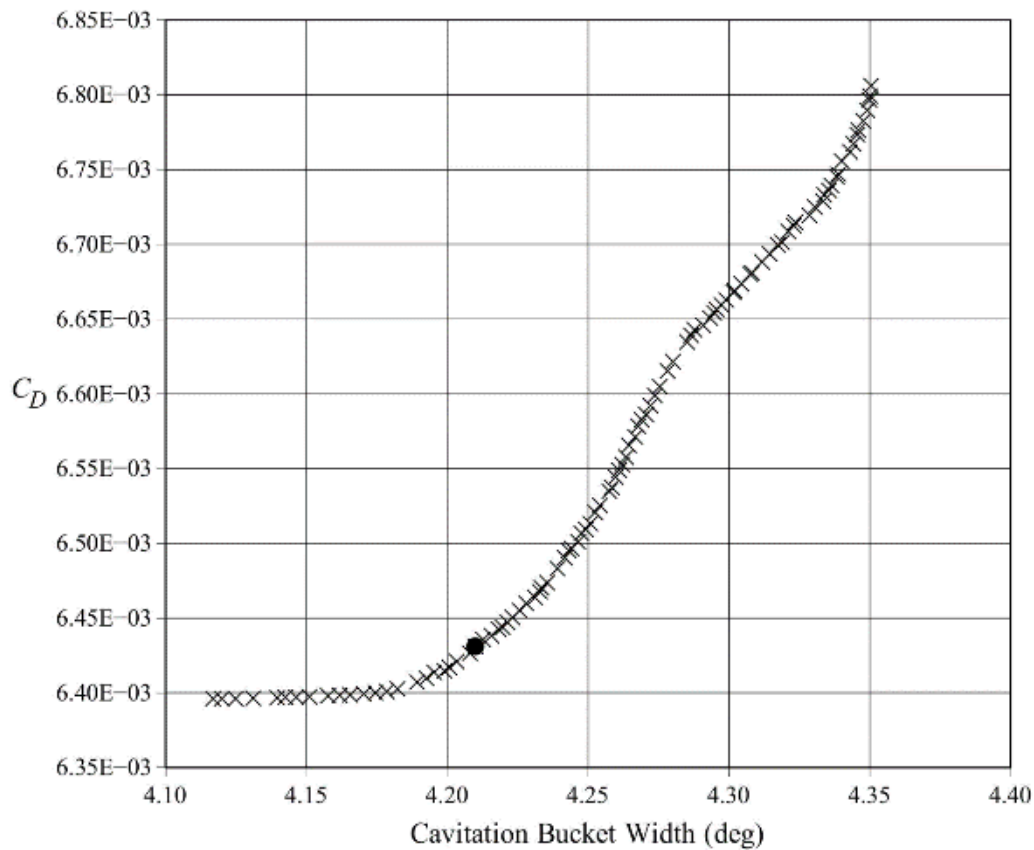
Figure 5.4 Computed Pareto-Optimal Front.
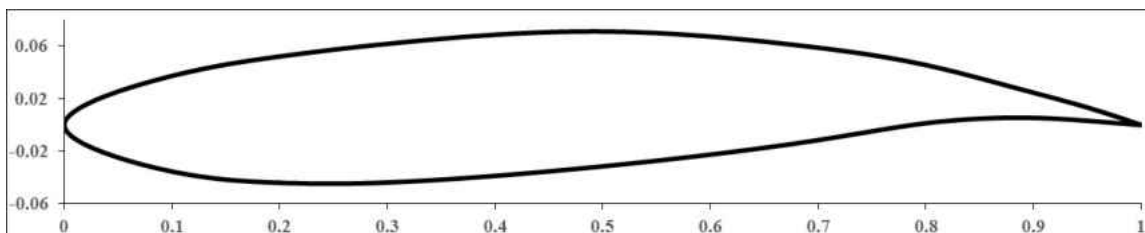
Figure 5.5 Pareto-Optimal Blade Section Profile.
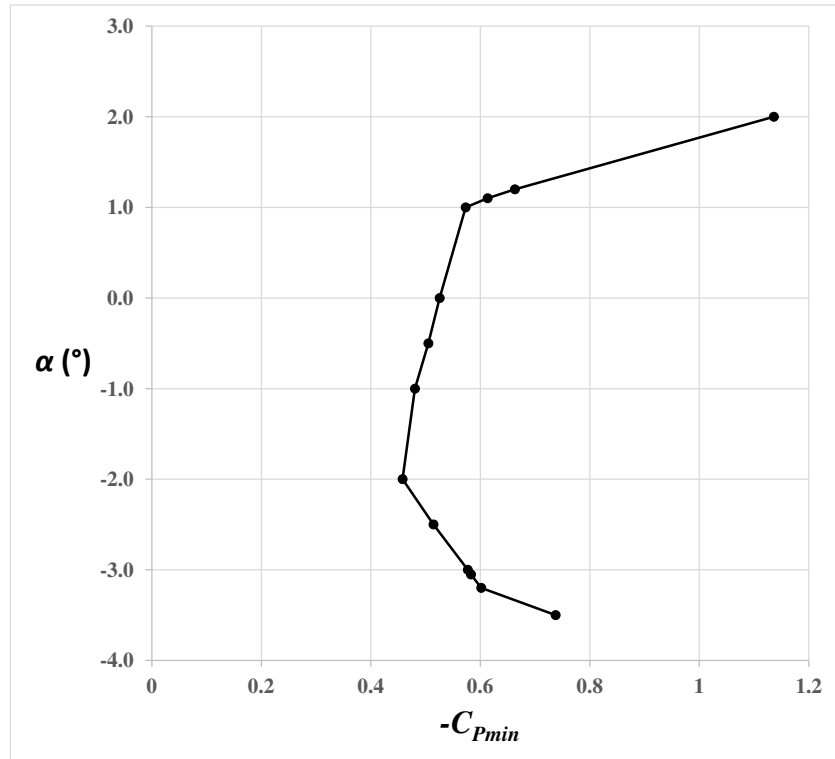
Figure 5.6 Bucket Diagram of Pareto-Optimal Profile.

5.7 Comparison between the Pareto-Optimal and the E817 Profiles

The two profiles are depicted in Figure 5.7, their geometric characteristics are listed in Table 5.3, and the results of their hydrodynamic performance analysis are listed in Table 5.4. The combined cavitation bucket diagrams are shown in Figure 5.8.

A comparison between the geometric characteristics of the two profiles reveals that the maximum-thickness-to-chord-length ratio of the selected Pareto-optimal profile is slightly smaller than the E817 profile value, however, the section modulus values of the two profiles are almost identical. The location of the maximum-thickness-to-chord-length ratio $x_{t,max}/c$ and of the maximum-camber-to-chord-length ratio $x_{f,max}/c$ along the new profile are both farther aft than the corresponding locations along the E817 profile. Lower camber near the leading edge can lead to a

widening of the cavitation bucket [6], which is evident from the data reported in Table 5.4. Furthermore, the lower drag coefficient value of the new profile at the design condition can be attributed to its relatively thinner forward part compared to that of the E817 profile, as shown in Figure 5.7.

Even though the E817 profile has a slightly larger bucket width at lower cavitation numbers, the selected Pareto-optimal profile achieves the design goals of the formulated optimization problem at the prescribed operating conditions. The observed improvement in both design objectives demonstrates the effectiveness of the proposed design optimization method.

	t_{max}/c (%)	$x_{t,max}/c$ (%)	f_{max}/c (%)	$x_{f,max}/c$ (%)	Section Modulus
E817	11.1	32.5	2.90	69.0	$9.29 \cdot 10^{-4}$
Optimal	10.8	38.1	2.43	75.8	$9.27 \cdot 10^{-4}$

Table 5.3 Geometric Characteristics of Selected Pareto-Optimal and E817 Profiles.

	a_d (deg)	C_D	$-C_{Pmin}$	Bucket Width (°)	D_M
E817	-1.40	$6.66 \cdot 10^{-3}$	0.477	3.91	0.123
Optimal	-0.76	$6.50 \cdot 10^{-3}$	0.492	4.25	0.108

Table 5.4 Hydrodynamic Analysis of Selected Pareto-Optimal and E817 Profiles.
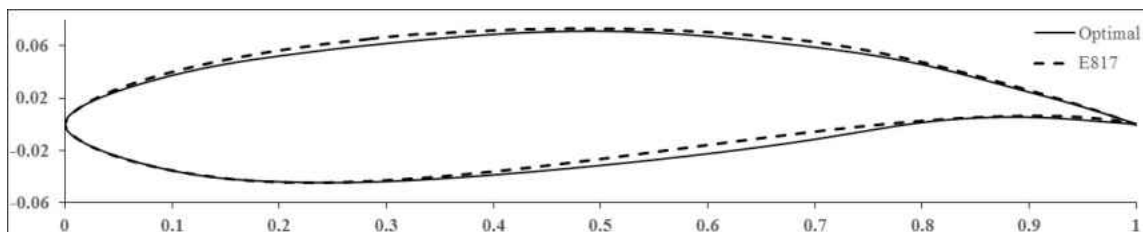
Figure 5.7 Comparison between the Selected Pareto-Optimal and the E817 Profile.
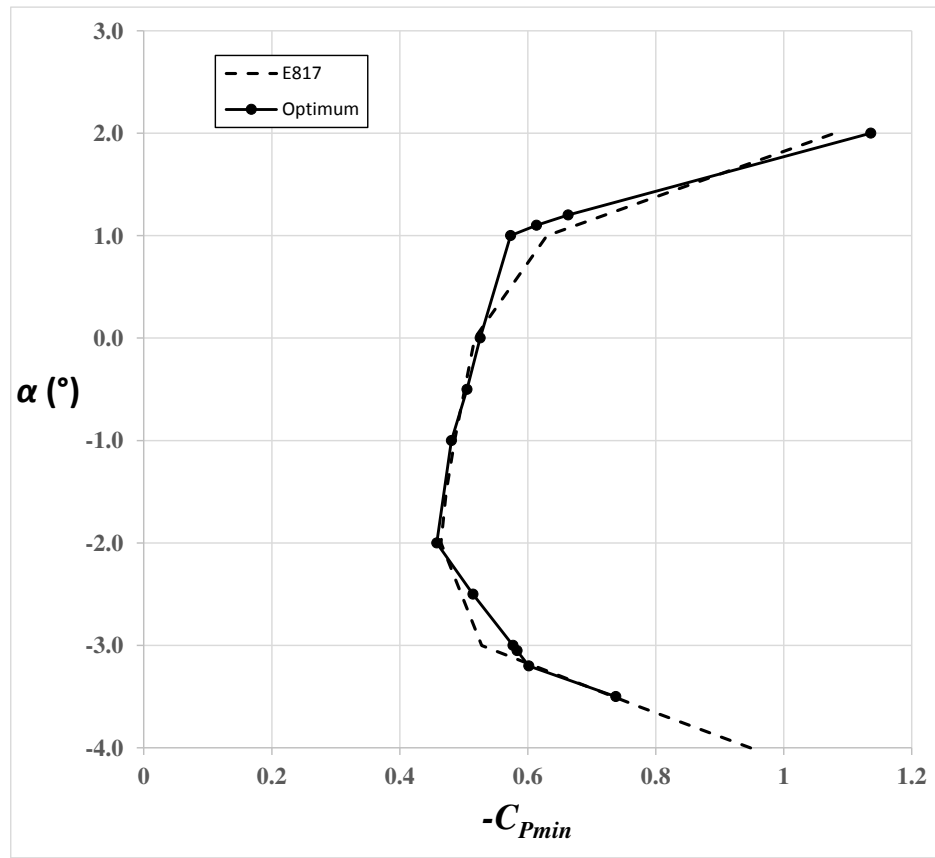
Figure 5.8 Bucket Diagrams of the Selected Pareto-Optimal and E817 Profiles.

CHAPTER 6

VALIDATION OF PROPELLER ANALYSIS AND DESIGN FRAMEWORK

6.1 Validation in Open-Water Tests

The initial validation of the lifting line code was carried out for three propellers operating in open-water conditions. All three propellers utilize blade sections with the modified NACA 66 thickness distribution [1] with the a = 0.8 camber line [56]. The experimental data available in the literature and the numerical results were obtained with the propellers operating in non-cavitating conditions.

6.1.1 DTMB 4119 Propeller

The DTMB 4119 is a three-bladed propeller with zero skew and design advance coefficient $J_d =$ 0.833 [69]. The geometric characteristics of the propeller are given in Table 6.1 as a function of the radial position. These characteristics are: the chord-length-to-diameter ratio c/D, the geometric-pitch-to-diameter ratio P/D, the maximum-camber-to-chord-length ratio f_{max}/c, and the maximum-thickness-to-chord-length ratio t_{max}/c.

r/R	c/D	P/D	f_{max}/c	t_{max}/c
0.2	0.3200	1.105	0.0143	0.2055
0.3	0.3625	1.102	0.0232	0.1553
0.4	0.4048	1.098	0.0230	0.1180
0.5	0.4392	1.093	0.0218	0.0902
0.6	0.4610	1.088	0.0207	0.0696
0.7	0.4622	1.084	0.0200	0.0542
0.8	0.4347	1.081	0.0197	0.0421
0.9	0.3613	1.079	0.0182	0.0332
0.95	0.2775	1.077	0.0163	0.0323
1.0	0.0020	1.075	0.0118	0.0316

Table 6.1 Geometric Characteristics of the DTMB 4119 Propeller.

The computed data correspond to the thrust and torque coefficients, K_T and K_Q, which are defined as follows:

$$K_T = \frac{T}{\rho n^2 D^4}$$

(6.1)

$$K_Q = \frac{Q}{\rho n^2 D^5}$$

(6.2)

where the thrust T and torque Q are computed from equations 2.41a and 2.41b, respectively, n is the propeller rotational speed, D is the propeller diameter, and ρ is the water density.

These are computed at the design advance coefficient J_d but also at off-design conditions. The advance coefficient is defined as:

$$J = \frac{V}{nD}$$

(6.3)

where V is the speed of the inflow to the propeller. As can be deduced from observing Figures 6.1 through 6.3, but also from Figure 2.1, the advance coefficient is inversely proportional to the angle of attack of the blade section with respect to the chord line.

The thrust and torque were computed using the proposed lifting line ('LL') method that was described in Chapter 2, with and without the lifting surface correction ('LSC') factors. The computed results, along with the experimental results obtained by Jessup [70], are listed in Table 6.2.

The lifting line code without the lifting surface correction factors significantly overestimates both thrust and torque at any given advance coefficient value. A similar outcome for the DTMB 4119 propeller was demonstrated in [22]. On the other hand, the predictions including the lifting surface

corrections are in very good agreement with the experimental results at the design but also at off-design conditions.

	Experimental		LL with LSC		LL without LSC	
J	K_T	K_Q	K_T	K_Q	K_T	K_Q
0.7	0.207	0.0363	0.2289	0.0396	0.4079	0.0760
0.833	0.155	0.0280	0.1383	0.0274	0.2949	0.0583
0.9	0.123	0.0243	0.0929	0.0208	0.2394	0.0492

Table 6.2 Computed and Experimental Data for the DTMB 4119 Propeller.

6.1.2 DTNSRDC 4381 and 4382 Propellers

The DTNSRDC 4381 and 4382 propellers were developed and tested at the David Taylor Naval Ship Research and Development Center (DTNSRDC) and the experimental data were obtained by Boswell [71]. The geometric characteristics of the propellers (taken from [15]) are given in Tables 6.3 and 6.4, respectively, and for the 4382 propeller include the skew angle θ_s.

r/R	c/D	P/D	f_{max}/c	t_{max}/D
0.2	0.174	1.332	0.0351	0.0434
0.25	0.202	1.338	0.0369	0.0396
0.3	0.229	1.345	0.0368	0.0358
0.4	0.275	1.358	0.0348	0.0294
0.5	0.312	1.336	0.0307	0.0240
0.6	0.337	1.280	0.0245	0.0191
0.7	0.347	1.210	0.0191	0.0146
0.8	0.334	1.137	0.0148	0.0105
0.9	0.280	1.066	0.0123	0.0067
0.95	0.210	1.031	0.0128	0.0048
1.0	0.0	0.995	-	0.0029

Table 6.3 Geometric Characteristics of the DTNSRDC 4381 Propeller.

r/R	c/D	P/D	f_{max}/c	t_{max}/D	θ_s (deg)
0.2	0.174	1.455	0.0430	0.0434	0.00
0.25	0.202	1.444	0.0395	0.0396	2.328
0.3	0.229	1.433	0.0370	0.0358	4.655
0.4	0.275	1.412	0.0344	0.0294	9.363
0.5	0.312	1.361	0.0305	0.0240	13.948
0.6	0.337	1.285	0.0247	0.0191	18.378
0.7	0.347	1.200	0.0199	0.0146	22.747
0.8	0.334	1.112	0.0161	0.0105	27.145
0.9	0.280	1.027	0.0134	0.0067	31.575
0.95	0.210	0.985	0.0140	0.0048	33.788
1.0	0.0	0.942	-	0.0029	36.000

Table 6.4 Geometric Characteristics of the DTNSRDC 4382 Propeller.

The computed data at the design advance coefficient but also at off-design conditions using the proposed lifting line method with and without the lifting surface correction factors, and the experimental results obtained by Boswell [71] for the 4381 and 4382 propellers are listed in Tables 6.5 and 6.6, respectively. The lifting line code without the lifting surface correction factors significantly overestimates both thrust and torque at any given advance coefficient value for both propellers. The lifting surface correction factors utilized for the 4381 propeller are provided by the surrogate models that were derived using the Morgan et al. [20] data. On the other hand, for the 4382 propeller, the utilized surrogate models were the ones derived based on the Cumming et al. [21] data for highly-skewed propellers. The predictions with the LSC factors included are in very good agreement with the experimental data for both propellers, at the design but also at off-design conditions. For the 4381 propeller operating at the design condition, the error in K_T is 4.3% and the error in K_Q is 2.3%. The corresponding errors for the 4382 propeller are 1.0% and 2.4%, respectively.

	Experimental		LL with LSC		LL without LSC	
J	K_T	K_Q	K_T	K_Q	K_T	K_Q
0.7	0.290	0.057	0.3076	0.0590	0.3661	0.0744
0.8	0.245	0.050	0.2480	0.0499	0.3009	0.0635
0.889	0.205	0.043	0.1959	0.0416	0.2451	0.0538

Table 6.5 Computed and Experimental Data for the DTNSRDC 4381 Propeller.

	Experimental		LL with LSC		LL without LSC	
J	K_T	K_Q	K_T	K_Q	K_T	K_Q
0.7	0.285	0.0550	0.3351	0.0613	0.4036	0.0824
0.8	0.245	0.0490	0.2634	0.0511	0.3298	0.0701
0.889	0.198	0.0425	0.2001	0.0414	0.2668	0.0592

Table 6.6 Computed and Experimental Data for the DTNSRDC 4382 Propeller.

6.2 KCS Propeller

The KCS propeller was specifically designed for a modern 3,600 TEU container ship that was conceived by the Korea Research Institute for Ships and Ocean Engineering (KRISO) in order to provide experimental data that could be utilized as benchmark for CFD validation purposes [72]. The KRISO container ship (KCS) has not been built in full scale. The KCS propeller has five blades, an expanded blade area ratio $A_e/A_0 = 0.80$, zero rake, and a tip skew angle of 32°. The blade sections have the NACA 66 thickness distribution combined with the a = 0.8 camber line [56]. Its geometric characteristics are listed in Table 6.7. The propeller has been tested in both open-water and behind-the-ship-hull conditions at the Ship Research Institute (now NMRI) in Tokyo [73]. For the latter case, i.e with the propeller operating in the ship wake, it was determined that the self-propulsion point of the KRISO container ship is at a K_T value of 0.1703. Operation of the propeller in non-cavitating conditions has also been assumed for this part of the numerical investigation.

r/R	P/D	c/D	f_{max}/c	t_{max}/D
0.18	0.8347	0.2313	0.02845	0.04585
0.25	0.8912	0.2618	0.02964	0.04071
0.3	0.9269	0.2809	0.02948	0.03712
0.4	0.9783	0.3138	0.02677	0.03047
0.5	1.0079	0.3403	0.02201	0.02459
0.6	1.0130	0.3573	0.01732	0.01947
0.7	0.9967	0.3590	0.01404	0.01492
0.8	0.9566	0.3376	0.01199	0.01073
0.9	0.9006	0.2797	0.01044	0.00693
0.95	0.8683	0.2225	0.01007	0.00528
1.0	0.8331	0.0001	-	0.00369

Table 6.7 Geometric Characteristics of the KCS-SRI Propeller.

6.2.1 Open-Water Performance Characteristics with Original Blade Sections

The thrust and torque coefficients were computed, first, for open-water operating conditions. The computed values of K_T and K_Q over the range of advance coefficient values between 0.4 and 0.75 are plotted against the experimental data [73] in Figures 6.1 and 6.2, respectively. The average error between the computed results and the experimental data is 5.1% and 2.7% for K_T and K_Q, respectively.
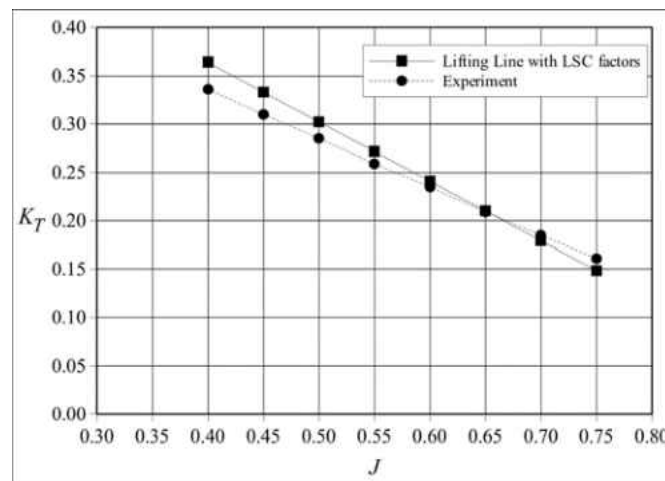
Figure 6.1 Plot of K_T vs. J for the KCS Propeller.
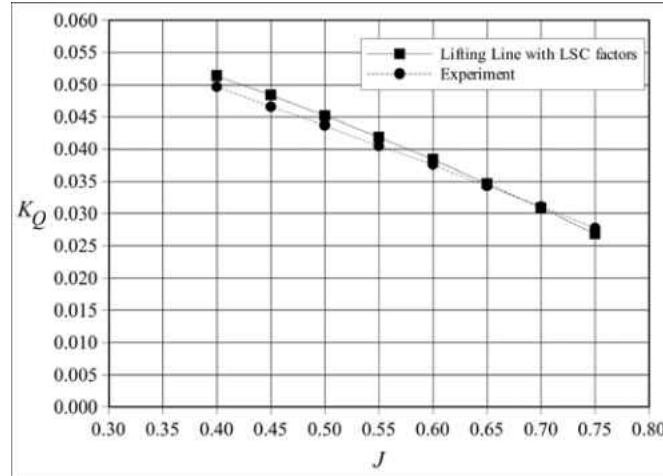
Figure 6.2 Plot of K_Q vs. J for the KCS Propeller.

6.2.2 Open-Water Performance Characteristics with New Blade Sections

Considering that the experimentally determined self-propulsion point of the KRISO container ship is available, it was decided to evaluate the effectiveness of the optimal blade section profile that was developed in Chapter V. The first step is to compute the K_T and K_Q distribution using the KCS propeller geometry provided in Table 6.7 combined with the new blade section profile. This task required the scaling of the foil profile using the thickness and camber distributions from Table 6.7, the computation of the lift and drag of the two-dimensional blade sections, and finally the computation of K_T and K_Q via the lifting line code with the LSC factors. The results are displayed in Figure 6.3, including the propeller efficiency η, which is calculated as:

$$\eta = \frac{K_T J}{2\pi K_Q} \tag{6.4}$$

6.2.3 Operation at the Ship Self-Propulsion Point

As mentioned at the beginning of Section 6.2, the thrust requirement of the full-scale ship corresponds to the experimentally determined value of $K_T = 0.1703$. The open-water curves that

were computed using the lifting line code for the KCS propeller with the original NACA sections and with the new blade sections developed in Chapter 5 can be utilized in order to find the corresponding advance coefficient, torque coefficient, and propeller efficiency. These results are reported in Table 6.8. When the new blade sections are utilized for the KCS propeller, the operating point is shifted to a higher advance coefficient, which can be attributed to the ability of the profile to produce the required lift at fairly small angles of attack, as evidenced in the results listed in Table 5.4. By employing the new blade section profile, an approximately 2.5% increase in the propulsion efficiency is observed. This number is obtained under the assumption that the effective wake is not affected by the utilization of a different section profile. The interaction between the propeller and the ship hull would have to be analyzed in order to obtain a more accurate estimate.
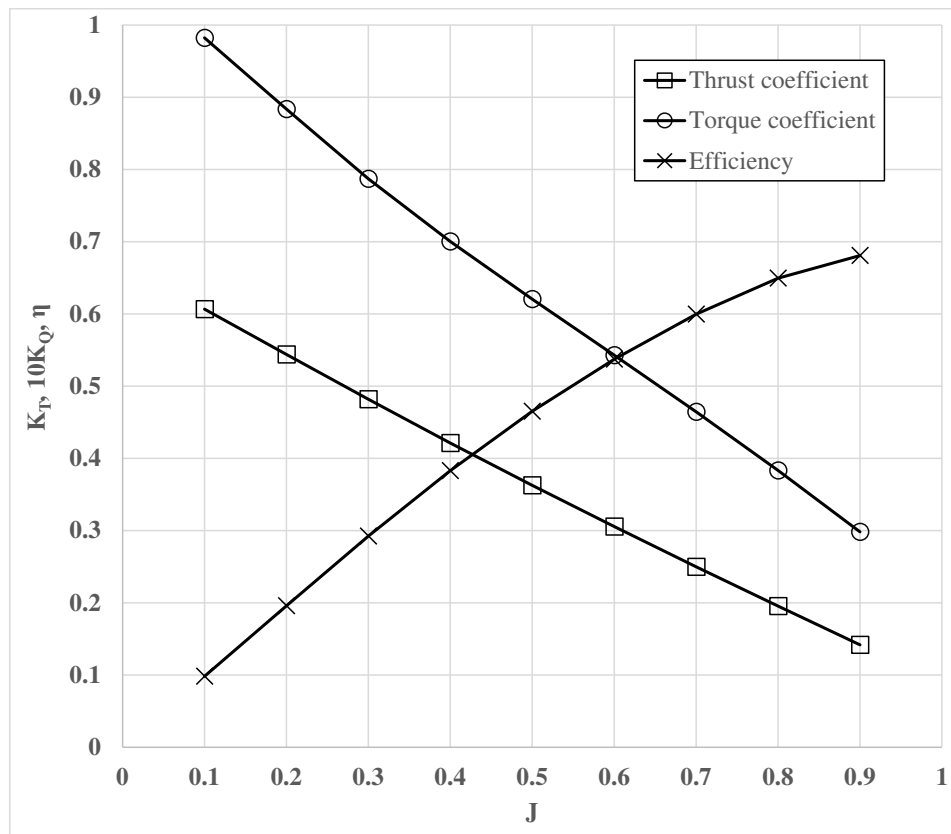
Figure 6.3 Performance Curves of the KCS Propeller with New Blade Section Profile.

	NACA66 with a=0.8 mean line profile	New blade section profile
J_d	0.715	0.847
K_T	0.1703	
K_Q	0.0297	0.0344
η	0.651	0.667

Table 6.8 KCS Propeller at Ship Self-Propulsion Point.

CHAPTER 7

CONCLUSIONS

The research presented in this dissertation focuses on the development of a framework for the analysis and design optimization of marine propellers. The goal is to develop an efficient synthesis tool that can be utilized for the design of propellers but also to propose and validate a methodology for the design of blade section profiles with low drag and a wide range of cavitation-free operation.

The propeller design framework is based on a hybrid lifting-line lifting-surface approach that can be used for the design of moderately-loaded propellers. One of the innovative elements of this research is the derivation of surrogate models to approximate the lifting surface correction factors using artificial neural networks and support vector regression. It is demonstrated that these new function approximation models have superior performance compared to polynomial representations and could be utilized not only for interpolation but also for extrapolation purposes.

The effect of viscosity is implemented in the design framework via the coupling of the hybrid lifting-line lifting-surface method to the open-source RANSE solver OpenFOAM and the utilization of an existing transition κ-ω SST model for the prediction of the hydrodynamic performance of the two-dimensional blade sections.

Another innovative aspect of this work is the formulation of the blade-section design task as a multi-objective problem with the goals being the design of blade sections with low drag but also with a wide range of cavitation-free operation. The blade sections are modeled using cubic B-splines and the corresponding control points are used as the design variables of the multi-objective

constrained optimization problem. This approach provides the designer with a set of solutions that correspond to trade-offs between the design objectives.

The effectiveness of the multi-objective design approach is validated through the analysis of the performance of a selected Pareto-optimal solution and subsequent comparison with the performance of a baseline profile. The observed improvement in both design objectives amply demonstrates the effectiveness of the proposed design optimization methodology.

The framework has been validated using available benchmark data for a number of ship propellers operating in open-water conditions. A case study regarding the analysis of the propeller of a high-speed container ship demonstrates that the proposed hybrid lifting-line lifting-surface approach is capable of computing the thrust and torque at the design operating point but also in a wide range of off-design conditions. The effectiveness of the selected optimal blade profile is also demonstrated as part of the case study, as it is shown that its utilization could lead to an increase in propulsion efficiency.

In the near future, the framework will be developed further to include a nonlinear approach regarding the propeller analysis method, in addition to incorporating a new cavitation prediction model in the OpenFOAM solver. The capability of modeling the interaction between the propeller and the ship hull by coupling the OpenFOAM solver with the lifting line code is currently under consideration. Further integration of the design framework components will also be explored.

REFERENCES

[1]   T. Brockett, "Minimum Pressure Envelopes for Modified NACA-66 Sections with NACA a = 0. 8 Camber and BUSHIPS Type I and Type II Sections," Technical Report 1780, DTNSRDC, 1966.

[2]   R. Eppler and Y. Shen, "Wing Sections for Hydrofoils - Part 1: Symmetrical Profiles," Journal of Ship Research, vol. 23, no. 3, pp. 209-217, 1979.

[3]   Y. Shen and R. Eppler, "Wing Sections for Hydrofoils - Part 2: Nonsymmetrical Profiles," Journal of Ship Research, vol. 25, no. 3, pp. 191-200, 1981.

[4]   R. Eppler and D. Somers, "A Computer Program for the Design and Analysis of Low-Speed Airfoils," NASA TM 80210, 1980.

[5]   G. Kuiper and S. D. Jessup, "A Propeller Design Method for Unsteady Conditions," SNAME Transactions, vol. 101, pp. 247-273, 1993.

[6]   H. Yamaguchi, H. Kato, M. Maeda, and M. Toyoda, "High Performance Foil Sections with Delayed Cavitation Inception," in Proceedings of the 3rd ASME/JSME Joint Fluids Engineering Conference, 1999.

[7]   S. Mishima, "Design of Cavitating Propeller Blades in Non-uniform Flow by Numerical Optimization," Ph.D. Thesis, Massachusetts Institute of Technology, 1996.

[8]   S. Mishima and S. A. Kinnas, "Application of a Numerical Optimization Technique to the Design of Cavitating Propellers in Non-Uniform Flow," Journal of Ship Research, vol. 41, pp. 93-107, 1997.

[9]   J.-L. Hwang, C.-Y. Hsin, Y.-H. Cheng, and S.-S. Chin, "Design of Inflow-Adapted Foil Sections by Using a Multi-Objective Optimization Method," in First International Symposium on Marine Propulsors, Trondheim, Norway, June 2009.

[10]   H. Ouyang, L. Weber, and A. Odgaard, "Design Optimization of a Two-Dimensional Hydrofoil by Applying a Genetic Algorithm," Engineering Optimization, vol. 38, pp. 529-540, 2006.

[11]   S. Kawada, "Induced Velocity by Helical Vortices," Journal of the Aeronautical Sciences, vol. 3, pp. 86-87, 1936.

[12]   H. W. Lerbs, "Moderately Loaded Propellers with a Finite Number of Blades and an Arbitrary Distribution of Circulation," SNAME Transactions, vol. 60, pp. 73-123, 1952.

[13]   P. C. Pien, "The Calculation of Marine Propellers based on Lifting Surface Theory," Journal of Ship Research, vol. 5, no. 2, pp. 1-14, 1961.

[14]   T. E. Brockett, "Lifting surface hydrodynamics for design of rotating blades, Propellers '81 Symposium, SNAME, 1981.

[15]   J. E. Kerwin and C. S. Lee, "Prediction of Steady and Unsteady Marine Propeller Performance by Numerical Lifting-surface Theory," SNAME Transactions, vol. 86, pp. 218-253, 1978.

[16]   D. A. Greeley and J. E. Kerwin, "Numerical Methods for Propeller Design and Analysis in Steady Flow," SNAME Transactions, vol. 90, 1982.

[17]   P. E. Griffin and S. A. Kinnas, "A Design Method for High-speed Propulsor Blades," ASME Journal of Fluids Engineering, vol. 120, pp. 556-562, 1998.

[18]   H. S. Lee and S. A. Kinnas, "Unsteady Wake Alignment for Propellers in Nonaxisymmetric Flows," Journal of Ship Research, vol. 49, no. 3, 2005.

[19] P. Van Oossanen, "Calculation of Performance and Cavitation Characteristics of Propellers including Effects on Non-uniform Flow and Viscosity," Ph.D. Thesis, Delft University of Technology, 1974.

[20] W. B. Morgan, V. Silovic, and S. B. Denny, "Propeller Lifting-surface Corrections," SNAME Transactions, vol. 76, 1968.

[21] R. A. Cumming, W. B. Morgan, and R. Boswell, "Highly Skewed Propellers," SNAME Transactions, vol. 80, pp. 98-135, 1972.

[22] B. E. Epps and R. W. Kimball, "Unified Rotor Lifting Line Theory," Journal of Ship Research, vol. 57, no. 4, pp. 181-201, 2013.

[23] J.-K. Choi and S. A. Kinnas, "Prediction of Unsteady Effective Wake by a Euler Solver/vortex-lattice Coupled Method," Journal of Ship Research, vol. 47, no. 2, pp. 131-144, 2003.

[24] F. Stern, H.-T. Kim, V. C. Patel, and H.-C. Chen, "A Viscous-flow Approach to the Computation of Propeller-hull Interaction," Journal of Ship Research, vol. 32, no. 4, pp. 246-262, 1988.

[25] F. Stern, H.-T. Kim, V. C. Patel, and H.-C. Chen, "Computation of Viscous-flow around Propeller-shaft Configuration" Journal of Ship Research, vol. 32, no. 4, pp. 263-284, 1988.

[26] J. Carlton, Marine Propellers and Propulsion, Butterworth-Heinemann, 2012.

[27] A. K. Singhal, M. M. Athavale, H. Li, and Y. Jiang, "Mathematical Basis and Validation of the Full Cavitation Model," Transactions - American Society of Mechanical Engineers, Journal of Fluid Engineering, vol. 124, pp. 617-624, 2002.

[28] C. E. Brennan, Cavitation and Bubble Dynamics, Oxford University Press, Oxford, UK, 1995.

[29]   R. Arazgaldi, A. Hajilouy, and B. Farhanieh, "Experimental and Numerical Investigation of Marine Propeller Cavitation," Scientia Iranica Transaction B-Mechanical Engineering, vol. 16, pp. 525-533, 2009.

[30]   S. Huang, M. He, C. Wang, and X. Chang, "Simulation of Cavitating Flow around a 2-D Hydrofoil," Journal of Marine Science and Application, vol. 9, pp. 63-68, 2010.

[31]   O. Coutier-Delgosha, F. Deniset, J. A. Astolfi, and J.-B. Leroux, "Numerical Prediction of Cavitating Flow on a Two-dimensional Symmetrical Hydrofoil and Comparison to Experiments," Journal of Fluids Engineering, vol. 129, pp. 279-292, 2007.

[32]   Y. Delannoy and J. L. Kueny,, "Two Phase Flow Approach in Unsteady Cavitation Modeling," Cavitation and Multiphase Flow Forum, ASME, New York, ASME-FED, vol. 98, pp. 153-158, 1990.

[33]   J. Astolfi, P. Dorange, J. Leroux, and J. Billard, "An Experimental Investigation of Cavitation Inception and Development of Partial Sheet Cavities on Two-Dimensional Hydrofoils," in Proceedings of the Twenty-Third Symposium on Naval Hydrodynamics, 2001.

[34]   E. Amromin, "Scale Effect of Cavitation Inception on a 2D Eppler Hydrofoil," Journal of Fluids Engineering, vol. 124, pp. 186-193, 2002.

[35]   The OpenFOAM Foundation, "OpenFOAM User Guide," v. 4.0, 2016.

[36]   J. W. Eaton, "GNU Octave User Manual," v. 4.2.1, 2017.

[37]   J. W. Nicholson, "The Approximate Calculation of Bessel Functions of Imaginary Argument," Philosophical Magazine, vol. 20, 1910.

[38]   S. Kawada, "On the Induced Velocity and Characteristics of a Propeller," Journal of the Faculty of Engineering, Tokyo, Imperial University, vol. 20, 1933.

[39]   H. Glauert, The Elements of Airfoil and Airscrew Theory, Cambridge University Press, 1947.

[40]   T. Sontvedt, "Theoretical Calculations of Hydrodynamic Loading on the Marine Propeller, Det Norske Veritas, Report no. 71-64-M, 1971.

[41]   M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA Data Mining Software: An Update," ACM SIGKDD Explorations Newsletter, vol. 11, no. 1, pp. 10-18, 2009.

[42]   R. C. Eberhart and Y. Shi, Computational Intelligence: Concepts to Implementations, Morgan Kaufman Publishers, 2007.

[43]   K. Hornik, M. Stinchcombe, and H. White, "Multilayer Feedforward Networks are Universal Approximators," Neural Networks, vol. 2, pp. 359-366, 1989.

[44]   D. E. Rumelhart, G. E. Hinton, R. J. Williams, "Learning Representations by Back-propagating Errors," Nature, vol. 323, pp. 533-536, 1986.

[45]   A. J. Smola and B. Schölkopf, "A Tutorial on Support Vector Regression," Statistics and Computing, vol. 14, no. 3, pp. 199-222, 2004.

[46]   B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A Training Algorithm for Optimal Margin Classifiers." In: D. Haussler (ed) Proceedings of the Annual Conference on Computational Learning Theory, ACM Press, Pittsburgh, pp. 144-152, 1992.

[47]   S. K. Shevade, S. S. Keerthi, C. Bhattacharyya, and K. R. K. Murthy, "Improvements to the SMO Algorithm for SVM Regression," IEEE Transactions on Neural Networks, vol. 11, no. 5, pp. 1188-1193, 2000.

[48] B. Ustun, W. J. Meissen, and L. M. C. Burdens, "Facilitating the Application of Support Vector Regression by Using a Universal Pearson VII Function Based Kernel," Chemometrics and Intelligent Laboratory Systems, vol. 8, no. 1, pp. 29-40, 2006.

[49] Y. T. Shen and P. E. Dimotakis, "Viscous and Nuclei Effects on Hydrodynamic Loadings and Cavitation of a NACA 66 (mod) Foil Section," Journal of Fluids Engineering, vol. 111, pp. 306 - 316, 1989.

[50] F. Menter, M. Kuntz, and R. Langtry, "Ten Years of Industrial Experience with the SST Turbulence Model," Turbulence, Heat, and Mass Transfer, vol. 4, pp. 625-632, 2003.

[51] F. R. Menter, R. Langtry, S. Likki, Y. Suzen, P. Huang, and S. Völker, "A Correlation-based Transition Model using Local Variables—Part I: Model Formulation," Journal of Turbomachinery, vol. 128, pp. 413-422, 2006.

[52] R. B. Langtry, F. Menter, S. Likki, Y. Suzen, P. Huang, and S. Völker, "A Correlation-based Transition Model using Local Variables—Part II: Test Cases and Industrial Applications," Journal of Turbomachinery, vol. 128, pp. 423-434, 2006.

[53] R. B. Langtry and F. R. Menter, "Correlation-based Transition Modeling for Unstructured Parallelized Computational Fluid Dynamics Codes," AIAA journal, vol. 47, no. 12, pp. 2894-2906, 2009.

[54] P. R. Spalart and C. L. Rumsey, "Effective Inflow Conditions for Turbulence Models in Aerodynamic Calculations," AIAA Journal, vol. 45, no. 10, pp. 2544-2553, 2007.

[55] I. B. Celik, U. Ghia, and P. J. Roache, "Procedure for Estimation and Reporting of Uncertainty Due to Discretization in CFD Applications," Journal of Fluids Engineering, vol. 130, 2008.

[56] I. H. Abbott and A. E. Von Doenhoff, Theory of Wing Sections, Dover Publications, 1959.

[57]   D. A. Bourgoyne, J. M. Hamel, S. L. Ceccio, and D. R. Dowling, "Time-averaged Flow over a Hydrofoil at High Reynolds Number," Journal of Fluid Mechanics, vol. 496, pp. 365-404, 2003.

[58]   R. Eppler, Airfoil Design and Data, Springer Science & Business Media, 1990.

[59]   J.-A. Astolfi, P. Dorange, J.-Y. Billard, and I. C. Tomas, "An Experimental Investigation of Cavitation Inception and Development on a Two-dimensional Eppler Hydrofoil," Journal of Fluids Engineering, vol. 122, pp. 164-173, 2000.

[60]   Y. T. Shen, "Wing Sections for Hydrojoils—Part 3: Experimental Verifications," Journal of Ship Research, vol. 29, pp. 39-50, 1985.

[61]   K.-T. Fang, "The Uniform Design: Application of Number-theoretic Methods in Experimental Design," Acta Math. Appl. Sinica, vol. 3, pp. 363-372, 1980.

[62]   K.-T. Fang, "Theory, Method and Applications of the Uniform Design," International Journal of Reliability, Quality and Safety Engineering, vol. 9, pp. 305-315, 2002.

[63]   T. W. Simpson, D. K. Lin, and W. Chen, "Sampling Strategies for Computer Experiments: Design and Analysis," International Journal of Reliability and Applications, vol. 2, pp. 209-240, 2001.

[64]   M. Kotinis, "Implementing Coevolution and Parallelization in a Multi-objective Particle Swarm Optimizer," Engineering Optimization, vol. 43, no. 6,, pp. 635-656, 2011.

[65]   M. Kotinis and A. Kulkarni, "Multi-objective Shape Optimization of Transonic Airfoil Sections using Swarm Intelligence and Surrogate Models," Structural and Multidisciplinary Optimization, vol. 45, no. 5, pp. 747-758, 2012.

[66]   J. Kennedy and R. C. Eberhart, "Particle Swarm Optimization," Proceedings of the IEEE International Conference on Neural Networks, pp. 1942-1948, 1995.

[67]   A. P. Engelbrecht, Fundamentals of Computational Swarm Intelligence, Wiley, Chichester, UK, 2005.

[68]   K. Deb, Multi-Objective Optimization using Evolutionary Algorithms, Wiley, Chichester, UK, 2001.

[69]   S. B. Denny, "Cavitation and Open-water Performance Ttests of a Series of Propellers Designed by Lifting-surface Methods," Technical Report, Naval Ship Research and Development Center, 1968.

[70]   S. D. Jessup, "An Experimental Investigation of Viscous Aspects of Propeller Blade Flow," Ph.D. Thesis, The Catholic University of America, 1989.

[71]   R. J. Boswell, "Design, Cavitation Performance, and Open-water Performance of a Series of Research Skewed Propellers," DTNSRDC Report, no. 3339, 1971.

[72]   W. J. Kim, S. H. Van, and D. H. Kim, "Measurement of Flows around Modern Commercial Ship Models," Experiments in Fluids, vol. 31, pp. 567-578, 2001.

[73]   J. Fujisawa, Y. Ukon, K. Kume, and H. Takeshi, "Local Velocity Field Measurements around the KCS Model (SRI MS No. 631) in the SRI 400m Towing Tank," Ship Performance Division Report No. 00±003±02. The Ship Research Institute of Japan, Mitaka, Japan, 2000.

APPENDICES

APPENDIX A

SCRIPT FOR THE LIFTING LINE METHOD

```matlab
%Calculate induced velocities, thrust and torque for propellers using
% the Lerbs' lifting line method and lifting surface corrections
% Script developed by Miltos Kotinis and Ashish Tamhane
% last updated on 3/29/2017

clear; clc;
global PropD PropB ch Nrps J GPitch x r th BTF VS Vp VA
global PropM Foil skew phi IdealA2D ZLiftA LiftSl Rn
global rho p_v mu p_atm grav pcsbm

PropM = 'KCS';
Foil = 'NACA66a0_8'; % select blade section (check available blade sections in Blade_section.m)
LSC = 'yes'; % 'yes': use lifting-surface correction factors, 'no': do not use lifting-surface
correction factors
NW = 'yes'; % 'yes': open-water propeller test, 'no': propeller operating in ship wake
PropGeom; % read propeller geometry input

J = 0.715; % ship advance coefficient
VS = 1; % ship speed
rho = 999.; % kg/m^3, water density
mu = 1.003e-3; % N*s/m^2, water dynamic viscosity

Nrps = VS/(J*PropD); % propeller operating speed (rps)
Rn = rho*Nrps*PropD^2/mu; % Reynolds number

phi_aug = repmat(linspace(0,r-1,r)',1,r-2).*repmat(phi(2:r-1),r,1);
Blade_section; % read blade section data
ZLiftA2D = ZLiftA*ones(1,r); % 2-D zero-lift angle
LiftSl2D = LiftSl*ones(1,r); % 2-D lift-curve slope
Ia = zeros(r,r); It = zeros(r,r); Iaf = zeros(r,r); Itf = zeros(r,r);
Fr_cos = cos(phi'*linspace(0,r-1,r)); % compute cosines of even Fourier series for induction
factors
x0_x = repmat(x',1,r)./repmat(x,r,1);
ha = zeros(r-2,r); ht = zeros(r-2,r);
Bratio = zeros(r,th); H = zeros(r,r,th);
Gm = zeros(r,th);
if strcmp(NW,'yes')
    VA = VS*ones(1,r);
elseif strcmp(NW,'no')
    Vp = [0.6 0.65 0.78 0.87 1.0 1.0 1.0 0.95 0.83 0.76 0.59];
    VA = VS*Vp;
end
Ua = VA; Ut = zeros(1,r);
beta_0p = atan(Ua./(x*pi*Nrps*PropD-Ut));
gamma_0 = GPitch-ZLiftA2D;
beta_ip = gamma_0-(gamma_0-beta_0p).*(0.135+0.053./(1.093-x));
ext = 0; delta = ones(length(x),1);
while (max(delta) > 1e-10 && ext < 500)
```

```
    ext = ext + 1;
    oldbetaip = beta_ip;
    Br = (1-x(1))*(tan(beta_ip)./tan(beta_0p)-1).*(Ua./VA); Br = Br(2:r-1)';
    % Calculate of the induction factors (based on the Lerbs' method)
    betai = repmat(beta_ip',1,size(x0_x,2));
    beta_0 = repmat(beta_0p',1,size(x0_x,2));
    y0 = 1./tan(betai);
    y = (1./x0_x).*y0;
    z = sqrt(1+y.^2);
    z0 = sqrt(1+y0.^2);
    A1 = (z-z0)-0.5*log((z0-1).*(z+1)./((z0+1).*(z-1)));
    x1_m = x0_x<1; A1 = A1.*x1_m;
    A2 = -(z-z0)+0.5*log((z0-1).*(z+1)./((z0+1).*(z-1)));
    x2_m = x0_x>1; A2 = A2.*x2_m;
    B1 = (((z0.^2)./(z.^2)).^0.25).*(1./(exp(PropB*A1)-1)-
(y0.^2./(2*PropB*(1+y0.^2).^1.5)).*log(1+1./(exp(PropB*A1)-1)));
    B2 = (((z0.^2)./(z.^2)).^0.25).*(1./(exp(PropB*A2)-
1)+(y0.^2./(2*PropB*(1+y0.^2).^1.5)).*log(1+1./(exp(PropB*A2)-1)));
    Ia = tril(PropB*y.*(x0_x-1).*(1+B2),-1) + triu(-PropB*y.*(x0_x-1).*B1,1) +
diag(cos(betai(:,1)));
    It = tril(PropB*(x0_x-1).*B2,-1) + triu(-PropB*(x0_x-1).*(1+B1),1) + diag(sin(betai(:,1)));
    I_m = find(x0_x==0); Ia(I_m) = 0; It(I_m) = PropB;
    I_m = isinf(x0_x); I_m = find(I_m==1); Ia(I_m) = PropB./tan(betai(I_m)); It(I_m) = 0;
    Iaf = Fr_cos\Ia; Itf = Fr_cos\It;
    % Expansion of the induction factors into an even Fourier series - calculation of the Glauert
integrals
    ha1 = sin(phi_aug).*cumsum(Iaf(:,2:r-1).*cos(phi_aug),1); ha2 = flip(cumsum(flip(Iaf(:,2:r-
1).*sin(phi_aug)),1));
    ha(:,2:r-1) = repmat(pi./sin(phi(2:r-1)),r-2,1).*(ha1(2:r-1,:) + cos(phi_aug(2:r-
1,:)).*ha2(3:r,:));
    ha1 = (0:r-1)'.*cumsum(Iaf(:,1),1); ha2 = flip(cumsum(flip(Iaf(:,1).*(0:r-1)'),1));
    ha(:,1) = pi*(ha1(2:r-1,:) + ha2(3:r,1));
    ha1 = (0:r-1)'.*cumsum(Iaf(:,r).*cos(pi*(0:r-1)'),1); ha2 =
flip(cumsum(flip(Iaf(:,r).*cos(pi*(0:r-1)').*(0:r-1)'),1));
    ha(:,r) = (-pi*cos(pi*(1:r-2)')).*(ha1(2:r-1,1) + ha2(3:r,1));
    ht1 = sin(phi_aug).*cumsum(Itf(:,2:r-1).*cos(phi_aug),1); ht2 = flip(cumsum(flip(Itf(:,2:r-
1).*sin(phi_aug)),1));
    ht(:,2:r-1) = repmat(pi./sin(phi(2:r-1)),r-2,1).*(ht1(2:r-1,:) + cos(phi_aug(2:r-
1,:)).*ht2(3:r,:));
    ht1 = (0:r-1)'.*cumsum(Itf(:,1),1); ht2 = flip(cumsum(flip(Itf(:,1).*(0:r-1)'),1));
    ht(:,1) = pi*(ht1(2:r-1,1) + ht2(3:r,1));
    ht1 = (0:r-1)'.*cumsum(Itf(:,r).*cos(pi*(0:r-1)'),1); ht2 =
flip(cumsum(flip(Itf(:,r).*cos(pi*(0:r-1)').*(0:r-1)'),1));
    ht(:,r) = (-pi*cos(pi*(1:r-2)')).*(ht1(2:r-1,1) + ht2(3:r,1));
    % Expansion of the circulation into an odd Fourier series - calculation of the circulation
distribution
    Gm = [0; (repmat((1:r-2)',1,r-2).*(ha(:,2:r-1) + tan(betai(2:r-1,2:r-1)).*ht(:,2:r-
1)./repmat(x(2:r-1),r-2,1)))'\Br; 0];
    G = sum(repmat(Gm,1,r).*sin(repmat(phi,r,1).*repmat((0:r-1)',1,r)));
    % Calculation of the induced velocities
    wa = (1/(1-x(1)))*VA.*sum(repmat(Gm(2:r-1,1),1,r).*(ha.*repmat((1:r-2)',1,r)));
wt = (1/(1-x(1)))*VA.*sum(repmat(Gm(2:r-1,1),1,r).*(ht.*repmat((1:r-2)',1,r)));
```

```matlab
    if strcmp(LSC,'yes')
        if(skew==0)
            Ka0 = arrayfun(@Ka_no_skew,tan(beta_ip),x);
            Kt0 = arrayfun(@Kt_no_skew,tan(beta_ip),x);
            Kc0 = arrayfun(@Kc_no_skew,tan(beta_ip),x);
        else
            if(skew(end)<=tan(pi*25/180))
                Ka0 = arrayfun(@Ka_skew,tan(beta_ip),x,skew);
                Kt0 = arrayfun(@Kt_skew,tan(beta_ip),x,skew);
                Kc0 = arrayfun(@Kc_skew,tan(beta_ip),x,skew);
            else
                Ka0 = arrayfun(@Ka_high_skew,tan(beta_ip),x,skew);
                Kt0 = arrayfun(@Kt_high_skew,tan(beta_ip),x,skew);
                Kc0 = arrayfun(@Kc_high_skew,tan(beta_ip),x,skew);
            end
        end
    elseif strcmp(LSC,'no')
        Ka0 = 1; Kc0 = 1; Kt0 = 0;
    end
    IdealA3D = Ka0.*IdealA2D + Kt0.*BTF;
    ZLiftA3D = ZLiftA2D + IdealA3D - IdealA2D;
    gamma_0 = GPitch-ZLiftA3D;
    LiftSl3D = LiftSl2D./Kc0;
    beta_ip = gamma_0-(2*pi*PropD*G./(LiftSl3D.*ch.*(Ua./VA+wa./VA))).*sin(beta_ip);
    delta = abs(oldbetaip-beta_ip);
end
if strcmp(LSC,'yes')  % calculate lifting-surface corrections
    if(skew==0)
        Ka0 = arrayfun(@Ka_no_skew,tan(beta_ip),x);
        Kt0 = arrayfun(@Kt_no_skew,tan(beta_ip),x);
        Kc0 = arrayfun(@Kc_no_skew,tan(beta_ip),x);
    else
        if(skew(end)<=tan(pi*25/180))
            Ka0 = arrayfun(@Ka_skew,tan(beta_ip),x,skew);
            Kt0 = arrayfun(@Kt_skew,tan(beta_ip),x,skew);
            Kc0 = arrayfun(@Kc_skew,tan(beta_ip),x,skew);
        else
            Ka0 = arrayfun(@Ka_high_skew,tan(beta_ip),x,skew);
            Kt0 = arrayfun(@Kt_high_skew,tan(beta_ip),x,skew);
            Kc0 = arrayfun(@Kc_high_skew,tan(beta_ip),x,skew);
        end
    end
 elseif strcmp(LSC,'no')
    Ka0 = 1; Kc0 = 1; Kt0 = 0;
 end
 IdealA3D = Ka0.*IdealA2D + Kt0.*BTF;
 ZLiftA3D = ZLiftA2D + IdealA3D - IdealA2D;
 gamma_0 = GPitch-ZLiftA3D;
 LiftSl3D = LiftSl2D./Kc0;
 beta_ip = gamma_0-(2*pi*PropD*G./(LiftSl3D.*ch.*(Ua./VA+wa./VA))).*sin(beta_ip);
 Cl = LiftSl3D.*(gamma_0-beta_ip);
Cd = arrayfun(@CD,1:length(x),GPitch-beta_ip);
 Vx = sqrt((Ua+wa).^2+(x*pi*Nrps*PropD-Ut-wt).^2);
```

```matlab
dT_x = 0.5*rho*ch.*(Vx.^2).*(Cl.*cos(beta_ip)-Cd.*sin(beta_ip));
dQ_x = 0.25*rho*PropD*x.*ch.*(Vx.^2).*(Cl.*sin(beta_ip)+Cd.*cos(beta_ip));

Thrust = PropB*trapz(x,dT_x);
Torque = PropB*trapz(x,dQ_x);
Kthrust = Thrust/(rho*Nrps^2*PropD^4);
Ktorque = Torque/(rho*Nrps^2*PropD^5);
eta = Kthrust*J/(2*pi*Ktorque);
if strcmp(NW,'no')
    p_v = 1704; % N/m^2, water vapor pressure
    p_atm = 101324; % N/m^2, atmospheric pressure
    grav = 9.81; % m/s^2, acceleration of gravity
    pcsbm = 6.7; % m, propeller center position relative to the free surface
    sigma = (p_atm + rho*grav*(pcsbm-(PropD/2)*x) - p_v)./(0.5*rho*Vx.^2); % calculate the
cavitation number
    disp(sigma)
end
disp(Rn); disp(J); disp(Kthrust); disp(Ktorque); disp(eta);
```

APPENDIX B

PROPELLER PARAMETERS INPUT

```matlab
% Specify propeller geometric characteristics

global PropD PropB PropAeAo skew ch tmax;
global GPitch x BTF r th PropM phi f0_c;

if strcmp(PropM,'KCS')
    PropB = 5; PropAeAo = 0.8; PropD = 1.5;
elseif strcmp(PropM,'4381')
    PropD = 1.4; PropB = 5; PropAeAo = 0.725;
elseif strcmp(PropM,'4382')
    PropD = 1.2; PropB = 5; PropAeAo = 0.725;
elseif strcmp(PropM,'4119')
    PropD = 1; PropB = 3; PropAeAo = 0.85;
end

if strcmp(PropM,'4119')
    x = [0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 0.95 1];
    phi = acos((1+x(1)-2*x)/(1-x(1))); phi = phi.*(1-(abs(phi)<1e-6));
else
    x = [0.2 0.25 0.3 0.4 0.5 0.6 0.7 0.8 0.9 0.95 1];
    phi = acos((1+x(1)-2*x)/(1-x(1))); phi = phi.*(1-(abs(phi)<1e-6));
end
r = length(x); th = 1;

if strcmp(PropM,'KCS')
    ch = [0.2313 0.2618 0.2809 0.3138 0.3403 0.3573 0.3590 0.3376 0.2797 0.2225 0.0001];
    tmax = [0.04585 0.04071 0.03712 0.03047 0.02459 0.01947 0.01492 0.01073 0.00693 0.00528
0.00369];
    P_D = [0.8347 0.8912 0.9269 0.9783 1.0079 1.0130 0.9967 0.9566 0.9006 0.8683 0.8331];
    f0_c = [0.02845 0.02964 0.02948 0.02677 0.02201 0.01732 0.01404 0.01200 0.01044 0.01007
0.0001];
    skew = [0 2.328 4.655 9.363 13.948 18.378 22.747 27.145 31.575 33.788 36.000];
elseif strcmp(PropM,'4119')
    ch = [0.3200 0.3625 0.4048 0.4392 0.4610 0.4622 0.4347 0.3613 0.2775 0.0001];
    tmax = [0.2055 0.1553  0.1180  0.09016 0.06960 0.05418 0.04206 0.03321 0.03228 0.03160];
    P_D = [1.105 1.102 1.098 1.093 1.088 1.084 1.081 1.079 1.077 1.075];
    f0_c = [0.0143 0.0232 0.0230 0.0218 0.0207 0.0200 0.0197 0.0182 0.0163 0.0118];
    skew = zeros(length(x));
    tmax = tmax.*ch;
elseif strcmp(PropM,'4381')
    ch = [0.174 0.202 0.229 0.275 0.312 0.337 0.347 0.334 0.280 0.210 0.00001];
    tmax = [0.0434 0.0396 0.0358 0.0294 0.0240 0.0191 0.0146 0.0105 0.0067 0.0048 0.0029];
    P_D = [1.332 1.338 1.345 1.358 1.336 1.280 1.210 1.137 1.066 1.031 0.995];
    f0_c = [0.0351 0.0369 0.0368 0.0348 0.0307 0.0245 0.0191 0.0148 0.0123 0.0128 0.00001];
    skew = zeros(length(x));
elseif strcmp(PropM,'4382')
    ch = [0.174 0.202 0.229 0.275 0.312 0.337 0.347 0.334 0.280 0.210 0.00001];
    tmax = [0.0434 0.0396 0.0358 0.0294 0.0240 0.0191 0.0146 0.0105 0.0067 0.0048 0.0029];
```

```
    P_D = [1.455 1.444 1.433 1.412 1.361 1.285 1.200 1.112 1.027 0.985 0.942];
    skew = [0 2.328 4.655 9.363 13.948 18.378 22.747 27.145 31.575 33.788 36.000];
    f0_c = [0.0430 0.0395 0.0370 0.0344 0.0305 0.0247 0.0199 0.0161 0.0134 0.0140 0.00001];
end
ch = ch*PropD;
tmax = PropD*tmax;
skew = tan(pi.*skew./180);
GPitch = atan(P_D./(pi*x)); % in radians
BTF = (tmax./PropD-0.003)./(1-x) + 0.003; % Calculate the blade thickness fraction for the
lifting-surface corrections
BTF(BTF==-Inf) = 0; BTF(BTF==Inf) = 0;
BTF = ones(th,1)*BTF;
```

APPENDIX C

AIRFOIL PARAMETERS INPUT

```matlab
function [] = Blade_section

% Calculate the zero lift angle, the lift-curve slope,
% and the ideal angle of attack

global Foil IdealA ZLiftA LiftSl IdealA2D f0_c;

if strcmp(Foil,'NACA66a0_8')
    angle = [-3 -2 -1 0 1 2 3 4 5 6];
    Cl = [-0.0647766 0.0417483 0.137447 0.241413 0.34893 0.450893 0.555188 0.657501 0.763605
0.859583];
    pCl = pchip(Cl,angle);
    ZLiftA = ppval(pCl,0)*pi/180; % zero lift angle
    pSl = pchip(angle,Cl);
    LiftSl = (ppval(pSl,2)-ppval(pSl,1))*180/pi; % lift-curve slope
    foil_coord = 'NACA66a0_8.dat';
elseif strcmp(Foil,'E817')
    angle = [-5 -4 -3 -2 -1 0 1 2 3 4 5 6];
    Cl = [-0.108932 0.0049831 0.119575 0.236489 0.345426 0.458736 0.571127 0.681849 0.786122
0.895762 0.989076 1.093120];
    pCl = pchip(Cl,angle);
    ZLiftA = ppval(pCl,0)*pi/180;
    pSl = pchip(angle,Cl);
    LiftSl = (ppval(pSl,2)-ppval(pSl,1))*180/pi;
    foil_coord = 'E817.dat';
elseif strcmp(Foil,'YS920')
    angle = [-4 -3 -2 -1 0 1 2 3 4 5 6];
    Cl = [-0.160605 -0.0530828 0.0572873 0.1655529 0.2733595 0.3772970 0.4890894 0.595214
0.700854 0.801731 0.902014];
    pCl = pchip(Cl,angle);
    ZLiftA = ppval(pCl,0)*pi/180;
    pSl = pchip(angle,Cl);
    LiftSl = (ppval(pSl,3)-ppval(pSl,2))*180/pi;
    foil_coord = 'YS920.dat';
elseif strcmp(Foil,'Optimal')
    angle = [-3.5 -3 -2 -1 0 1 2 3 4 5 6];
    Cl = [-0.006518 0.049626 0.161169 0.271703 0.382361 0.494020 0.602857 0.712856 0.816406
0.921012 1.02655];
    pCl = pchip(Cl,angle);
    ZLiftA = ppval(pCl,0)*pi/180;
    pSl = pchip(angle,Cl);
    LiftSl = (ppval(pSl,2)-ppval(pSl,1))*180/pi;
    foil_coord = 'Optimal.dat';
end
coord = dlmread(foil_coord);
a = size(coord,1);
a = round(a/2);
A(:,1) = coord(1:a,1);
```

```matlab
A(:,2) = coord(1:a,2);
A(:,3) = flip(coord(a:end,2));
A(:,4) = (A(:,2) + A(:,3))./2;
A = flip(A,1);
x_id = linspace(0.0001,0.9999,1500);
y_c_h = pchip(A(:,1),A(:,4));
y_id = ppval(y_c_h,x_id);
f_c = max(y_id);
f3f = (1-2*x_id)./(2*pi*(x_id.*(1-x_id)).^(1.5));
y_id = y_id.*f3f;
IdealA = trapz(x_id,y_id);
IdealA2D = (IdealA/f_c)*f0_c; % ideal angle of attack
end
```

APPENDIX D

FUNCTIONS FOR CALCULATING K_c

```matlab
function [Kc_ns] = Kc_no_skew(lambda,xp)

% Calculate the camber correction factor for propellers with zero skew
global PropB PropAeAo;
inp = [PropB pi*xp*lambda PropAeAo xp]; % Z, pi*lambda, Ae/Ao, r
min_io = [4 0.4 0.35 0.3 0.888]; % Z, pi*lambda, Ae/Ao, r, Kc
max_io = [6 2.0 1.15 0.9 3.469];
% input - hidden layer 1 weights - Sigmoid nodes 1 - 11
inp_h1_w = [-2.75089    1.38E-05       -2.38078       1.29877        -0.67293;
        -9.21712        0.07400        0.10600        1.32341        -8.77407;
        -2.75609        -0.16949       0.29642        2.30102        -2.56402;
        -0.88719        0.20515        -0.04049       0.82357        -0.21766;
        -0.48911        0.48015        -0.03110       1.60261        -1.52023;
        -2.26040        -0.43267       0.55785        1.08771        0.05458;
        -8.10760        -0.19176       -0.36607       1.56113        6.57974;
        -2.76684        -0.51862       -0.04346       1.50977        1.99701;
        -7.55100        -0.08488       -4.46464       0.21148        -1.72298;
        -1.44129        -0.50309       1.92424        0.03409        -3.19089;
        -4.36605        -1.05028       -0.68677       0.93710        1.19048];
% hidden layer 1 - hidden layer 2 weights - Sigmoid nodes 12 - 15
h1_h2_w = [-2.094 1.688 -0.880 -0.718 -1.087 -0.702  -0.823 -3.799 -0.459 -0.725 0.905   0.040;
        -3.443 3.764   -1.488 -2.323 -0.044 -0.249 -1.207 -3.256 1.150   -0.215 -0.504 0.560;
        1.889   0.159   -1.907 -0.987 -0.220 1.519   -1.828 -1.687 -0.673 -2.582 0.658   -2.600;
        -0.584  -0.037 -0.691 -0.062 -2.352 0.649   -1.865 -3.058 -2.296 -0.396 -0.635 -
0.706];
% hidden layer 2 - output layer weights - Linear node 0
h2_out_w = [1.74284    -1.580634       -1.70493       -2.058410       -2.234610];
% normalize input and add bias node input
nrm_inp = ([1 (2*inp-max_io(1:end-1)-min_io(1:end-1))./(max_io(1:end-1)-min_io(1:end-
1))]'*ones(1,size(inp_h1_w,1)))';
% find network output
h1 = ([1 (1./(1+exp(-sum(inp_h1_w.*nrm_inp,2))))']'*ones(1,size(h1_h2_w,1)))';
h2 = [1 (1./(1+exp(-sum(h1_h2_w.*h1,2))))'];
Kc_ns = ((h2_out_w*h2')*(max_io(end)-min_io(end))+min_io(end)+max_io(end))/2;
end
```

```matlab
function [Kc_s] = Kc_skew(lambda, xp, skew1)

% Calculate the camber correction factor for moderately-skewed propellers
global PropB PropAeAo;
inp = [PropB pi*lambda skew1 PropAeAo xp]; % Z, pi*lambda, skew, Ae/Ao, r
min_io = [4    0.4    0.00188 0.35 0.3 0.888]; % Z, pi*lambda, skew, Ae/Ao, r, Kc
max_io = [6    2    0.30596 1.15 0.9 3.469];
% input - hidden layer 1 weights - Sigmoid nodes 1 - 10
inp_h1_w =[-7.3708    -0.0541    -0.9372    0.3215    1.3895    -5.9625;
           -3.9309    -1.4767    -0.1731    0.0823    2.2177    0.6842;
           -3.4057    -0.0687    0.7386    -0.2199    3.2336    -0.4574;
           -1.7213    0.2475    -0.1269    0.9351    4.9080    -0.6952;
           -3.7551    -0.0339    -4.0335    0.0751    0.2746    -3.7777;
           -3.7886    0.1834    -2.2141    -0.0719    1.8391    -0.0810;
           -4.8859    -0.6965    -0.3294    -0.1960    2.7482    2.9662;
           -1.8442    -0.1833    0.0511    -0.1917    0.7315    1.1107;
           -7.2947    -0.2509    -0.2369    0.1331    1.5791    5.4331;
           -0.7391    0.0207    0.2562    1.7192    0.5096    -1.3462;
% hidden layer 1 - hidden layer 2 weights - Sigmoid nodes 11 - 13
h1_h2_w = [-1.1176    -2.6608    -1.5085    -0.8632    -0.1613    -1.3015
1.2945    -0.9657    -1.6414    -4.3167    -0.9436;
           -2.7976    -2.6923    -2.2971    -1.8130    -1.1073    1.2447
2.0496    5.1877    -1.3111    -3.1326    1.0880;
           2.0183    -5.9550    -0.4939    -1.0714    2.0459    2.9006
-1.2865 -1.6151    -1.0650    -2.0069    -1.0498];
% hidden layer 2 - output layer weights - Linear node 0
h2_out_w = [2.1357    -3.7576    -2.4639    -2.3800];
% normalize input and add bias node input
nrm_inp = ([1 (2*inp-max_io(1:end-1)-min_io(1:end-1))./(max_io(1:end-1)-min_io(1:end-1))]'*ones(1,size(inp_h1_w,1)))';
% find network output
h1 = ([1 (1./(1+exp(-sum(inp_h1_w.*nrm_inp,2))))']'*ones(1,size(h1_h2_w,1)))';
h2 = [1 (1./(1+exp(-sum(h1_h2_w.*h1,2))))'];
```

```matlab
function [Kc_hs] = Kc_high_skew(lambda, xp, skew1)

% Calculate the camber correction factor for highly-skewed propellers
global PropB
inp = [PropB pi*xp*lambda skew1 xp]; % Z, pi*lambda, skew, r
min_io = [4 0.8 0.000 0.3 1.076]; % Z, pi*lambda, skew, r, Kc
max_io = [6 1.2 1.374 0.9 2.998];
% input - hidden layer 1 weights - Sigmoid nodes 1 - 10
inp_h1_w = [ 4.322     0.3838        0.1161        8.7919        -1.8055;
            -0.845    -3.5389        0.1065        1.9652        -0.0527;
             0.5292   -1.3396        0.1398        1.2689        -0.4630;
            -2.1702   -1.8448       -0.2647       -0.7886         1.4392;
             4.8184   -0.3081        0.1945       -0.2441         6.8379;
            -4.1490   -0.2636       -0.0796       -0.8972         4.6860;
            -0.6598   -0.7109        1.9875       -0.9525         1.8857;
            -1.5384    3.0282       -0.5272        3.3387        -1.3810;
            -1.2413   -0.5073        0.3175        5.3477        -1.8784;
            -1.9857    1.3565       -0.7461        1.7310         1.2331];
% hidden layer 1 - hidden layer 2 weights - Sigmoid nodes 11 - 14
h1_h2_w = [-0.0836    0.3505    0.9113       -1.4599        0.5166        -4.3023
            2.8366    1.1484   -1.6371        1.1880       -0.7610;
           -0.5590    0.5636    0.9037       -1.8626        1.1313        -5.2025
            4.0347   -0.9342    0.0600        0.3668        2.0028;
           -1.2163    3.6231   -0.0338       -1.3159       -0.6340        -3.4351
            1.9714   -0.4032   -1.2857        1.3890       -0.3794;
           -2.0623    1.9346   -0.5861        0.3367        2.9570        -0.5876
            2.5335    1.4386   -1.3082        2.1111        2.5049];
% hidden layer 2 - hidden layer 3 weights - Sigmoid nodes 15 - 16
h2_h3_w = [4.8287    -3.0077       -4.0986       -3.6240       -2.7791;
          -1.5304    -2.8085       -1.8913       -1.9713       -3.3977];
% hidden layer 3 - output layer weights - Linear node 0
h3_out_w = [2.9634    -3.6963       -2.2195];
% normalize input and add bias node input
nrm_inp = ([1 (2*inp-max_io(1:end-1)-min_io(1:end-1))./(max_io(1:end-1)-min_io(1:end-1))]'*ones(1,size(inp_h1_w,1)))';
% find network output
h1 = ([1 (1./(1+exp(-sum(inp_h1_w.*nrm_inp,2))))']'*ones(1,size(h1_h2_w,1)))';
h2 = ([1 (1./(1+exp(-sum(h1_h2_w.*h1,2))))']'*ones(1,size(h2_h3_w,1)))';
h3 = [1 (1./(1+exp(-sum(h2_h3_w.*h2,2))))'];
Kc_hs = ((h3_out_w*h3')*(max_io(end)-min_io(end))+min_io(end)+max_io(end))/2;
if(Kc_hs > 2.188)
    Kc_hs = 2.818;
end

end
```

APPENDIX E

FUNCTIONS FOR CALCULATING K_α

```
function [Ka_ns] = Ka_no_skew(lambda,xp)

% Calculate the ideal angle of attack correction factor for propellers with zero skew
global PropB PropAeAo;
inp = [PropB pi*xp*lambda PropAeAo xp]; % Z, pi*lambda, Ae/Ao, r
min_io = [4 0.4 0.35 0.3 1.016]; % Z, pi*lambda, Ae/Ao, r, Ka
max_io = [6 2.0 1.15 0.9 3.500];
% input - hidden layer 1 weights - Sigmoid nodes 1 - 11
inp_h1_w = [-3.2396          0.1588        -0.4433         1.6336        -2.6346;
            -2.4662          0.0890         0.4858        -1.6518         0.8280;
            -4.7411         -0.1188        -4.2562        -0.1537         0.6396;
            -5.1159         -0.1815        -2.5170         0.3574        -2.4205;
             0.2349         -0.0696         1.8986         0.5743        -1.2201;
            -3.0515         -0.7387         0.2331         1.7308         2.2967;
            -4.7952         -1.5913        -0.1770         1.8101         0.8190;
            -0.4842          0.5245        -0.1562        -0.8480        -1.6385;
            -7.7872         -0.0056        -0.5569         0.4775         6.4588;
            -1.9507         -0.7595        -1.4147         1.7424         0.4068;
            -1.5883          0.0443        -0.7193        -1.9871         2.6478];
% hidden layer 1 - hidden layer 2 weights - Sigmoid nodes 12 - 14
h1_h2_w = [-0.9977    1.4166   -3.5937   -4.3793    2.4574    0.5075    1.4305
          1.8694    -1.6091    1.1231    0.0331   -0.3294;
           -4.0839    1.6104   -2.0474   -0.4516    2.5765    1.3357    0.9096
          1.7009    -0.1189    2.9098    2.2796    2.5415;
           -6.0861    0.5262   -0.8460    1.4517    4.5334    2.4112    1.3192
          1.8723    -0.4456    5.1123    0.1393    1.1120];
% hidden layer 2 - hidden layer 3 weights - Sigmoid nodes 15 - 16
h2_h3_w = [1.2724     -1.0506       -0.8209      -5.0199;
          -0.8071     -4.7283       -4.0975      -2.2539];
% hidden layer 3 - output layer weights - Linear node 0
h3_out_w = [1.3416     -2.1962       -2.6900];
% normalize input and add bias node input
nrm_inp = ([1 (2*inp-max_io(1:end-1)-min_io(1:end-1))./(max_io(1:end-1)-min_io(1:end-
1))]'*ones(1,size(inp_h1_w,1)))';
% find network output
h1 = ([1 (1./(1+exp(-sum(inp_h1_w.*nrm_inp,2))))']'*ones(1,size(h1_h2_w,1)))';
h2 = ([1 (1./(1+exp(-sum(h1_h2_w.*h1,2))))']'*ones(1,size(h2_h3_w,1)))';
h3 = [1 (1./(1+exp(-sum(h2_h3_w.*h2,2))))'];
Ka_ns = ((h3_out_w*h3')*(max_io(end)-min_io(end))+min_io(end)+max_io(end))/2;

end
```

```matlab
function [Ka_s] = Ka_skew(lambda,xp,skew1)

% Calculate the ideal angle of attack correction factor for moderately-skewed propellers
global PropB PropAeAo;
inp = [PropB pi*lambda skew1 PropAeAo xp]; % Z, pi*lambda, skew, Ae/Ao, r
min_io = [4 0.4 0.00188 0.35 0.3 -1.098]; % Z, pi*lambda, skew, Ae/Ao, r, Ka
max_io = [6 2  0.30596 1.15 0.9  4.467];
% input - hidden layer 1 weights - Sigmoid nodes 1 - 12
inp_h1_w = [-20.7772     0.0677      0.1784     -16.8691     -0.0413     -2.9537;
              1.8114     0.2930     -1.9146      -0.8758     -0.1207     -0.9325;
             -3.8812    -0.0762     -0.0441       2.3463      0.4428      1.6676;
             -1.5333     0.6353      0.1802       1.0250      7.1541     -1.3893;
             -7.6179    -0.4049     -1.4173      -0.3182      1.6869      4.1564;
              1.3249     0.0967     -0.3061       4.2440      0.2124     -5.2859;
              0.0991    -0.0564     -0.4045      -0.3525      0.5233     -2.3068;
             -4.1100     0.4961      0.9567      -0.1354      4.0693     -1.0667;
             -4.5075     0.1672      0.2909      -3.7670     -0.4152     -1.9120;
             -0.5577     0.0627     -0.2455       8.5951      0.4728    -10.4247;
              1.8938    -0.3303      1.9359      -0.1228      1.6938     -1.7295;
             -1.6155     0.0650     -1.8644      -0.5764      0.5146     -1.9903];
% hidden layer 1 - hidden layer 2 weights - Sigmoid nodes 13 - 15
h1_h2_w = [-1.5197    1.2509    2.0672    -2.4704    -0.9725    3.5905    1.5224    1.6054
2.3325    1.1542    -2.9494    1.1129    4.2004;
           -5.0220    -6.9846    -2.7936    -4.5682    2.4895    1.4495    1.3349    3.1620
-3.5418    -2.2157    2.6791    3.7712    -1.1699;
           -2.5199    -9.2044    0.7765    -6.4019    -1.8318    6.0157    3.1372    0.1909
2.5956    -3.0926    2.7383    0.9484    -0.7601];
% hidden layer 2 - hidden layer 3 weights - Sigmoid nodes 16 - 17
h2_h3_w = [2.6276     -1.3245      -2.9243      -2.0779;
           -0.0045    -1.8953      -6.5179      -5.7574];
% hidden layer 3 - output layer weights - Linear node 0
h3_out_w = [1.2933     -1.6327      -2.7021];
% normalize input and add bias node input
nrm_inp = ([1 (2*inp-max_io(1:end-1)-min_io(1:end-1))./(max_io(1:end-1)-min_io(1:end-
1))]'*ones(1,size(inp_h1_w,1)))';
% find network output
h1 = ([1 (1./(1+exp(-sum(inp_h1_w.*nrm_inp,2))))']'*ones(1,size(h1_h2_w,1)))';
h2 = ([1 (1./(1+exp(-sum(h1_h2_w.*h1,2))))']'*ones(1,size(h2_h3_w,1)))';
h3 = [1 (1./(1+exp(-sum(h2_h3_w.*h2,2))))'];
Ka_s = ((h3_out_w*h3')*(max_io(end)-min_io(end))+min_io(end)+max_io(end))/2;

end
```

```matlab
function [Ka_hs] = Ka_high_skew(lambda, xp, skew1)

% Calculate the ideal angle of attack correction factor for highly-skewed propellers
global PropB;
inp = [PropB pi*xp*lambda skew1 xp]; % Z, pi*lambda, skew, r
min_io = [4 0.8 0.000 0.3 -12.356]; % Z, pi*lambda, skew, r, Ka
max_io = [6 1.2 1.374 0.9  12.649];
% input - hidden layer 1 weights - Sigmoid nodes 1 - 11
inp_h1_w = [0.0879     -0.0356      1.4122      1.5302       1.2035;
           -1.6550      0.2199     -0.4810     -0.0068       1.2704;
           -1.3156     -0.5600     -0.5930      0.5198       1.6379;
            4.7972      0.0080     -0.0871     -1.9387       7.5794;
           -2.2202     -0.1771      0.0273     -5.5446       3.5249;
           -1.8495      1.0734      0.1728      2.9787       0.6287;
           -1.3140      1.3068      0.0528      4.8941       2.0539;
           -7.6463      0.1794     -0.0636      2.6686       6.2724;
           -1.3494      0.1202      0.1762     -0.9548       1.7401;
           -0.8335     -1.3672     -0.0599     -0.5705      -0.2015;
            0.6120     -0.4885      0.0382      3.7280       0.4213];
% hidden layer 1 - hidden layer 2 weights - Sigmoid nodes 12 - 15
h1_h2_w = [0.2099      -0.5287     -0.2288     -0.5184      -4.5411      -1.6695
           0.0652      -1.3357     -1.7562     -0.4505      -0.0548      -1.4139;
          -0.4459       0.9893      1.2948      1.0299       3.6351      -1.8281
           1.8858      -1.7607     -4.5737     -0.6502       0.0201      -1.9219;
          -0.8115       0.3629     -0.3938     -0.3511      -3.0162      -1.6446
          -0.0937      -1.2404     -1.8691     -0.6986       0.2564      -1.0443;
          -0.4917      -0.0282     -0.9093     -0.2069      -1.3352      -2.0544
          -0.4662      -2.0782     -1.8886     -1.3239       1.3432      -0.0369];
% hidden layer 2 - output layer weights - Linear node 0
h2_out_w = [-1.6108    2.8075    1.9625      2.0289      1.5734];
% normalize input and add bias node input
nrm_inp = ([1 (2*inp-max_io(1:end-1)-min_io(1:end-1))./(max_io(1:end-1)-min_io(1:end-
1))]'*ones(1,size(inp_h1_w,1)))';
% find network output
h1 = ([1 (1./(1+exp(-sum(inp_h1_w.*nrm_inp,2))))']'*ones(1,size(h1_h2_w,1)))';
h2 = [1 (1./(1+exp(-sum(h1_h2_w.*h1,2))))'];
Ka_hs = ((h2_out_w*h2')*(max_io(end)-min_io(end))+min_io(end)+max_io(end))/2;
if(Ka_hs > 3.909)
   Ka_hs = 3.909;
end
if(Ka_hs < 0.808)
   Ka_hs = 0.808;
end
```

APPENDIX F

FUNCTIONS FOR CALCULATING K_t

```matlab
function [Kt_ns] = Kt_no_skew(lambda,xp)
% Calculate the thickness correction factor for propellers with zero skew
global PropB PropAeAo;
inp = [PropB pi*xp*lambda PropAeAo xp]; % Z, pi*lambda, Ae/Ao, r
omega = 2.0; sigma = 1.7;
Input_training = csvread('Kt_no_skew.csv');
min_io = min(Input_training); % Z, pi*lambda, skew, Ae/Ao, r, Kt
max_io = max(Input_training);
%Normalize input
norm_inp = (inp - min_io(1:end-1))./(max_io(1:end-1) - min_io(1:end-1));
%Normalize training set
Input_training_norm = (Input_training(:,1:end-1) -ones(size(Input_training,1),1)*min_io(1:end-
1))./(ones(size(Input_training,1),1)*(max_io(1:end-1) - min_io(1:end-1)));
sum_d = sum((Input_training_norm-ones(size(Input_training,1),1)*norm_inp).^2,2);
dist = sum_d.^0.5;
krn = [(ones(size(dist,1),1)./(ones(size(dist,1),1)+(2*dist(:,1)*sqrt(2^(1/omega)-
1)/sigma).^2)).^omega; 1];
alpha = csvread('SMOPUK_Kt_coef_noskew.csv');
Kt_ns = (krn'*alpha)*(max_io(end)-min_io(end))+min_io(end);
end




function [Kt_s] = Kt_skew(lambda, xp, skew1)
% Calculate the thickness correction factor for moderately-skewed propellers
global PropB PropAeAo;
inp = [PropB pi*lambda skew1 PropAeAo xp]; % Z, pi*lambda, skew, Ae/Ao, r
omega = 8.0; sigma = 1.2;
Input_training = csvread('WEKA_skew_Kt_only.csv');
min_io = min(Input_training); % Z, pi*lambda, skew, Ae/Ao, r, Kt
max_io = max(Input_training);
%Normalize input
norm_inp = (inp - min_io(1:end-1))./(max_io(1:end-1) - min_io(1:end-1));
%Normalize training set
Input_training_norm = (Input_training(:,1:end-1) -ones(size(Input_training,1),1)*min_io(1:end-
1))./(ones(size(Input_training,1),1)*(max_io(1:end-1) - min_io(1:end-1)));
sum_d = sum((Input_training_norm-ones(size(Input_training,1),1)*norm_inp).^2,2);
dist = sum_d.^0.5;
krn = [(ones(size(dist,1),1)./(ones(size(dist,1),1)+(2*dist(:,1)*sqrt(2^(1/omega)-
1)/sigma).^2)).^omega; 1];
alpha = csvread('SMOPUK_Kt_coef_skew.csv');
Kt_s = (krn'*alpha)*(max_io(end)-min_io(end))+min_io(end);
end
function [Kt_hs] = Kt_high_skew(lambda, xp, skew1)
% Calculate the thickness correction factor for highly-skewed propellers
```

```matlab
global PropB;
inp = [PropB pi*xp*lambda skew1 xp]; % Z, pi*lambda, skew, r
min_io = [4 0.8 0.000 0.3 0.04]; % Z, pi*lambda, skew, r, Kt
max_io = [6 1.2 1.374 0.9 1.08];
% input - hidden layer 1 weights - Sigmoid nodes 1 - 11
inp_h1_w = [-2.3389     0.8812     -0.1176     -0.4608     -0.6899;
            -7.2481     -0.3465    -0.3095     -5.8303     -1.2365;
            -0.5705     -0.5424    -1.1134      1.1510     -1.0798;
            -1.2669      0.5577    -0.5276      0.1765     -0.6462;
            -0.5819     -0.6033    -1.0507      1.0623     -1.0534;
            -3.3499      0.6082    -1.8140      1.7003     -2.0239;
            -0.3924     -1.1913     0.4627     -0.4776      0.6528;
            -1.0315     -1.3395    -0.2457      0.5172     -0.5596;
            -1.5218      0.5656     0.0876     -0.0387     -1.8291;
            -0.7541     -0.3925     2.4073     -2.3118     -0.8916;
            -0.5218      0.2178     0.0540      0.3864      2.2308];
% hidden layer 1 - hidden layer 2 weights - Sigmoid nodes 12 - 14
h1_h2_w = [-3.6102     1.0699     5.1424     -1.3795     0.0649     -1.3846     1.6032     -1.3742
-1.5902   -0.5663     0.8176     -1.6717;
           -1.7470     0.0803     -0.4573     0.8781     0.5534     0.8041     1.6923     -1.0556
-0.6465    1.1165     1.0939     -2.4513;
           -1.9698     0.3062     1.6848     -0.1677     0.6012     -0.3174     1.7107     -0.8219
-0.9374   -0.0767     0.7726     -1.3720];
% hidden layer 2 - hidden layer 3 weights - Sigmoid nodes 15 - 16
h2_h3_w = [-0.6711     -2.0731     -2.8482     -2.0310
            2.7668     -3.6794     -2.1632     -1.9678];
% hidden layer 3 - output layer weights - Linear node 0
h3_out_w = [2.7321     -2.1266     -3.2688];
% normalize input and add bias node input
nrm_inp = ([1 (2*inp-max_io(1:end-1)-min_io(1:end-1))./(max_io(1:end-1)-min_io(1:end-
1))]'*ones(1,size(inp_h1_w,1)))';
% find network output
h1 = ([1 (1./(1+exp(-sum(inp_h1_w.*nrm_inp,2))))']'*ones(1,size(h1_h2_w,1)))';
h2 = ([1 (1./(1+exp(-sum(h1_h2_w.*h1,2))))']'*ones(1,size(h2_h3_w,1)))';
h3 = [1 (1./(1+exp(-sum(h2_h3_w.*h2,2))))'];
Kt_hs = ((h3_out_w*h3')*(max_io(end)-min_io(end))+min_io(end)+max_io(end))/2;
end
```

APPENDIX G

FUNCTION FOR CALCULATION OF SECTION C_D

```matlab
function [Cd] = CD(xp,alpha)

% Calculate the blade section drag coefficient
global Foil tmax ch;
if strcmp(Foil,'NACA66a0_8')
    angle = [-3 -2 -1 0 1 2 3 4 5 6];
    Cdr = [0.00699129 0.00671751 0.00661256 0.00664296 0.00676150 0.00698097 0.00729192
0.00778509 0.00852383 0.00913079];
    Cd = interp1(angle,Cdr,alpha*180/pi,'pchip','extrap');
elseif strcmp(Foil,'E817')
    angle = [-5 -4 -3 -2 -1 0 1 2 3 4 5 6];
    Cdr = [0.00708446 0.00677365 0.0000655027 0.00668078 0.00669658 0.00687248 0.00717984
0.00767871 0.00815160 0.00917460 0.01007430 0.01152160];
    Cd = interp1(angle,Cdr,alpha*180/pi,'pchip','extrap');
elseif strcmp(Foil,'YS920')
    angle = [-4 -3 -2 -1 0 1 2 3 4 5 6];
    Cdr = [0.00681847 0.00657998 0.00649218 0.00615860 0.00660001 0.00644495 0.00696518
0.00745026 0.00799587 0.00886487 0.0097655];
    Cd = interp1(angle,Cdr,alpha*180/pi,'pchip','extrap');
elseif strcmp(Foil,'Optimal')
    inp = [tmax(xp)/ch(xp) alpha*180/pi];
    min_io = [0.02373 -2.1 0.005529477];
    max_io = [0.198227 6.0 0.07983381];
    % input - hidden layer 1 weights - Sigmoid nodes 1 - 3
    inp_h1_w = [6.5017        4.3216         -3.8593;
               -1.7239        1.3155          0.4409;
               -0.2528        1.3500         -0.5462];
    % hidden layer 1 - output layer weights - Linear node 0
    h1_out_w = [1.3634       -2.3262         0.3481        -0.1229];
    % normalize input and add bias node input
    nrm_inp = ([1 (2*inp-max_io(1:end-1)-min_io(1:end-1))./(max_io(1:end-1)-min_io(1:end-
1))]'*ones(1,size(inp_h1_w,1)))';
    % find network output
    h1 = [1 (1./(1+exp(-sum(inp_h1_w.*nrm_inp,2))))'];
    Cd = ((h1_out_w*h1')*(max_io(end)-min_io(end))+min_io(end)+max_io(end))/2;
end

end
```

APPENDIX H

SCRIPT FOR GENERATING OPENFOAM GRID

```
clc
clear
pkg load all

Fpts = 200; %No of points on the foil
X = linspace(0,1,Fpts);

var = [0.01299 0.04052 0.05250 0.06250 0.07620 0.05975 0.0475 0.024 0.01420 -0.01260 -0.0400   -
0.0445 -0.0454 -0.0325 -0.013 0.0036 0.007 0.00324] % Control Points

ucp = [0.000 0.000 0.100 0.200 0.300 0.500 0.700 0.800 0.900 0.950 1.000; 0.000, var(1:9),
0.000];
lcp = [0.000 0.000 0.100 0.200 0.300 0.500 0.700 0.800 0.900 0.950 1.000; 0.000, var(10:18),
0.000];

ktsu  = [0 0 0 0 1/8 2/8 3/8 4/8 5/8 6/8 7/8 1 1 1 1];
ktsl  = [0 0 0 0 1/8 2/8 3/8 4/8 5/8 6/8 7/8 1 1 1 1];

arflu = bspeval(3,ucp,ktsu,X);
arfll = bspeval(3,lcp,ktsl,X);

area = trapz(arflu(1,:),arflu(2,:))-trapz(arfll(1,:),arfll(2,:));
zbar = 0.5*(trapz(arflu(1,:),arflu(2,:).^2)-trapz(arfll(1,:),arfll(2,:).^2))/area; % neutral axis
location
ixx = (trapz(arflu(1,:),(arflu(2,:)-zbar).^3)-trapz(arfll(1,:),(arfll(2,:)-zbar).^3))/3; %
compute second moment of area about x-axis
[f_max,ind_f] = max((arflu(2,:) + arfll(2,:))/2)
arflu(1,ind_f)
[t_max,ind_t] = max((arflu(2,:) - arfll(2,:)))
arflu(1,ind_t)
c_max = max(max(arflu(2,:)-zbar),max(abs(arfll(2,:)-zbar))); % maximum distance from neutral axis
z_mod = ixx/c_max % section modulus

Xu = arflu(1,:);
Xl = arfll(1,:);
Zu = arflu(2,:);
Zl = arfll(2,:);
x_mid = 101;

plot(Xu,Zu,"b");
grid minor on;
hold on;
plot(Xl,Zl,"b");
hold off;

% Geometric parameters
c = 1;                      % Geometric chord length
alpha = 0;                  % Angle of attack (in degrees)
alpha = alpha*pi/180;
```

```
scale = 1;                              % Grid scaling factor
H = 15;                                 % Radius of 'C' segment
W = 0.5;                                % Width of 3-D grid (y-axis)
D = 50;                                 % Length of downstream segment
c_r = 2*c;                              % Inner grid dimension

% Grid parameters
N_foil = 250;                   % Number of mesh cells along the foil
F_lead = 0.6;                           % Number of cells between leading edge and mid-chord
(fraction of total cells)
N_lead = round(F_lead*N_foil);          % Number of cells between leading edge and mid-chord
N_trail = N_foil-N_lead;
N_wake = 250;                           % Number of cells in the wake (downstream of the
airfoil)
N_trans_1 = 85;
N_trans_2 = 35;                         % Number of cells in the transverse direction - outer
grid
N_width = 1;                            % Number of cells along the y-axis (typically set to 1)

% Expansion ratios
Exp_trans_1 = 30000;            % Expansion ratio in the transverse direction - inner grid
Exp_trans_2 = Exp_trans_1/10000;  % Expansion ratio in the transverse direction - outer grid
Exp_trans_3 = Exp_trans_1/1000;   % Expansion ratio in the transverse direction - inner grid
outlet edges
Exp_lead_1 = 5;                % Expansion ratio on the airfoil near the leading edge
Exp_lead_2 = 1;                % Expansion ratio near the leading edge - interface of inner and
outer grids
Exp_trail_1 = 0.2;             % Expansion ratio on the airfoil near the trailing edge
Exp_trail_2 = 5*Exp_trail_1;   % Expansion ratio near the trailing edge - interface of inner
and outer grids
Exp_wake = 600;                % Expansion ratio in the downstream direction

% Grid points
v(1,1) = -H+Xu(x_mid);   v(2,1) = 0;     % Outer grid inlet center point - point 0
v(1,2) = -c_r+Xu(x_mid); v(2,2) = 0;     % Inner grid upstream center point - point 4
v(1,3) = Xu(x_mid);      v(2,3) = c_r;   % Inner grid suction side midchord point - point 5
v(1,4) = Xu(x_mid);      v(2,4) = -c_r;  % Inner grid pressure side midchord point - point 13
v(1,5) = Xu(end);        v(2,5) = c_r;   % Inner grid suction side trailing edge point - point 6
v(1,6) = Xu(end);        v(2,6) = -c_r;  % Inner grid pressure side trailing edge point - point
14
v(1,7) = (D+c);          v(2,7) = 0;     % Inner grid outlet center point - point 11
v(1,8) = (D+c);          v(2,8) = c_r;   % Inner grid outlet upper point - point 7
v(1,9) = (D+c);          v(2,9) = -c_r;  % Inner grid outlet lower point - point 15
v(1,10) = Xu(x_mid);     v(2,10) = H;    % Outer grid slip upper midchord point - point 1
v(1,11) = Xl(x_mid);     v(2,11) = -H;   % Outer grid slip lower midchord point- point 16
v(1,12) = Xu(end); v(2,12) = H;   % Outer grid slip upper trailing edge point -  point 2
v(1,13) = Xu(end); v(2,13) = -H;   % Outer grid slip lower trailing edge point - point 17

% Rotate the airfoil points according to the specified angle of attack
upper = [cos(alpha), sin(alpha); -sin(alpha), cos(alpha)] * [Xu ; Zu];
lower = [cos(alpha), sin(alpha); -sin(alpha), cos(alpha)] * [Xl ; Zl];
Xu = upper(1,:)'; Zu = upper(2,:)'; Xl = lower(1,:)'; Zl = lower(2,:)';
```

```matlab
% Rotate the grid points according to the specified angle of attack
v = [cos(alpha), sin(alpha); -sin(alpha), cos(alpha)] * v; v = v';


v(3,1) = v(3,1)-(v(3,1)-Xu(x_mid))*(v(3,2)-c_r)/(v(3,2)-Zu(x_mid)); % interpolation at c_r
v(4,1) = v(4,1)-(v(4,1)-Xl(x_mid))*(v(4,2)+c_r)/(v(4,2)-Zl(x_mid));
v(5,1) = v(5,1)-(v(5,1)-Xu(end))*(v(5,2)-c_r)/(v(5,2)-Zu(end));
v(6,1) = v(6,1)-(v(6,1)-Xl(end))*(v(6,2)+c_r)/(v(6,2)-Zl(end));
v(10,1) = v(10,1)-(v(10,1)-v(3,1))*(v(10,2)-H)/(v(10,2)-v(3,2)); % interpolation at H
v(11,1) = v(11,1)-(v(11,1)-v(4,1))*(v(11,2)+H)/(v(11,2)-v(4,2));
v(12,1) = v(12,1)-(v(12,1)-v(5,1))*(v(12,2)-H)/(v(12,2)-v(5,2));
v(13,1) = v(13,1)-(v(13,1)-v(6,1))*(v(13,2)+H)/(v(13,2)-v(6,2));
v(8,2) = v(5,2)+(v(8,2)-v(5,2))*(v(7,1)-v(5,1))/(v(8,1)-v(5,1)); % interpolation at D
v(9,2) = v(6,2)+(v(9,2)-v(6,2))*(v(7,1)-v(6,1))/(v(9,1)-v(6,1));


vertx(1,:)  = [v(1,1),     W,       v(1,2)];
vertx(2,:)  = [v(10,1),    W,          H];
vertx(3,:)  = [v(12,1),    W,          H];
vertx(4,:)  = [v(7,1),     W,          H];
vertx(5,:)  = [v(2,1),     W,       v(2,2)];
vertx(6,:)  = [v(3,1),     W,       v(3,2)];
vertx(7,:)  = [v(5,1),     W,       v(5,2)];
vertx(8,:)  = [v(7,1),     W       v(8,2)];
vertx(9,:)  = [Xu(1),      W,        Zu(1)];
vertx(10,:) = [Xu(x_mid),  W,    Zu(x_mid)];
vertx(11,:) = [Xu(end),    W,      Zu(end)];
vertx(12,:) = [v(7,1),     W,       v(7,2)];
vertx(13,:) = [Xl(x_mid),  W,    Zl(x_mid)];
vertx(14,:) = [v(4,1),     W,       v(4,2)];
vertx(15,:) = [v(6,1),     W,       v(6,2)];
vertx(16,:) = [v(7,1),     W,       v(9,2)];
vertx(17,:) = [v(11,1),    W,         -H];
vertx(18,:) = [v(13,1),    W,         -H];
vertx(19,:) = [v(7,1),     W,         -H];


% Mirror the vertices on the negative y-axis
vertx = [vertx; vertx(:,1), -vertx(:,2), vertx(:,3)];


% Edges 8-9 and 27-28
pts1 = [Xu(2:x_mid-1), W*ones(size(Xu(2:x_mid-1))), Zu(2:x_mid-1)];
pts9 = [pts1(:,1), -pts1(:,2), pts1(:,3)];


% Edges 9-10 and 28-29
pts2 = [Xu(x_mid+1:end-1), W*ones(size(Xu(x_mid+1:end-1))), Zu(x_mid+1:end-1)];
pts10 = [pts2(:,1), -pts2(:,2), pts2(:,3)];


% Edges 8-12 and 27-31
pts3 = [Xl(2:x_mid-1), W*ones(size(Xl(2:x_mid-1))), Zl(2:x_mid-1)];
pts11 = [pts3(:,1), -pts3(:,2), pts3(:,3)];


% Edges 12-10 and 31-29
pts4 = [Xl(x_mid+1:end-1), W*ones(size(Xl(x_mid+1:end-1))), Zl(x_mid+1:end-1)];
pts12 = [pts4(:,1), -pts4(:,2), pts4(:,3)];
```

```matlab
% Edges 0-1 and 19-20
pts5 = [-H*cos(pi/4)+Xu(x_mid), W, H*sin(pi/4)];
pts13 = [pts5(:,1), -pts5(:,2), pts5(:,3)];

% Edges 0-16 and 19-35
pts6 = [-H*cos(pi/4)+Xu(x_mid), W, -H*sin(pi/4)];
pts14 = [pts6(:,1), -pts6(:,2), pts6(:,3)];

% Edges 4-5 and 23-24
pts7 = [-c_r*cos(pi/4)+Xu(x_mid), W, c_r*sin(pi/4)];
pts15 = [pts7(:,1), -pts7(:,2), pts7(:,3)];

% Edges 4-13 and 23-32
pts8 = [-c_r*cos(pi/4)+Xu(x_mid), W, -c_r*sin(pi/4)];
pts16 = [pts8(:,1), -pts8(:,2), pts8(:,3)];

% Generate 'blockMeshDict' file
foil = fopen('blockMeshDict', 'w');

% Write file
fprintf(foil, '/*--------------------------------*- C++ -*----------------------------------*\\
\n');
fprintf(foil, '| =========                 |                                                 |
\n');
fprintf(foil, '| \\\\\\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
\n');
fprintf(foil, '|  \\\\\\    /   O peration     | Version:  4.0                                   |
\n');
fprintf(foil, '|   \\\\\\  /    A nd           | Web:      www.OpenFOAM.com                      |
\n');
fprintf(foil, '|    \\\\\\/     M anipulation  |                                                 |
\n');
fprintf(foil, '\\\\*---------- ---------------------------------------------------------------*/
\n');
fprintf(foil, 'FoamFile
\n');
fprintf(foil, '{
\n');
fprintf(foil, '    version     4.0;
\n');
fprintf(foil, '    format      ascii;
\n');
fprintf(foil, '    class       dictionary;
\n');
fprintf(foil, '    location    "system";
\n');
fprintf(foil, '    object      blockMeshDict;
\n');
fprintf(foil, '}
\n');
fprintf(foil, '// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
\n');
fprintf(foil, '\n');
```

```
fprintf(foil, 'convertToMeters %f; \n', scale);
fprintf(foil, '\n');
fprintf(foil, 'vertices \n');
fprintf(foil, '( \n');
fprintf(foil, '    (%f %f %f)\n', vertx');
fprintf(foil, '); \n');
fprintf(foil, '\n');
fprintf(foil, 'blocks \n');
fprintf(foil, '( \n');

fprintf(foil, '    hex (8 9 5 4 27 28 24 23) (%i %i %i) edgeGrading (%f %f %f %f %f %f %f %f 1 1
1 1) \n', N_lead, N_trans_1, N_width, Exp_lead_1, Exp_lead_2, Exp_lead_2, Exp_lead_1,
Exp_trans_1, Exp_trans_1, Exp_trans_1, Exp_trans_1);
fprintf(foil, '    hex (9 10 6 5 28 29 25 24) (%i %i %i) edgeGrading (%f %f %f %f %f %f %f %f 1 1
1 1) \n',  N_trail, N_trans_1, N_width, Exp_trail_1, Exp_trail_2, Exp_trail_2, Exp_trail_1,
Exp_trans_1, Exp_trans_1, Exp_trans_1, Exp_trans_1);

fprintf(foil, '    hex (4 5 1 0 23 24 20 19) (%i %i %i) edgeGrading (%f 1 1 %f %f %f %f %f 1 1 1
1) \n', N_lead, N_trans_2, N_width, Exp_lead_2, Exp_lead_2, Exp_trans_2, Exp_trans_2,
Exp_trans_2, Exp_trans_2);
fprintf(foil, '    hex (5 6 2 1 24 25 21 20) (%i %i %i) edgeGrading (%f 1 1 %f %f %f %f %f 1 1 1
1) \n',  N_trail, N_trans_2, N_width, Exp_trail_2, Exp_trail_2, Exp_trans_2, Exp_trans_2,
Exp_trans_2, Exp_trans_2);

fprintf(foil, '    hex (27 31 32 23 8 12 13 4) (%i %i %i) edgeGrading (%f %f %f %f %f %f %f %f 1
1 1 1) \n', N_lead, N_trans_1, N_width, Exp_lead_1, Exp_lead_2, Exp_lead_2, Exp_lead_1,
Exp_trans_1, Exp_trans_1, Exp_trans_1, Exp_trans_1);
fprintf(foil, '    hex (31 29 33 32 12 10 14 13) (%i %i %i) edgeGrading (%f %f %f %f %f %f %f %f
1 1 1 1) \n',  N_trail, N_trans_1, N_width, Exp_trail_1, Exp_trail_2, Exp_trail_2, Exp_trail_1,
Exp_trans_1, Exp_trans_1, Exp_trans_1, Exp_trans_1);

fprintf(foil, '    hex (23 32 35 19 4 13 16 0) (%i %i %i) edgeGrading (%f 1 1 %f %f %f %f %f 1 1
1 1) \n', N_lead, N_trans_2, N_width, Exp_lead_2, Exp_lead_2, Exp_trans_2, Exp_trans_2,
Exp_trans_2, Exp_trans_2);
fprintf(foil, '    hex (32 33 36 35 13 14 17 16) (%i %i %i) edgeGrading (%f 1 1 %f %f %f %f %f 1
1 1 1) \n',  N_trail, N_trans_2, N_width, Exp_trail_2, Exp_trail_2, Exp_trans_2, Exp_trans_2,
Exp_trans_2, Exp_trans_2);

fprintf(foil, '    hex (10 11 7 6 29 30 26 25) (%i %i %i) edgeGrading (%f %f %f %f %f %f %f %f 1
1 1 1) \n', N_wake, N_trans_1, N_width, Exp_wake, Exp_wake, Exp_wake, Exp_wake, Exp_trans_1,
Exp_trans_3, Exp_trans_3, Exp_trans_1);
fprintf(foil, '    hex (6 7 3 2 25 26 22 21) (%i %i %i) simpleGrading (%f %f 1) \n', N_wake,
N_trans_2, N_width, Exp_wake, Exp_trans_2);

fprintf(foil, '    hex (29 30 34 33 10 11 15 14) (%i %i %i) edgeGrading (%f %f %f %f %f %f %f %f
1 1 1 1) \n', N_wake, N_trans_1, N_width, Exp_wake, Exp_wake, Exp_wake, Exp_wake, Exp_trans_1,
Exp_trans_3, Exp_trans_3, Exp_trans_1);
fprintf(foil, '    hex (33 34 37 36 14 15 18 17) (%i %i %i) simpleGrading (%f %f 1) \n', N_wake,
N_trans_2, N_width, Exp_wake, Exp_trans_2);

fprintf(foil, '); \n');
fprintf(foil, '\n');
fprintf(foil, 'edges \n');
```

```
fprintf(foil, '( \n');

fprintf(foil, '    spline 8 9 \n');
fprintf(foil, '        ( \n');
fprintf(foil, '            (%f %f %f) \n', pts1');
fprintf(foil, '        ) \n');

fprintf(foil, '    spline 9 10 \n');
fprintf(foil, '        ( \n');
fprintf(foil, '            (%f %f %f)\n', pts2');
fprintf(foil, '        ) \n');

fprintf(foil, '    spline 8 12 \n');
fprintf(foil, '        ( \n');
fprintf(foil, '            (%f %f %f) \n', pts3');
fprintf(foil, '        ) \n');

fprintf(foil, '    spline 12 10 \n');
fprintf(foil, '        ( \n');
fprintf(foil, '            (%f %f %f)\n', pts4');
fprintf(foil, '        ) \n');

fprintf(foil, '    spline 27 28 \n');
fprintf(foil, '        ( \n');
fprintf(foil, '            (%f %f %f) \n', pts9');
fprintf(foil, '        ) \n');

fprintf(foil, '    spline 28 29 \n');
fprintf(foil, '        ( \n');
fprintf(foil, '            (%f %f %f)\n', pts10');
fprintf(foil, '        ) \n');

fprintf(foil, '    spline 27 31 \n');
fprintf(foil, '        ( \n');
fprintf(foil, '            (%f %f %f) \n', pts11');
fprintf(foil, '        ) \n');

fprintf(foil, '    spline 31 29 \n');
fprintf(foil, '        ( \n');
fprintf(foil, '            (%f %f %f)\n', pts12');
fprintf(foil, '        ) \n');

fprintf(foil, '    arc 0 1 (%f %f %f) \n', pts5');
fprintf(foil, '    arc 0 16 (%f %f %f) \n', pts6');
fprintf(foil, '    arc 19 20 (%f %f %f) \n', pts13');
fprintf(foil, '    arc 19 35 (%f %f %f) \n', pts14');

fprintf(foil, '    arc 4 5 (%f %f %f) \n', pts7');
fprintf(foil, '    arc 4 13 (%f %f %f) \n', pts8');
fprintf(foil, '    arc 23 24 (%f %f %f) \n', pts15');
fprintf(foil, '    arc 23 32 (%f %f %f) \n', pts16');

fprintf(foil, '); \n');
```

```
fprintf(foil, '\n');
fprintf(foil, 'boundary \n');
fprintf(foil, '( \n');

fprintf(foil, '    inlet \n');
fprintf(foil, '    { \n');
fprintf(foil, '        type patch; \n');
fprintf(foil, '        faces \n');
fprintf(foil, '        ( \n');
fprintf(foil, '            (0 1 20 19) \n');
fprintf(foil, '            (0 16 35 19) \n');
fprintf(foil, '        ); \n');
fprintf(foil, '    } \n');
fprintf(foil, '\n');

fprintf(foil, '    outlet \n');
fprintf(foil, '    { \n');
fprintf(foil, '        type patch; \n');
fprintf(foil, '        faces \n');
fprintf(foil, '        ( \n');
fprintf(foil, '            (18 15 34 37) \n');
fprintf(foil, '            (15 11 30 34) \n');
fprintf(foil, '            (11 7 26 30) \n');
fprintf(foil, '            (7 3 22 26) \n');
fprintf(foil, '        ); \n');
fprintf(foil, '    } \n');
fprintf(foil, '\n');

fprintf(foil, '    topAndBottom \n');
fprintf(foil, '    { \n');
fprintf(foil, '        type patch; \n');
fprintf(foil, '        faces \n');
fprintf(foil, '        ( \n');
fprintf(foil, '            (3 2 21 22) \n');
fprintf(foil, '            (2 1 20 21) \n');
fprintf(foil, '            (18 17 36 37) \n');
fprintf(foil, '            (17 16 35 36) \n');
fprintf(foil, '        ); \n');
fprintf(foil, '    } \n');
fprintf(foil, '\n');

fprintf(foil, '    airfoil \n');
fprintf(foil, '    { \n');
fprintf(foil, '        type wall; \n');
fprintf(foil, '        faces \n');
fprintf(foil, '        ( \n');
fprintf(foil, '            (8 9 28 27) \n');
fprintf(foil, '            (9 10 29 28) \n');
fprintf(foil, '            (8 12 31 27) \n');
fprintf(foil, '            (12 10 29 31) \n');
fprintf(foil, '        ); \n');
fprintf(foil, '    } \n');
fprintf(foil, '); \n');
```

```
fprintf(foil, ' \n');
fprintf(foil, 'mergePatchPairs \n');
fprintf(foil, '( \n');
fprintf(foil, '); \n');
fprintf(foil, ' \n');
fprintf(foil, '// ********************************************************************* //
\n');

% Close file
fclose(foil)
```

APPENDIX I

COORDINATES OF THE NEW PROPELLER BLADE SECTION

x	Upper	Lower
0.00000	0.00000	0.00000
0.00024	0.00157	-0.00152
0.00094	0.00315	-0.00306
0.00208	0.00472	-0.00459
0.00364	0.00630	-0.00613
0.00559	0.00788	-0.00767
0.00791	0.00946	-0.00921
0.01058	0.01102	-0.01074
0.01357	0.01259	-0.01226
0.01687	0.01414	-0.01377
0.02045	0.01567	-0.01526
0.02429	0.01720	-0.01674
0.02836	0.01871	-0.01819
0.03264	0.02019	-0.01963
0.03711	0.02166	-0.02104
0.04175	0.02310	-0.02242
0.04654	0.02452	-0.02376
0.05144	0.02591	-0.02508
0.05644	0.02727	-0.02635
0.06152	0.02860	-0.02759
0.06665	0.02989	-0.02878
0.07181	0.03115	-0.02993
0.07698	0.03237	-0.03103
0.08213	0.03355	-0.03208
0.08724	0.03468	-0.03308
0.09229	0.03577	-0.03401
0.09727	0.03681	-0.03489
0.10217	0.03781	-0.03572
0.10699	0.03876	-0.03649
0.11175	0.03967	-0.03720
0.11643	0.04054	-0.03787
0.12105	0.04137	-0.03849
0.12562	0.04217	-0.03906
0.13012	0.04293	-0.03959
0.13457	0.04366	-0.04007
0.13897	0.04436	-0.04052
0.14332	0.04502	-0.04093
0.14762	0.04566	-0.04130
0.15189	0.04628	-0.04164
0.15611	0.04687	-0.04195
0.16030	0.04744	-0.04223

0.16446	0.04798	-0.04249
0.16859	0.04851	-0.04272
0.17270	0.04902	-0.04292
0.17678	0.04952	-0.04311
0.18085	0.05000	-0.04328
0.18490	0.05048	-0.04343
0.18893	0.05094	-0.04357
0.19296	0.05139	-0.04370
0.19698	0.05184	-0.04382
0.20101	0.05228	-0.04393
0.20503	0.05272	-0.04403
0.20906	0.05316	-0.04413
0.21310	0.05359	-0.04422
0.21717	0.05402	-0.04430
0.22126	0.05445	-0.04437
0.22539	0.05488	-0.04444
0.22956	0.05531	-0.04449
0.23377	0.05574	-0.04454
0.23804	0.05617	-0.04457
0.24237	0.05660	-0.04459
0.24677	0.05703	-0.04460
0.25124	0.05746	-0.04459
0.25579	0.05789	-0.04457
0.26042	0.05832	-0.04454
0.26515	0.05875	-0.04449
0.26997	0.05919	-0.04442
0.27490	0.05963	-0.04434
0.27995	0.06007	-0.04424
0.28511	0.06052	-0.04412
0.29040	0.06097	-0.04399
0.29582	0.06143	-0.04383
0.30137	0.06189	-0.04366
0.30708	0.06235	-0.04346
0.31293	0.06282	-0.04325
0.31894	0.06330	-0.04301
0.32511	0.06378	-0.04275
0.33143	0.06426	-0.04247
0.33790	0.06474	-0.04216
0.34450	0.06523	-0.04184
0.35125	0.06571	-0.04149
0.35811	0.06618	-0.04113
0.36510	0.06665	-0.04075
0.37220	0.06710	-0.04034
0.37941	0.06754	-0.03992
0.38672	0.06797	-0.03948
0.39412	0.06838	-0.03903

0.40161	0.06878	-0.03856
0.40918	0.06915	-0.03807
0.41683	0.06949	-0.03756
0.42455	0.06982	-0.03704
0.43232	0.07011	-0.03650
0.44016	0.07037	-0.03595
0.44804	0.07061	-0.03539
0.45596	0.07080	-0.03481
0.46392	0.07096	-0.03422
0.47191	0.07109	-0.03361
0.47992	0.07117	-0.03299
0.48794	0.07120	-0.03236
0.49598	0.07120	-0.03172
0.50402	0.07114	-0.03107
0.51206	0.07104	-0.03041
0.52008	0.07089	-0.02974
0.52809	0.07070	-0.02906
0.53608	0.07047	-0.02837
0.54404	0.07020	-0.02767
0.55196	0.06989	-0.02697
0.55984	0.06955	-0.02626
0.56768	0.06917	-0.02554
0.57545	0.06877	-0.02482
0.58317	0.06834	-0.02409
0.59082	0.06788	-0.02336
0.59839	0.06740	-0.02263
0.60588	0.06690	-0.02189
0.61328	0.06639	-0.02115
0.62059	0.06585	-0.02041
0.62780	0.06530	-0.01966
0.63490	0.06474	-0.01892
0.64189	0.06417	-0.01818
0.64875	0.06359	-0.01744
0.65550	0.06301	-0.01669
0.66210	0.06242	-0.01596
0.66857	0.06183	-0.01522
0.67489	0.06125	-0.01449
0.68106	0.06067	-0.01376
0.68707	0.06009	-0.01303
0.69292	0.05952	-0.01231
0.69863	0.05896	-0.01160
0.70418	0.05840	-0.01089
0.70960	0.05784	-0.01019
0.71489	0.05728	-0.00950
0.72005	0.05673	-0.00881
0.72510	0.05617	-0.00814

0.73003	0.05562	-0.00747
0.73485	0.05506	-0.00682
0.73958	0.05450	-0.00617
0.74421	0.05394	-0.00554
0.74876	0.05337	-0.00492
0.75323	0.05280	-0.00431
0.75763	0.05222	-0.00372
0.76196	0.05163	-0.00314
0.76623	0.05103	-0.00258
0.77044	0.05042	-0.00203
0.77461	0.04980	-0.00150
0.77874	0.04917	-0.00099
0.78283	0.04853	-0.00049
0.78690	0.04787	-0.00002
0.79094	0.04720	0.00044
0.79497	0.04651	0.00088
0.79899	0.04580	0.00130
0.80301	0.04508	0.00170
0.80703	0.04434	0.00207
0.81105	0.04358	0.00243
0.81506	0.04281	0.00276
0.81907	0.04203	0.00307
0.82306	0.04123	0.00337
0.82705	0.04043	0.00364
0.83103	0.03961	0.00390
0.83499	0.03879	0.00413
0.83895	0.03796	0.00435
0.84288	0.03712	0.00454
0.84680	0.03628	0.00472
0.85070	0.03544	0.00489
0.85457	0.03459	0.00503
0.85843	0.03374	0.00515
0.86226	0.03290	0.00526
0.86606	0.03206	0.00535
0.86984	0.03122	0.00543
0.87359	0.03039	0.00548
0.87731	0.02956	0.00553
0.88100	0.02874	0.00555
0.88465	0.02793	0.00556
0.88827	0.02713	0.00556
0.89185	0.02634	0.00553
0.89539	0.02556	0.00550
0.89890	0.02480	0.00545
0.90237	0.02405	0.00538
0.90582	0.02330	0.00530
0.90927	0.02256	0.00521

0.91272	0.02181	0.00510
0.91619	0.02106	0.00498
0.91968	0.02030	0.00484
0.92321	0.01952	0.00469
0.92679	0.01872	0.00453
0.93042	0.01790	0.00435
0.93414	0.01706	0.00415
0.93793	0.01618	0.00394
0.94182	0.01527	0.00372
0.94581	0.01431	0.00349
0.94992	0.01332	0.00324
0.95417	0.01227	0.00298
0.95855	0.01117	0.00270
0.96308	0.01002	0.00241
0.96777	0.00880	0.00211
0.97264	0.00752	0.00179
0.97770	0.00617	0.00146
0.98295	0.00475	0.00111
0.98841	0.00325	0.00076
0.99409	0.00167	0.00038
1.00000	0.00000	0.00000

VITA

Name: Ashish Tamhane

Address: Department of Mechanical and Aerospace Engineering, Old Dominion University

238 Kaufman Hall, Norfolk VA 23529

Email: ashtam82@gmail.com

Education:

- Old Dominion University, Norfolk, U.S.A (August 05 – December 08)

  Master of Science in Aerospace Engineering

  Thesis Title: "Experimental Study of the Frequency Response of a Hot Film Anemometer"

- University of Mumbai, Mumbai, INDIA. (September 2000 – May 04)

  Bachelor of Engineering (Mechanical)