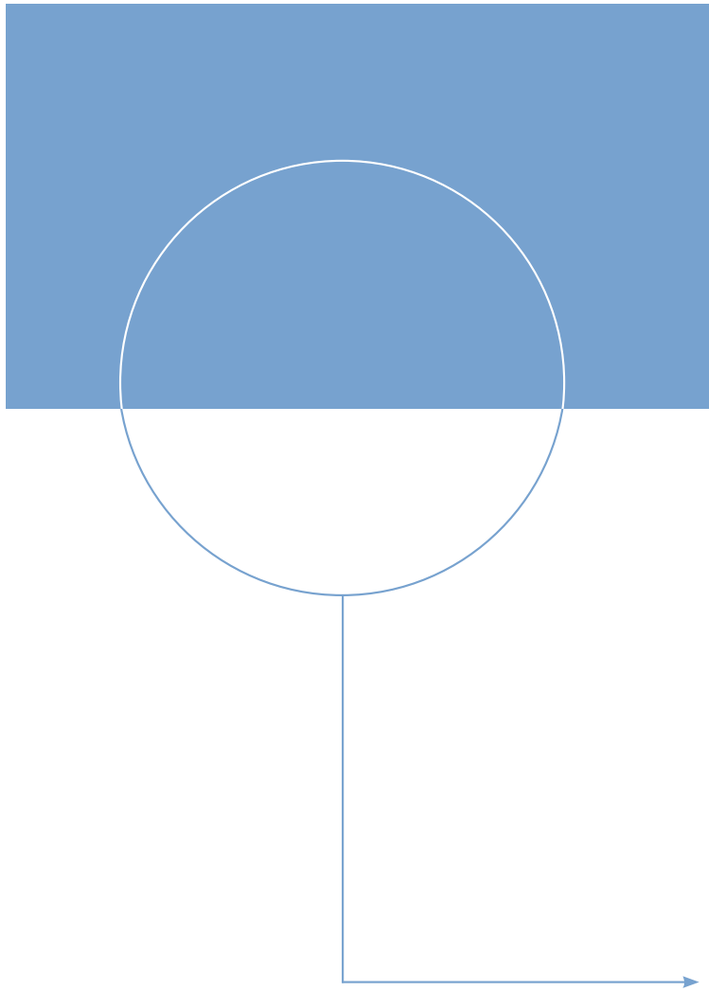

Master's thesis

2020

Ane Følgesvold Reines

NTNU
Norwegian University of
Science and Technology
Faculty of Information Technology and Electrical
Engineering
Department of Energy and Process Engineering

Ane Følgesvold Reines

Analysis and improvement of a mathematical turbine model

February 2020

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of
Science and Technology

Analysis and improvement of a mathematical turbine model

Ane Følgesvold Reines

Energy and Environmental Engineering
Submission date: February 2020
Supervisor: Bjørnar Svingen

Norwegian University of Science and Technology
Department of Energy and Process Engineering

NTNU

Norwegian University
of Science and Technology

Department of Energy
and Process Engineering

EPT-M-2019-12

MASTER THESIS

for

Student Ane Følgesvold Reines

Autumn/Spring 2019/2020

Analysis and improvement of a mathematical turbine model

Analyse og forbedring av matematisk turbinmodell

Background and objective

A mathematical turbine model is essential when analyzing and doing transient simulations of hydro power plants. A mathematical model can be simple, or it can be complex. It can be built on empirical data, or equations describing the physics (1st principles approach). Often models are a combination of empirical relations and 1st principles approach where the focus is put on one or the other. The drawback with empirical models is dependence on laboratory data for a particular turbine. Such data are often not available for new plants, and/or they are not generally available due to ownership. They are considered to be sensitive corporate data, and detailed competence is needed to use that data correct in a model. The drawback with models based on 1st principles approach is the dependence on "tuning parameters" when higher accuracy is needed/wanted. The advantage with models based on 1st principles approach is that on a generic level, a correct mathematical description is obtained independent of the degree of detailing, and therefore the model can be used by a much larger "audience".

Through the project work the candidate has intrinsic knowledge of the 1st principles approach model first developed by Nielsen, and about the special difficulties of simulating hydro power plants. In this Master thesis the candidate shall analyze the model further and look at ways to improve it. This work can for instance include, but not limited to; Characteristic values based on speed numbers of real turbines, implementing the improved model in LVTrans and/or in Simulink, compare results from simulations and measured data from real powerplants and so on.

This Master thesis has a high degree of risk with respect to the results. It is not a given that the chosen procedures and analyzes will lead to the expected results. All results, good and bad, therefore has to be considered as valuable and should be documented in the thesis.

The following tasks are to be considered:

1 Make a level headed preliminary plan based on the existing status (the project work).
2 A literature study in relevant topics with respect to the plan.
3 Execute the plan and continuously evaluate it based on current results.
4 Eventual adjustments of the plan taking into account the time frame, and continue with 2 and 3.
5 Write the Master Thesis

Acknowledgements

First of all, I would like to thank my thesis supervisor Bjørnar Svingen for his professional guidance and support throughout my final year at NTNU. He has been very accessible, and always had time and patience for questions and discussions about my work. Given the high degree of freedom in what to do and how to do it, as well as the detailing in this work, Bjørnar's deep knowledge and experience with this model has been highly useful. I am very happy that I chose his topic for my thesis, it has been educational and rewarding. I also want to acknowledge professor Torbjørn K. Nielsen and his work on developing the turbine model in the first place. It has been very interesting to work with his equations.

I would like to express my gratitude to turbine manufacturer Rainpower for providing me with the necessary experimental data. I fully understand that these are sensitive corporate data, which is why I appreciate the confidence they have given me by allowing me this access. I hope that they see value in my work.

I would like to thank graphical designer and my big brother Jonas for creating the velocity diagrams in the theoretical chapter. Further, my boyfriend Lars has been very supportive and patient during the busiest times working on my thesis. Finally, I want to acknowledge my fellow students at the Waterpower Laboratory at NTNU, who have created a great environment for learning, research and friendship. I have enjoyed working in such a competent and pleasant environment.

Ane Følgesvold Reines
Trondheim, February 2020

Abstract

In this master thesis, analysis of the performance of a mathematical hydro turbine model has been conducted. The model is based on a first principles approach, making it physically correct and applicable to any turbine. It was developed several years ago, and is based on Euler's turbine equation and the definition of turbine opening degree. It has later been modified and simplified, as well as linearized for implementation into simulation software.

Based on experimental turbine data, runner wheels were designed and models constructed. Characteristic curves were plotted using the equations in order to compare model behaviour to measured behaviour. Attention was focused on how the model captures energy loss and predicts efficiency at different operating conditions. The analysis showed relatively good agreement of efficiency as a function of runner speed, but efficiency as a function of flow rate is overpredicted. The equations struggle to capture hydraulic losses caused by irreversible flow phenomena like friction, turbulence, swirl in draft tube, etc.

With the intention to correct for this weakness, so-called "incipient efficiency", η_i, curves were fitted to measurements and implemented into the model. They were approximated as a function of flow rate, and can also be generalized with respect to turbine speed number, Ω. The speed number relates nominal flow, head and rotational speed in a single dimensionless parameter. The overall objective is to improve model accuracy without loss of generality, and for it to remain independent from empirical relations.

The results showed how the hill shape in the performance diagram is altered by η_i. In general, the behaviour is predicted well around the efficiency peak, and along the lines of constant nominal speed or opening degree. This implies that the model can be used in stability- or grid analysis. For more general transient analysis, especially for operation far away from optimal, further work to improve model accuracy was demonstrated to be necessary.

Sammendrag

I denne masteroppgaven har en matematisk modell for vannkraftturbiner blitt analysert. Modellen er basert på en "grunnprinsipp"-tilnærming, hvilket gjør den fysisk korrekt og anvendbar for enhver turbin. Den ble utviklet for flere år siden, og er basert på Eulers turbinlikning og definisjonen på turbinens åpningsgrad. Modellen har senere blitt modifisert og forenklet, så vel som linearisert for implementering i simuleringsprogramvare.

Basert på eksperimentelle turbindata har løpehjul blitt designet og modeller konstruert. Karakteristiske kurver ble plottet ved hjelp av likningene for å sammenligne simuleringer mot målinger. Det ble lagt vekt på hvordan modellen fanger opp energitap og predikerer turbinens virkningsgrad ved varierende driftsforhold. Analysen viste relativt god overensstemmelse for virkningsgrad som funksjon av rotasjonshastighet, mens virkningsgrad som funksjon av volumstrømning er overestimert. Likningene har problemer med å inkludere hydrauliske tap forårsaket av irreversible strømningsfenomener som friksjon, turbulens, virvling i sugerøret, osv.

Med hensikt å forbedre denne svakheten, ble såkalte "incipient efficiency", η_i, kurver tilpasset målinger og implementert i modellen. De ble tilnærmet som en funksjon av volumstrømning, og kan i prinsippet også generaliseres med hensyn på turbinens fartstall. Fartstallet relaterer nominell volumstrøm, fallhøyde og rotasjonshastighet i én enkelt dimensjonsløs parameter. Overordnet mål er å forbedre modellens nøyaktighet uten å gå på bekostning av generalitet, samt at den skal forbli uavhengig av empiriske sammenhenger.

Resultatene viste hvordan virkningsgradsdiagrammet endres av η_i. Generelt er ytelsen godt predikert rundt virkningsgradstoppen, samt langs kurver for konstant optimal rotasjonshastighet eller åpningsgrad. Dette innebærer at modellen kan anvendes i stabilitets- eller kraftnettanalyse. For mer generell analyse av transienter, spesielt ved drift godt utenfor optimale forhold, ble det vist at ytterligere arbeid for å forbedre modellens nøyaktighet er nødvendig.

Table of Contents

Nomenclature

Symbol	Definition	Unit
t	Time	$[s]$
x	Distance	$[m]$
g	Local gravitational constant	$[m/s^2]$
ρ	Water density	$[kg/m^3]$
f	Darcy-Weisbach friction factor	$[-]$
a	Pressure wave propagation velocity	$[m/s]$
L	Penstock pipe length	$[m]$
A	Inner pipe cross section area	$[m^2]$
D	Inner pipe diameter, or turbine runner diameter (specified)	$[m]$
r	Turbine runner radius	$[m]$
B	Turbine runner inlet height	$[m]$
Z_P	Number of pole pairs in the generator	$[-]$
a_{ij}	Linearization coefficient/ characteristic	$[-]$
s	Complex frequency, $j\omega$, variable of the Laplace domain	$[rad/s]$
Z	Hydraulic impedance, ratio of complex head to flow	$[s/m^2]$
C	Capacitance of fluid in pipeline	$[m]$
Ω	Speed number/ specific speed	$[-]$
$\dot{m}$	Mass flow rate	$[kg/s]$
Q	Volume flow rate	$[m^3/s]$
q	Per unit volume flow rate, Q/Q_R	$[-]$
H	Available piezometric (pressure) head	$[m]$
h	Per unit available head, H/H_R	$[-]$
n	Runner rotational speed	$[rpm]$ or $[rad/s]$
ω	Runner rotational speed	$[rad/s]$
$\tilde{\omega}$	Per unit runner rotational speed, ω/ω_R	$[-]$
P	Mechanical power	$[W]$
p	Per unit mechanical power, P/P_R	$[-]$
Y	Guide vane opening degree	$[m^2]$
y	Per unit guide vane opening degree, Y/Y_R	$[-]$
η	Hydraulic (also referred to as Euler) efficiency	$[-]$
$\tilde{\eta}$	Per unit hydraulic efficiency, η/η_R	$[-]$
η_i	Incipient efficiency	$[-]$
T	Mechanical torque	$[Nm]$
t	Per unit mechanical torque, T/T_R	$[-]$
T_S	Starting ($\omega = 0$) torque	$[Nm]$
t_S	Specific starting torque, $T_S/\dot{m}$	$[Nm/(kg/s)]$
m_S	Per unit specific starting torque t_S/t_R	$[-]$

Symbol	Definition	Unit
Q_{11}	Flow of turbine having unit head and unit outlet diameter	$[m^{1/2}/s]$
N_{11}	Speed of turbine having unit head and unit outlet diameter	$[m^{1/2}\ rpm]$ or $[m^{1/2}\ rad/s]$
T_{11}	Torque of turbine having unit head and unit outlet diameter	$[kg/(m^2s^2)]$
Q_{ED}	Dimensionless flow factor	$[-]$
N_{ED}	Dimensionless speed factor	$[-]$
T_{ED}	Dimensionless torque factor	$[-]$
T_{wt}	Hydraulic (water) inertia time constant for the turbine	$[s]$
T_{wp}	Hydraulic (water) inertia time constant for the penstock	$[s]$
T_a	Rotating masses (generator) inertia time constant	$[s]$
ψ	Machine constant	$[-]$
ξ	Machine constant	$[-]$
s_D	Geometrical constant	$[m^2]$
σ	Geometrical constant	$[-]$
c	Absolute water velocity	$[m/s]$
u	Runner peripheral velocity, ωr	$[m/s]$
v	Relative water velocity	$[m/s]$
α	Absolute flow direction	$[^\circ]$
α_{1R}	Best efficiency (rated/ nominal) guide vane angle	$[^\circ]$
β	Relative flow direction	$[^\circ]$
c_m	Absolute water velocity in meridional direction	$[m/s]$
c_u	Absolute water velocity in peripheral direction	$[m/s]$
v_m	Relative water velocity in meridional direction	$[m/s]$
v_u	Relative water velocity in peripheral direction	$[m/s]$
R_f	Hydraulic friction loss coefficient	$[s^2/m^5]$
R_i	Incidence loss coefficient	$[s^2/m^5]$
R_d	Draft tube loss coefficient	$[s^2/m^5]$
R_k	Minor kinetic loss coefficient	$[s^2/m^5]$
R_{df}	Disk friction loss coefficient	$[kgm^2]$
R_{mf}	Mechanical friction loss coefficient	$[kgm^2]$
p_j	Polynomial coefficients	$[-]$
a_j	Fourier series even (cosine) coefficients	$[-]$
b_j	Fourier series odd (sinus) coefficients	$[-]$
ω_0	Fundamental frequency of the Fourier series	$[rad/s]$
a, b, c, d	Exponential function coefficients	$[-]$
a, b, c	Power function coefficients	$[-]$

Abbreviations

Abbreviation	Definition
1D	One-dimensional
2D	Two-dimensional
3D	Three-dimensional
BEP	Best Efficiency Point
CFD	Computational Fluid Dynamics
CTO	Chief technical officer
e.g.	Exempli gratia (for example)
EGL	Energy Grade Line
FDM	Finite Difference Method
FEM	Finite Element Method
FSI	Fluid-structure interaction
IAHR	International Association for Hydro-Environment Engineering and Research
i.e.	Id est (that is)
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
MOC	Method of Characteristics
Nm	Newtonmeter
ODE	Ordinary Differential Equation
p.u.	Per unit (dimensionless)
rpm	Rounds per minute
VRES	Variable Renewable Energy Source

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background and motivation

The electrical power system is undergoing great changes as technological development and political action struggle to combat the greatest challenge of the world today, climate change. Implementation of variable renewable energy sources (VRES) like wind and solar is accelerating, but this involves more variability and uncertainty. Electrical production to the grid must always match total consumption, following seasonal variations and instantaneous fluctuations, in order to keep the grid frequency constant. This is necessary to maintain power system stability and reliability [1]. The need for flexible resources and energy storage capacity is growing fast. Hydro power can provide flexible generation, and pumped hydro can also provide storage when there is excess production in the system [2]. Hydro power is today a sophisticated technology, and highly prevalent in Norway given our topography. Total installed capacity in 2019 was 32 256 MW, the largest amount of installed hydro power per country in Europe [3].

To ensure optimal design and operation of new and existing hydro power plants, modelling the system is crucial. Mathematical turbine models are necessary for performing dynamic and transient analysis of systems and system components. The overall objective is the ability to predict behaviour and test scenarios without performing full-scale testing, which may not be a feasible option.

A mathematical model describes the system of interest with equations, and can be distinguished according to its basis. First principles models are based on established physical laws, without having to make any assumptions. Empirical models are based on data from experiments or observations, not necessarily supported by theory. Most hydro turbine models in practical use are a combination. Further classification according to the nature of the equations includes linear versus nonlinear models, or steady versus nonsteady models. The former involves the relationships between dependent and independent parameters, and the latter involves dependency upon time in the equations (any transient terms).

A turbine model strives to predict performance at different operating conditions as accurate as possible. This includes the efficiency, generally defined as the energy transformed to mechanical energy on the shaft by the runner, divided by the hydraulic energy available in the water [4]. For a reaction turbine like Francis, the water is a continuum from upper reservoir (or any closer open surface like a surge shaft), through the runner to the lower reservoir. Pressure transients in the system are due to changes in the flow, which is defined by the turbine and depends on available pressure head, angular speed of the runner and guide vane opening. The turbine's torque on the shaft, which defines the mechanical power output and thus the efficiency with respect to hydraulic power, depends on the same variables through the flow. These intrinsic functionalities between different variables, call for accurate mathematical models.

Hill diagrams, also known as performance or characteristic diagrams, are a common and convenient way of presenting (stationary) turbine performance for a wide operating range. In such diagrams, the abscissa represents runner speed and the ordinate represents flow rate, usually given in parameters incorporating also the available pressure head. Characteristic curves relating flow to speed at constant guide vane opening (Francis, Kaplan, etc.) or nozzle opening (Pelton) are plotted together with contour lines of constant efficiency. These contour lines will form a hill towards the point of maximum efficiency.

To accurately present a turbine by its performance diagram, requires its complete design to be known and laboratory tests to have been performed on a geometrical similar model turbine. To allow for system simulations in the early stages of a project for constructing a new plant, one must rely on diagrams from similar runners [5], or model the turbine in another way.

1.2 Previous work: project thesis

During the spring of 2019, the project thesis titled ”Modelling of hydro power plants” was written. The main objective was to model and simulate a simplified hydro power plant using turbine models of various complexity. Figure 1.1 illustrates the system of interest, in which the turbine unit was either a Francis or a Pelton.

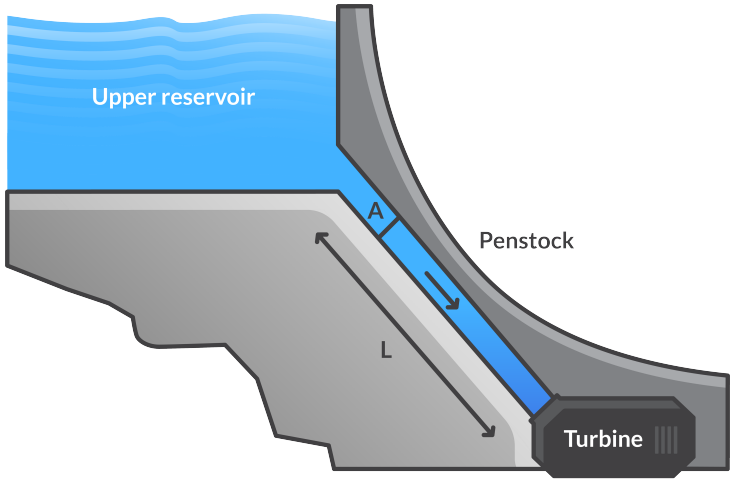

Figure 1.1: The simplified hydro power plant modelled in the project thesis, consisting of turbine unit, uniform penstock pipe and upper reservoir.

The waterways, consisting of a simple penstock connected directly to a free water surface representing the upper reservoir, was modelled by an inelastic friction-free linear model in the frequency domain. The turbine unit was modelled by two different mathematical models. The simplest was a valve model, and the more complex and emphasized one was a model developed from Eulers turbine equation and the opening degree definition. This is a steady version of the model first developed by professor Torbjørn K. Nielsen in his doctoral thesis [6], which has later been modified and linearized in works like [7] or [8]. Both models are based completely on physics and involve no empirical relations, a first principles approach. Nevertheless, measurements of a given turbine can be used for ”tuning” to increase model accuracy, making it no longer independent from experiments.

In the project work, focus was put on a thorough derivation of the equations, in order to gain understanding of the physics behind and their validity. During derivation of an expression for the total available specific energy gH, the opening degree definition was substituted in, and the general efficiency function or loss term was lost. The opening degree simply relates flow and head to guide vane/ nozzle opening, while the energy loss is a constant only correct at rated speed. Mathematically, this is the ”root” of the model weakness described in [5] [7] [8], and in the project thesis. The problem stated is that without additional loss models, its capability to predict hydraulic losses at off-design conditions is

highly inaccurate. The model simply does not comply with practical experience [5].

The model's efficiency prediction was briefly demonstrated and attempted corrected for in the project thesis. Using a simple set of efficiency data from prototype measurements, a curve was fitted to correct for the underestimate of losses by the model.

The linearized version was also emphasized due to its simple application in existing simulation software. Analytical derivation of the characteristics (linearization coefficients) was thoroughly performed for an arbitrary working point and at best efficiency point (BEP). For simulations in SIMULINK[1], characteristic values were calculated by these expressions as well as numerically. The consistency of these values were in the range of 10^{-8} to 10^{-14}, confirming the analytical expressions with the model equations.

[1] Graphical programming environment for modelling, simulating and analyzing multi-domain dynamic systems. It is based in MATLAB, a numerical programming environment and language.

1.3 Delimitation and objective of master thesis

The problem description states that the master thesis involves further *analysis* and *improvement* of the model of subject, but it does not specify *how* these parts are to be carried out. Problem delimitation, and exactly how to perform the analysis and the improvement, was completely up to the author (master student), supported by continuous guidance and conversations with supervisor Bjørnar Svingen.

The first main objective of this work is to proceed the analysis of the mathematical turbine model from the project thesis, both the original nonlinear as well as the linearized version. The nonlinear equations can produce turbine performance characteristics. The linear equations, by calculating the gradients in a certain operational point, can predict behaviour for small deviations from this working point. The focus of the model analysis will be on the nonlinear equations and their accuracy for different Francis runner designs. Special attention will be devoted to their capability to predict losses at off-design conditions, without refinements by empirical data.

The second main objective of this work is to improve the model by applying experimental data. Based on accessed measurements from Rainpower[2], improved *incipient efficiency curves*, η_i, will be proposed. Their intention is to capture off-design losses more accurately than the first proposed parabola in [5]. There are a number of fundamental energy losses occurring when hydraulic pressure head is transformed to mechanical torque on the shaft by the runner, these will be investigated from a modelling perspective in the Theory section. Even though the incipient efficiency approach does not distinguish between type or location, some basic understanding is crucial.

In stead of only tuning the model to fit the measurements for a specific turbine, the data will be used to generalize. Ideally, a large quantity of performance data from a variety of different runners should be available to do so, but such data are usually considered sensitive corporate information, which limits the access. To demonstrate the concept, generalization will be based on three model Francis turbines of diverse optimum working points. First, the improvement will be demonstrated for these runners specifically. Second, generalization will be attempted based on runner type/ estimated design point. The intention is for the model to remain independent from specific measurements after generalization. Therefore, η_i should be a function of a priori known variables.

The full block diagram implementation from the project thesis will be omitted in this work, but the linear model version is still relevant. Comparison of analytically derived characteristics against gradients from the data will be performed at BEP. Analysis of characteristic values can provide insight into the relationships of turbine variables.

[2]Turbine manufacturer, among other things.

To get a better grip of this thesis' structure and progress, the below presented flow chart in figure 1.2 can be helpful for the reader. To facilitate continuous and easy reading, discussion will not be treated in a separate chapter. The presented work will be discussed along the way, however in separate subsections clearly marked with a headline.

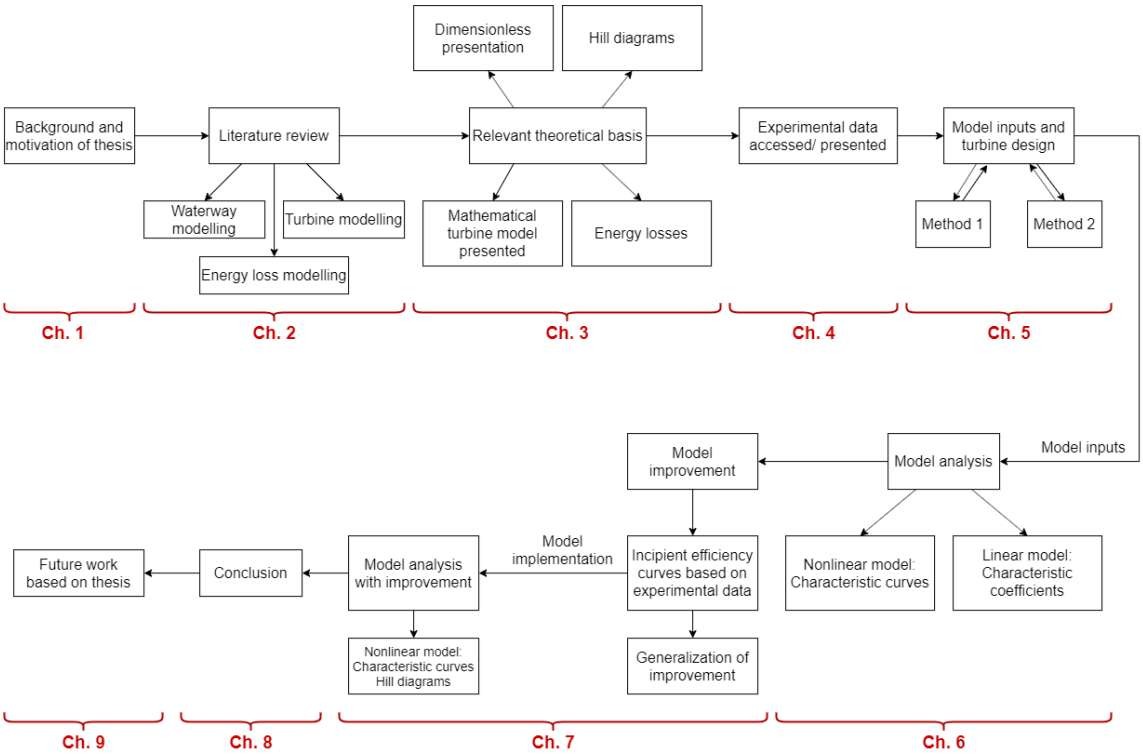

Figure 1.2: Master thesis' structure and progress including chapter numbers. The flow chart is included as an aid for the reader.

1.4 Paper for the IAHR Symposium 2020

A paper about the current development on this model has been written by B. Svingen, A. F. Reines, T. K. Nielsen and P. T. Storli. Parts of the results produced and presented in this master thesis, has also been included in this paper. The paper was submitted and accepted for presentation at the IAHR 30th Symposium on Hydraulic Machinery and Systems to be held in July 2020, Lausanne in Switzerland. The paper was submitted about one week prior to the thesis' submission, and is therefore included in Appendix A1.

Chapter 2

Literature Review

This chapter contains investigation into how hydro power plants are modelled in literature. Even though closely coupled, the literature review will distinguish between modelling of the waterway system and the hydraulic turbine. The study of turbine modelling had two main focus. First, gain some overview over general structure of mathematical turbine models. Second, gain understanding on how efficiency or energy losses are commonly modelled. Given the thesis' emphasis on performance or efficiency prediction, understanding of energy losses is important.

2.1 Modelling the waterway system

For a complete hydro power plant model, the waterways upstream and downstream the turbine must be modelled. The downstream part - for reaction turbines usually consisting of a draft tube and sometimes a discharge channel - may in some simplified models be included in the turbine model.

Briefly discussed in the project thesis, effects of water compressibility and pipe elasticity mainly depends on two factors:

1. Length of penstock, or length from turbine unit to the closest upstream open surface.

2. The intended opening or closure time of guide vanes/ nozzles, i.e. how rapid a transient is performed.

The longer penstock or the more rapid transient, the more significant is the elasticity of the upstream water column. This will influence the choice of model and its accuracy for a given system.

The starting point for most mathematical waterway models is the *Water hammer equations*; the equation of motion and the continuity equation. These differential equations were presented in the project thesis taken from [9], and are frequently presented in literature. They couple the dependent variables hydraulic head H and discharge Q, using the independent variables distance x and time t, for a fluid-filled pipe of constant cross section. The solution procedure and any simplifications to these fundamental equations are usually what sets different waterway models apart. The equations can be solved in the time domain or the frequency (Laplace) domain. A few methods frequently occurring in the literature will be presented or referenced to in the following.

2.1.1 Frequency domain solution with inelastic approximation

In the project thesis, a simplified model solving the equations in the frequency domain, was used. Water compressibility effects and pipe friction was neglected, resulting in a very simple transfer function relating per unit head h and flow q:

$$\frac{h}{q} \;=\; -\,T_{wp}\;s \tag{2.1}$$

Where

$$T_{wp} \;=\; \frac{QL}{gAH} \tag{2.2}$$

The penstock dynamics is described purely by the "hydraulic inertia time constant", also known as the "inelastic water time constant". T_{wp} is given by pipe dimensions L and A, and by flow Q and net head H at BEP. This is a "lumped mass" model assuming rigid pipe. As demonstrated in the project thesis, its implementation into block diagram software in combination with some linear turbine model, is highly convenient. Its accuracy, however, mainly depends on the two factors stated above.

2.1.2 Other frequency domain solutions

More complex solutions to the Water hammer equations in the frequency domain can include the elasticity of water and pipe. In literature like [9] ch. 12, or [10] [11], the differential equations are solved using a linear impedance method based on electrical transmission line theory. The resulting transfer functions for head and discharge at the upstream "U" and downstream "D" sections of a single pipe becomes [9]:

$$H_D \;=\; H_U \cosh\left(Z_C C L s\right) \;-\; Q_U Z_C \sinh\left(Z_C C L s\right) \tag{2.3}$$

$$Q_D \;=\; -\frac{H_U}{Z_C} \sinh\left(Z_C C L s\right) \;+\; Q_U \cosh\left(Z_C C L s\right) \tag{2.4}$$

Where Z_C is the hydraulic surge impedance, $C = \dfrac{gA}{a^2}$ is the capacitance of fluid in a pipeline, L is pipe length, A is pipe cross section and a is the propagation wave speed.

In general, hydraulic impedance Z in a fluid system is defined as the ratio of complex head to complex discharge at that particular point; $Z(x) = H(x)/Q(x)$, and is therefore independent of time [9]. These equations are presented slightly different in [10] or [11]. For instance, the head transfer function in equation 2.3 can also include a head loss term.

To study the frequency response of a fluid-filled pipe, the above transfer functions for head, discharge and hydraulic impedance, must be applied together with boundary conditions presented in a suitable manner for complex number calculations [9]. For a simple plant with a single penstock, the upstream boundary condition can for instance be the reservoir or a surge shaft having a specified pressure head given by the water level. The downstream boundary condition can be a lower reservoir water level, or a reaction turbine and its associated model. For a penstock supplying a Pelton, the downstream boundary condition can be the atmospheric pressure head at the location of the nozzle opening.

Another transfer function method is the *Structure Matrix Method* developed by Hermod Brekke in his doctoral thesis, described in detail in works like [12] [13].

There also exists Finite Element Methods (FEM) to numerically solve the governing equations in the frequency domain. An example from literature is the fluid-structure interaction (FSI) analysis performed in the doctoral thesis of supervisor Bjørnar Svingen [14].

2.1.3 Time domain solutions and Method of Characteristics

Numerical solutions to the Water hammer equations in the time domain can include different Finite Difference Methods (FDM) or Finite Element Methods (FEM). To narrow the literature study of waterway models, these were not investigated, only the Method of Characteristics (MOC) was.

The characteristics method is a numerical method for solving hyperbolic partial differential equations in time and space [6]. It is commonly used for calculating pressure transients in hydro power systems. In the software LVTrans, which uses the turbine model of investigation in this thesis, MOC is ”the brain” [15]. The solution is approximately analytically exact and the method is computationally inexpensive (fast).

The procedure is applied in a so-called ”staggered grid” in space (one-dimensional along the pipeline) and in time, a ”xt-grid”. The Water hammer equations are transformed to a set of ordinary differential equations (ODEs), which are solved analytically to obtain two algebraic equations. These equations describe the transient propagation of pressure head and flow in a pipeline, and are to be solved for H and Q at a given position and time.

Even though MOC was carefully studied during the literature review, a detailed recipe will not be included as it is somewhat outside the scope of this thesis. The method has been well documented in literature, for example [9] or [6] (Appendix A) provide thorough and understandable formulations.

2.2 Modelling the turbine

Turbine models can be used when experiential data are unavailable or limited, or when iterating in a diagram based on curve fitting of measured points [5], is too much trouble. There exists a large number of turbine models in literature and in use. They can be linear or nonlinear, steady or unsteady, simple or complex, and so on. The exact functionalities between variables, any transient terms or empirical relations, are mainly what sets different mathematical models apart. In this section, some relevant literature on turbine modelling will be reviewed.

2.2.1 Characteristic curves in dynamic modelling

The Hill chart with its characteristic curves represents the *steady* performance of a turbine. This is why there has been discussion about using it also for modelling the *dynamic* response. The error of doing so was investigated by prof. Nielsen in his doctoral thesis [6]. He proposed that the hydraulic inertia in the turbine, I, also expressed through the "turbine water" time constant T_{wt}, is the main reason for the deviation between stationary measurements and dynamic path during a transient. He proposed a redefinition of the turbine net head including the hydraulic inertia (a dynamic net head), resulting in improved consistency between stationary and dynamic measurements when corrected for.

2.2.2 Linear models

For small disturbances from an operating point, linear models based on transfer coefficients representing the gradients (linearization coefficients), can be applied. For larger disturbances, either the transfer coefficients must be recalculated within each subsection, or nonlinear models must be applied [16].

Characteristic values of linear models are usually calculated based on performance curves, and must be recalculated from the diagram for every operating point of investigation. There also exists methods for calculating them based on "internal characteristics equations" such that prototype measurements are not needed [16] [17]. In [17], both methods for the linear model are explained and exemplified by simulations. In the project thesis, mathematical expressions for the characteristic coefficients of the linearized model of investigation, were derived for an arbitrary linearization point and for best efficiency point. This is a way of calculating without measurements, but their accuracy was not investigated.

2.2.3 Nonlinear models

The work of the 1992 IEEE Working Group on Prime Mover and Energy Supply Models for System Dynamic Performance Studies [18] is frequently highlighted in literature. They presented different formulations of linear and nonlinear turbine models taking into account both inelastic and elastic water column. One of their main objectives was to present basic

physics of the hydraulic turbine and its controller. This is why their proposed models are extensively used as *base models*, both in literature and in industry, requiring refinements according to the plant configuration and the intended simulation [10]. Following the work of [18], several subsequent works have provided different formulations and improvements to their models.

In [10], the authors state the main developments since the 1992 IEEE formulation have been within plant control and governor models. The turbine energy transfer is still modelled by a linearization around a working point or some variant of the IEEE nonlinear model. [10] proposed a "new" turbine formulation taking into account real sources of major loss, eliminating the use of ambiguous correction factors. Simulations with the proposed model are compared to simulations using the 1992 IEEE inelastic waterway model [18] and the model by Kundur [19], and validated against full-scale test data. The proposed model shows the most consistency with respect to the measurements.

2.2.4 The Euler turbine equation

It is common to use the Euler turbine equation as a starting point when formulating model equations, as performed in works like [5] [6] [8] [10] [20], among many. The Euler equation is correct for *any* turbomachine and holds in its *entire* operating range, but the exact functionality for the "Euler efficiency" (or hydraulic efficiency) making it hold, is not specified. In [20], the authors presented simulations using three very simple models. All three having the Euler equation as a starting point, formulating the same torque equation but different head equations. They are independent from measurements, i.e. contain no empirical relations and require no tuning. None of them can be used directly for modelling a turbine as they are all too simple and inaccurate. However, the investigation provided insight into the actual physics and illustrated the importance of including additional loss models when using Euler's equation. Especially the importance of "incipient loss" (also called incidence or shock loss) at runner inlet was highlighted.

The simplest of the models in [20] was also used in [10], however they subtracted from the theoretical available specific energy (given by Euler) a number of loss terms based on empirical formulas. As expected, this significantly improved accuracy with respect to real turbine performance.

2.2.5 Physics-based versus empirical based

One tendency from the literature seems evident; Mathematical models based on a first principles approach are always correct as they are based purely on fundamental physical laws. However, they can become quite inaccurate when applied to a specific system, or they may require tuning, which can be challenging without detailed knowledge about the system and its behaviour.

The complexity of physically based models can vary. For example Euler theory simplifies the complex three-dimensional runner blade structure to a two-dimensional axisymmetric geometry, and applies to this control volume the law of conservation of angular momentum using the velocity components at inlet and outlet (for some streamline). In comparison, solving the Navier-Stokes equations for the three-dimensional flow field inside the turbine using computational fluid dynamics (CFD) techniques, are also a way of modelling based on first principles approach. The latter example is significantly more complex and usually *very* computational expensive (slow). Analysis involving CFD has not been studied in this Literature review as it was considered outside the thesis' scope.

It appears from the literature that most industrially applied turbine models are also based on empirical relations. This makes them dependent on experimental data for the specific unit, or on previous works on empirical relations. An example of such work is [21], which examined scale effects of a Francis turbine away from its optimum operating condition. Based on analysis of component losses of performance measurements from a number of Francis turbines, loss coefficients of an "efficiency conversion formula" were determined. [10] used the work in [21] to determine specific energy loss components.

2.2.6 Steady versus unsteady

Several of the models encountered in literature are *steady*, but the significance of including transient terms is also discussed in some references. In [11], two turbine models were compared, one using an algebraic equation for the torque and the other using a differential equation. In the latter formulation, they obtain transfer coefficients similar to the characteristics of traditional linear models. Unlike the linearization coefficients, their transfer coefficients are calculated analytically using the dynamic parameters of the turbine during transients. Based on a single step response simulation, they conclude the two models to behave very similar and thus the transient version to be unnecessary. They state the dynamic of the torque to mainly be determined by the dynamic of the hydraulic system at turbine inlet, emphasising the importance of a good model for the upstream waterways.

2.2.7 An alternative derivation method

Similar to several other models, [6] has a 1D approach using Euler's turbine equation and includes empirical correction terms for modelling the losses. The method to obtain the governing differential equations stands out by using Bond Graph Theory. In the hydraulic and mechanical systems intended to model, prof. Nielsen recognizes effort or flow sources and sinks, resistance, compliance and inertia elements. He obtains from the system bond graph the differential equations by using simple bond graph laws. The same equations excluding the empirical loss terms, are presented in his more recent work [5], which are the unsteady version of the model equations being studied in this thesis.

2.3 Modelling the energy losses

Modelling the turbine involves modelling the losses when hydraulic energy is converted to mechanical energy. The Euler equation as presented in works like [10] represents the *theoretical maximum* specific energy transfer, but in reality, a number of losses will prevent the *real* specific energy transfer from being equal to this. In [20], certain shortcomings related to capturing losses when using Euler's equation directly, were demonstrated.

To understand the physics behind different types and under what operating conditions they are present, works like [4] were useful. In [22], typical percentages for different types are presented. Further theoretical background of the main energy losses occurring in a reaction turbine is included in Theory section 3.5.

2.3.1 Scale effects

It appears essential to distinguish between scalable and non-scalable losses, as some types do not scale proportionally to size for geometrically similar turbines. Such losses and scaling formulas are discussed in works like [22] [23], and more. When using model turbine efficiency data to improve a turbine model, which may later be applied to prototype turbines, scale effects can impact the accuracy of such improvement. Nevertheless, for simplicity of the work, this topic has not been studied in more detail.

2.3.2 Modelling the Hill diagram

Since Hill charts are a convenient way to present turbine performance, to calculate the Hill diagram with a certain desired accuracy can be the objective of a turbine model. This is the case in [24], where diagrams for two low specific speed (high head) Francis turbines of equal main dimensions but different designs, were calculated by a simplified 1D approach based on runner inlet and outlet velocity diagrams. Unlike many other 1D models, the calculations in [24] are based on the shroud streamline in the meridional section, not on the mean one. Four simple loss models intended to capture inlet incidence loss, outlet residual swirl loss, runner blade friction loss and finally friction and diffuser losses in the draft tube, are included. These models are functions of vector components from the diagrams and include empirical coefficients. Surely, other losses present could have been included to increase accuracy, but these four types were considered most decisive for the general shape and inclination of the turbine Hill chart. The relatively good correspondence in diagram shape between calculations and measurements confirms so.

2.3.3 Functionalities and empirical relations

It appears from the reviewed literature that many turbine loss models include coefficients or terms based on empiricism (experiments or observations). They can be formulated as a head $[m]$ or specific energy $[J/kg]$ loss, or as a power $[W]$ loss, subtracting them from

some expression for the ideal (theoretical maximum) energy or power transfer.

Further, it appears common to divide the losses according to *type*, as performed for example in [6] [10] [11] [16] [24], and more. Alternatively or additionally, losses may be divided according to *flow domain* where they occur, i.e. location in the system, as performed for example in [10] [24]. They identify the different losses in spiral casing, stay vanes, guide vanes (or collectively in the distributor), runner and draft tube. [10] further distinguish between frictional and kinetic losses. The former are continuous and present to various degrees in all flow fields, while the latter are "singularities" like wake loss, residual swirl in the draft tube, losses due to bends, channel divergence/ convergence, etc.

Depending on type and location, the loss models often functionally depend on velocity vector components, flow rate or flow rate *deviation* from nominal flow rate, no-load flow rate or runner rotational speed. Which, indirectly, means they all depend on different velocity vector components at specific locations of a certain streamline.

In [6], losses are modelled by resistance elements in the system bond graph and from this included in the differential equations. In the hydraulic domain, there is one element for so-called Bernoulli losses and one for diffusion losses. In the mechanical domain, there is one element for all turbine losses, which is divided further into head loss, leakage loss and mechanical loss. Functionalities were determined according to physics, but also including empirical coefficients. The per unit version of these loss coefficients are tuned by a trial-and-error method based on peak efficiency operational point, starting torque and runaway speed. That is, the loss modelling in [6] is dependent on experimental data.

The dependency upon at least some empirical relations seems to be a trend for many mathematical turbine models. What distinguishes them is mainly the degree of detailing (number of loss terms) and the calibration methods for determining these. An example of such calibration procedure is given in [16].

Chapter 3

Theory

This chapter contains mathematical and physical definitions and relations relevant to the subsequent work. Basic theory on dimensionless numbers and turbine similarity is relevant for the processing of experimental data and model inputs/ outputs. Hill diagrams will be briefly explained and visualized. Following this background, the turbine model of investigation is presented, including all governing equations and input definitions. The efficiency prediction is investigated from a mathematical point of view. Entering the world of energy loss, some theoretical background on the most important types occurring in a reaction turbine, will be presented.

3.1 Dimensionless numbers

3.1.1 Reduced properties and speed number

When classifying or dimensioning hydraulic turbines, it is useful to work with reduced values and dimensionless numbers. A reduced parameter is achieved by dividing with the reference velocity defined as the maximum water velocity possible to achieve for a given system, $\sqrt{2gH}$. It represents all available energy, and can be obtained if total net head available to the turbine is transferred without losses to kinetic energy [4] [25]. Dividing all velocities, including flow rate and angular speed, by this value results in the *reduced properties*, denoted by underlined letters.

A highly relevant parameter is obtained by combining the reduced rotational speed with the reduced flow rate [4]:

$$\Omega \;=\; \underline{\omega}\,\sqrt{\underline{Q}} \;=\; \frac{\omega}{\sqrt{2gH}}\,\sqrt{\frac{Q}{\sqrt{2gH}}} \;=\; \frac{\omega\, Q^{1/2}}{(2gH)^{3/4}} \tag{3.1}$$

This dimensionless number is called the *speed number* or the *specific speed*. It implies design condition, meaning that all parameters in 3.1 are rated/ nominal values corresponding to BEP. Some references also use the definition:

$$\Omega \;=\; \frac{\omega\, Q^{1/2}}{(gH)^{3/4}} \tag{3.2}$$

Either way, Ω relates nominal speed, flow and head in a single parameter, and can be used as a reference value for classification. It can in most cases be calculated early in the design phase of a new turbine, and may assist in deciding appropriate type, size and shape [4] [25]. In this thesis, the speed number Ω will be calculated according to equation 3.1.

3.1.2 Geometrical similarity

Geometrical similar turbines have similar velocity diagrams, i.e. equal shape (angles), and equal reduced velocities, in the same operating mode. They also have equal speed number Ω, but equal speed number in itself does not guarantee similarity of the velocity diagrams. The Euler equation using reduced velocities expresses the hydraulic efficiency as [4]:

$$\eta_h \;=\; 2\,(\;\underline{c}_{u1}\;\underline{u}_1 - \underline{c}_{u2}\;\underline{u}_2\;) \tag{3.3}$$

If the reduced velocities of geometrical similar turbines are equal, this implies their efficiency to be equal, however *scaling effects* due to non-equal Reynolds numbers will make this statement inaccurate in reality. Friction losses will in general be smaller for larger turbines due to larger Reynolds numbers, shifting the efficiency slightly for geometrical similar turbines [4] [22]. The main example is scale-up of turbine efficiency from model to prototype. Scaling effects may also result in a slightly shift in BEP [22].

Regardless the phenomena of scaling effects, which can be corrected for by scaling laws, turbine similarity sure has its usefulness.

3.1.3 Unit parameters

It is appropriate to present the definitions of the *unit parameters*, which are the flow, speed and torque for a turbine having unit outlet diameter $D_2 = 1m$ and unit net available head $H = 1m$ [6] [25]:

$$Q_{11} = \frac{Q}{D_2^2\sqrt{H}} \tag{3.4}$$

$$N_{11} = \frac{nD_2}{\sqrt{H}} \tag{3.5}$$

$$T_{11} = \frac{T_m}{D_2^3 H} \tag{3.6}$$

Speed n can be given in rad/s or rpm, mechanical torque T_m is in Nm. It can easily be demonstrated that these unit parameters are not dimensionless. Defining the mechanical or hydraulic efficiency η as mechanical output to hydraulic input, it can be expressed as:

$$\eta = \frac{P_m}{P_h} = \frac{T_m\,\omega}{\rho g Q H} = \frac{T_{11}N_{11}}{\rho g Q_{11}} \tag{3.7}$$

The speed number can be expressed in unit parameters as well by substitution into equation 3.1:

$$\Omega = \frac{\left(\frac{N_{11}\sqrt{H}}{D_2}\right)(Q_{11}D_2^2\sqrt{H})^{1/2}}{(2gH)^{3/4}} = \frac{N_{11}\,Q_{11}^{1/2}}{(2g)^{3/4}} \tag{3.8}$$

For equations 3.7 and 3.8, if speed n is given in rpm in stead of rad/s, a factor $\frac{\pi}{30}$ must be included in the nominator to achieve correct dimensions.

Notice how the net available head H appears in equations 3.4 to 3.6. A varying 11-parameter can not only reflect variation in its associated parameter, but also variation in head. For example variation in N_{11} can represent variation in speed at constant head, but also variation in head at constant speed (like at synchronous speed). Physically, variation in pressure head has the same effect as variation in speed, since the runner is then spinning too fast or too slow compared to the pressure head [4]. This concept is part of what makes the unit parameters convenient.

When performing model tests in a lab, it can be difficult to vary the head as this is determined by the lab configuration. The rotational speed on the other hand can easily be altered, and by presenting the measurements in this format, the speed can also represent variation in head. Since most prototypes are suppose to run steady at synchronous speed

for most of its operation, the speed range of its associated model tests in unit parameters can also reflect the variation in pressure head the prototype is meant to operate within. What variation to consider is simply a processing choice.

In theory, unit parameters will result in identical performance characteristics (Hill diagrams) for geometrical similar turbines, independent of actual size and net head [6]. This implies that for turbines of equal speed number Ω and similar velocity diagrams at design point, one can scale measurements presented in unit values according to rated head H and outlet diameter D_2.

The above parameters are slightly outdated compared to the following *unit parameters*, defined and used for instance in [5]:

$$Q_{ED} = \frac{Q}{D_2^2\sqrt{gH}} \tag{3.9}$$

$$N_{ED} = \frac{nD_2}{\sqrt{gH}} \tag{3.10}$$

$$T_{ED} = \frac{T_m}{D_2^3\rho gH} \tag{3.11}$$

Which for n given in rad/s becomes dimensionless, unlike the 11-parameters. Today, these definitions appear to be more common. The 11- or the ED-parameters can be used for similar purposes, it is only a matter of definition.

3.2 Hill diagrams

The illustration below is taken directly from [4] (p.33), and displays a typical turbine performance diagram.

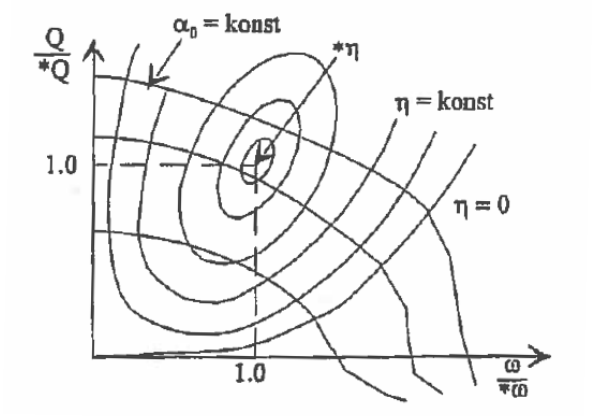

Figure 3.1: Complete performance diagram having relative speed and relative flow referring to best efficiency values on the x- and y-axis. Constant guide vane opening lines are plotted together with contour lines of constant efficiency, forming a hill towards the maximum efficiency.

Hill diagrams are obtained by performance tests on scaled models in a laboratory. The guide vane opening and rotational speed are changed successively, while the head, flow and torque are measured. Performance charts can cover the entire operating range, also outside what is possible in steady state operation for prototypes. This is useful for determining runaway conditions and behaviour for the case of load rejection [26]. Up-scaling from model to prototype is regulated by scaling conventions or laws set by the International Electrotechnical Commission (IEC). This is required due to scaling effects, mainly on the efficiency because of different Reynolds number [22]. In practice, the up-scaled Hill diagram from model to prototype can only be verified in a small part of its total operational range, since prototype turbines are designed to run at synchronous speed. Usually, runaway speed is also allowed for, such that the range between synchronous and runaway can also be measured if the turbine is disconnected from the grid [6].

Performance diagrams are commonly presented with some version of reduced or unit parameters for the axes. In this way, the net available head is also incorporated. Non-dimensionalization by best efficiency values such that the peak appears in (1,1), is often done. It is relevant to notice how different turbine types or designs will generate quite different Hill charts. To demonstrate, the below figure 3.2 from [27] also presented in [26], shows four different turbine performance diagrams.

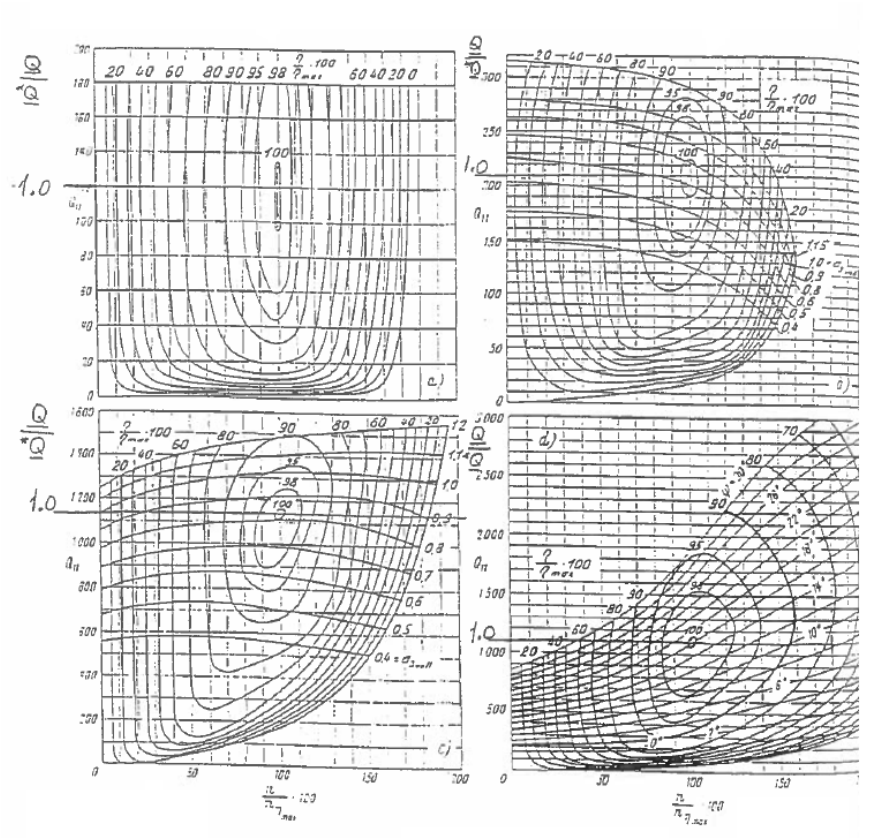

Figure 3.2: Hill charts for four different turbine types: Pelton (top left), low speed number (high head) Francis (top right), high speed number (low head) Francis (bottom left) and Kaplan (bottom right). The x- and y-axis of these diagrams are reduced quantities relative to best efficiency values.

Notice how flow-speed characteristic curves are completely flat for a Pelton turbine (top left). Flow is only influenced by nozzle opening and head, runner speed is irrelevant. The flow-speed characteristics for a Kaplan turbine (bottom right) shows highly unstable self-governing properties, since increasing runner speed will increase the flow rate. For this work, the top right and bottom left diagrams of two different Francis runners, are the most relevant. Notice the differences in gradients of the flow-speed characteristics; the low speed number one decreasing while the high speed number one slightly increasing in certain parts of the chart. Also, the general hill shape as well as the shape of the zero efficiency curve, the ”runaway curve”, is worth noticing.

Even though the Hill diagram describes *steady* performance, it is often used for *dynamic* modelling, assuming it may represent the turbine in stationary as well as dynamic performance [6]. A common way to construct a model for the entire plant is to use the Hill chart (often not directly but transformed) as a boundary condition between models for the penstock and the draft tube (upstream and downstream waterways).

3.3 Mathematical hydro turbine models

Structure and complexity of a few turbine models were studied in Chapter 2. Only the model of investigation, which is the steady version of a nonlinear turbine model based on physical laws, is considered from this point on. Since Euler's equation is part of the starting point for its derivation, a few remarks on this equation will first be included.

3.3.1 The Euler efficiency

Turbine loss involves all energy lost between spiral casing inlet and draft tube outlet, as this is the usual geometrical definition of the reaction turbine. The hydraulic power removed from the flowing water is $\rho g Q H$, where Q is the water discharge rate, and net pressure head H is the difference in energy grade line (sometimes referred to as total head) between the defined turbine inlet and outlet. In an *ideal machine*, the theoretical available specific energy given by Euler's equation, $u_1 c_{u1} - u_2 c_{u2}$, times the mass flow rate of water $\dot{m} = \rho Q$, is exactly equal to the extracted hydraulic power:

$$\rho\ g\ Q\ H\ =\ \dot{m}\ (u_1\ c_{u1} - u_2\ c_{u2})\ \Longrightarrow\ g\ H\ =\ (u_1\ c_{u1} - u_2\ c_{u2}) \tag{3.12}$$

The *Euler* or *hydraulic efficiency*, often named η_h or just η, is defined as the deviation from this ideal machine, that is, the difference between left and right side of equation 3.12:

$$\eta_h\ =\ \frac{u_1\ c_{u1} - u_2\ c_{u2}}{g\ H} \tag{3.13}$$

Equation 3.13 is correct for *any* turbomachine and holds in its *entire* operating range. There will always be a numerical value for η_h making it hold [7]. But the Euler equation does not specify *what* the efficiency function should be for the relation to hold. The accuracy of turbine models based upon Euler's equation depends on how to model this so far unknown loss function. A Hill diagram can surely be used, but might not be available.

3.4 Model derived from Euler and the opening degree

3.4.1 Nonlinear version

The turbine model under investigation is the same as in the project thesis. It is the steady version of the model proposed by prof. Nielsen in [6], excluding the empirical loss correction terms as in [5] [7] [8]. It is worth mentioning that the model is one-dimensional and based on the velocity diagrams at runner inlet and outlet for the mean streamline. A thorough derivation starting from Euler's turbine equation and the opening degree definition including consideration about equation validity, was performed in the project thesis and will not be repeated. Only the resulting governing equations for p.u. flow and torque are presented below:

$$q \;=\; y\,\sqrt{h-\sigma\,(\tilde{\omega}^2-1)} \;=\; f(h,y,\tilde{\omega}) \tag{3.14}$$

$$t \;=\; q\,(m_S-\psi\,\tilde{\omega}) \;=\; g(q,y,\tilde{\omega}) \tag{3.15}$$

Where q, h, y, $\tilde{\omega}$ and t are p.u. turbine flow rate, net head, opening degree (also denoted as κ in several works), rotational speed and torque, respectively. α_1 is defined as the angle between guide vanes and peripheral direction and is equal to the absolute flow angle at runner inlet assuming absolute flow velocity c_1 leaves the guide vane perfectly aligned. For a Francis, α_1 relates to the p.u. opening degree y (or κ) according to:

$$y \;=\; \frac{\sin\alpha_1}{\sin\alpha_{1R}} \tag{3.16}$$

The geometrical constant σ is:

$$\sigma \;=\; \frac{\omega_R^2}{8\;g\;H_R}\,(D_1^2-D_2^2) \tag{3.17}$$

The non-dimensional specific starting (when $\tilde{\omega}=0$) torque m_S is:

$$m_S \;=\; \xi\,\frac{q}{y}\,(\cos\alpha_1+\tan\alpha_{1R}\;\sin\alpha_1) \tag{3.18}$$

The two machine constants ψ and ξ are defined by the velocity vectors at BEP as:

$$\psi \;=\; \frac{u_{2R}^2}{\eta_R\;g\;H_R} \tag{3.19}$$

$$\xi \;=\; \frac{u_{1R}\; c_{1R}}{\eta_R\; g\; H_R} \tag{3.20}$$

For ξ, it was derived an expression for calculating it from ψ and α_{1R}, which is necessary for calculations in this work:

$$\xi \;=\; (1+\psi)\;\cos\alpha_{1R} \tag{3.21}$$

Subscript "R" denotes rated/ nominal values, which are assumed to correspond to BEP.

3.4.2 Linear version and characteristic coefficients

In the linearized model version, deviation in the dependent variables q or t, can be estimated from deviation in the independent variables times the partial derivatives with respect to these at the point of linearization. The steady state point is denoted with subscript '0' and the variation around it as $\Delta x = x - x_0$. The linearized equations are on the form:

$$\Delta q \;=\; a_{11}\;\Delta h \;+\; a_{12}\;\Delta y \;+\; a_{13}\;\Delta\tilde{\omega} \tag{3.22}$$

$$\Delta t \;=\; a_{21}\;\Delta q \;+\; a_{22}\;\Delta y \;+\; a_{23}\;\Delta\tilde{\omega} \tag{3.23}$$

Where the partial derivatives are the *characteristic coefficients* of the linear equations, a_{ij} for $i = 1, 2$, $j = 1, 2, 3$:

$$a_{11} \;=\; \left.\frac{\partial q}{\partial h}\right|_0 \qquad a_{12} \;=\; \left.\frac{\partial q}{\partial y}\right|_0 \qquad a_{13} \;=\; \left.\frac{\partial q}{\partial\tilde{\omega}}\right|_0 \tag{3.24}$$

$$a_{21} \;=\; \left.\frac{\partial t}{\partial q}\right|_0 \qquad a_{22} \;=\; \left.\frac{\partial t}{\partial y}\right|_0 \qquad a_{23} \;=\; \left.\frac{\partial t}{\partial\tilde{\omega}}\right|_0 \tag{3.25}$$

Mathematical expressions for these characteristics were thoroughly derived both for Francis and Pelton in the project thesis by performing partial derivation of equations 3.14 and 3.15. The resulting expressions for Francis only are repeated in Appendix A2.

Since the dependent variable q is an independent variable in the torque expression, equation 3.23 (or its nonlinear version 3.15) can be rewritten to have the same independent variables as the flow equation. Substituting equation 3.22 into 3.23 yields:

$$\Delta t \;=\; (a_{21}\;a_{11})\;\Delta h \;+\; (a_{21}\;a_{12} + a_{22})\;\Delta y \;+\; (a_{21}\;a_{13} + a_{23})\;\Delta\tilde{\omega} \tag{3.26}$$

Equation 3.26 implies that:

$$a_{21}\ a_{11}\ =\ \left.\frac{\partial t}{\partial h}\right|_0 \qquad a_{21}\ a_{12}+a_{22}\ =\ \left.\frac{\partial t}{\partial y}\right|_0 \qquad a_{21}\ a_{13}+a_{23}\ =\ \left.\frac{\partial t}{\partial \tilde{\omega}}\right|_0 \tag{3.27}$$

All of the partial derivatives presented above implies the other independent variables in the equation as constant during differentiation. Physically, guide vane opening y and flow rate q are closely coupled. It is nonphysical for one parameter to change without affecting the other, but this is in fact what the characteristic coefficients a_{21} and a_{22} implies. Even though easily calculated analytically, they are more difficult to extract in reality. The purpose of including the above reformulation will be more clear when gradients are extracted from experimental data later.

Continuing, a linear version of the p.u. mechanical power $p = t\tilde{\omega}$, must be included:

$$\Delta p\ =\ a_{31}\ \Delta t\ +\ a_{32}\ \Delta\tilde{\omega} \tag{3.28}$$

Where the characteristics for an arbitrary linearization point "0" are the gradients:

$$a_{31}\ =\ \left.\frac{\partial p}{\partial t}\right|_0\ =\ \left.\tilde{\omega}\right|_0 \qquad a_{32}\ =\ \left.\frac{\partial p}{\partial \tilde{\omega}}\right|_0\ =\ \left.t\right|_0 \tag{3.29}$$

3.4.3 Block diagram representation

Combining equations 3.22, 3.23 and 3.28 with the simplest transfer function penstock model from section 2.1.1, $\dfrac{\Delta h}{\Delta q} = -T_{wp}s$, yields the block diagram presented below. This waterway model was the simple incompressible friction-free pipe model also used in the project thesis. In the below diagram, deviation from linearization point can be assumed implicit such that 'Δ' can be omitted, this is only a matter of definition.

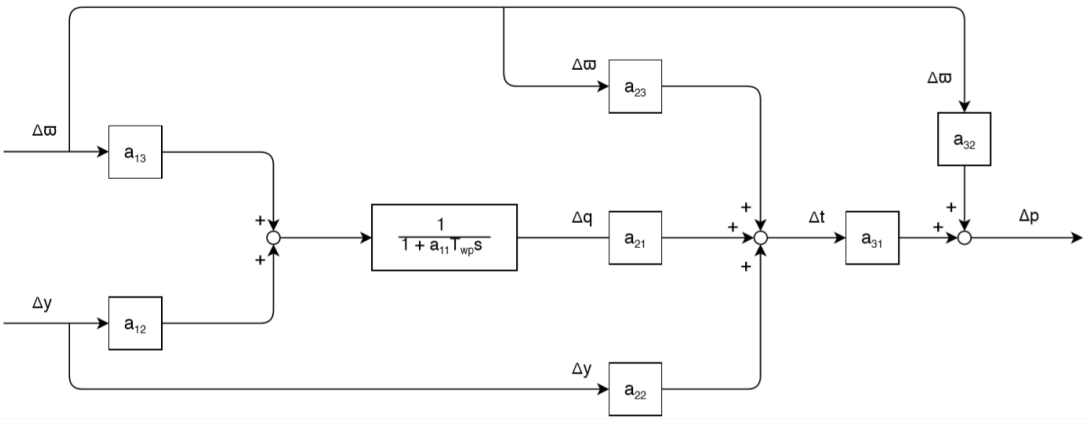

Figure 3.3: Block diagram representation of the waterway and turbine linear models for an arbitrary point of linearization.

This can easily be implemented into existing simulation software, as performed in the project thesis in SIMULINK.

3.4.4 Efficiency prediction

The accuracy of the presented equations in predicting real turbine performance, will be investigated. It will first be investigated analytically/ mathematically in this section as part of the theoretical background.

In the project thesis, p.u. hydraulic efficiency $\tilde{\eta}$ was found as:

$$\eta \;=\; \frac{T\;\omega}{\rho\;g\;Q\;H} \;=\; \frac{t\;T_R\;\tilde{\omega}\;\omega_R}{\rho\;g\;q\;Q_R\;h\;H_R} \;=\; \eta_R\;\frac{t\;\tilde{\omega}}{q\;h} \tag{3.30}$$

$$\Longrightarrow \tilde{\eta} \;=\; \frac{\eta}{\eta_R} \;=\; \frac{t\;\tilde{\omega}}{q\;h} \tag{3.31}$$

Substituting for p.u. head h and torque t resulted in:

$$\tilde{\eta} \;=\; \frac{(m_S - \psi\;\tilde{\omega})\;\tilde{\omega}}{\left(\frac{q}{y}\right)^2 \;+\; \sigma\;(\tilde{\omega}^2 - 1)} \tag{3.32}$$

Assuming geometry, machine constants and guide vane angle at BEP to be constants for a given unit. For any operating point, the efficiency have the following functionality:

$$\tilde{\eta} \;=\; f(q, y, \tilde{\omega}) \tag{3.33}$$

The *direct* functional dependency upon pressure disappeared, but q is functional dependent upon h such that it is included *indirectly*. For a given opening degree, efficiency is a function of flow (or head) and rotational speed. It can be visioned like a 3D surface in a 3D diagram, which is the Hill diagram where contour lines represent constant efficiency levels. It is common to present the abscissa and ordinate axes using unit speed and unit flow, such that the head is also included, as explained previously.

To investigate the model's efficiency prediction, one may look at the functionality of equation 3.32 when each of the independent variables are held constant at their rated value. As before, we assume σ, ψ, ξ and α_{1R} to be known constants for a given unit.

- **Constant rated guide vane angle;** $\alpha_1 = \alpha_{1R} \Longrightarrow y = 1$

$$\tilde{\eta}\,\Big|_{y=1} \;=\; \frac{\left(\dfrac{\xi\, q}{\cos\alpha_{1R}} \;-\; \psi\,\tilde{\omega}\right)\tilde{\omega}}{q^2 \;+\; \sigma\,(\tilde{\omega}^2 - 1)} \;=\; f(q,\tilde{\omega}) \tag{3.34}$$

  In the Hill diagram, we are moving along the curves of constant opening degree, also known as the flow-speed characteristic curves. The shape of these depend on turbine self-governing properties. For a Pelton turbine, keeping the opening constant is equivalent to keeping the flow constant (if head is also constant). That is, $\tilde{\eta}|_{y=1} = f(\tilde{\omega})$ for impulse turbines. This does not hold for a Francis where also runner speed impacts the flow (or the head), not just the opening.

  According to [5] [7] [8], along constant y lines losses are mainly determined by the speed of the turbine, spinning too slow or too fast compared to optimal inlet and outlet angles (velocity diagrams). Such effects are already included to a great extent in the model equations, due to the Euler turbine equation being the starting point for derivation. It can be seen as a strong mathematical dependency of $\tilde{\eta}$ with respect to $\tilde{\omega}$ in relation 3.34 above.

- **Constant rated flow;** $q = 1$

$$\tilde{\eta}\,\Big|_{q=1} \;=\; \frac{\left(\dfrac{\xi}{y}\,(\cos\alpha_1 + \tan\alpha_{1R}\,\sin\alpha_1) \;-\; \psi\,\tilde{\omega}\right)\tilde{\omega}}{\left(\dfrac{1}{y}\right)^2 \;+\; \sigma\,(\tilde{\omega}^2 - 1)} \;=\; f(y,\tilde{\omega}) \tag{3.35}$$

  In the Hill diagram, we are moving horizontally along the x-axis. Opening degree, head and speed may vary for a constant flow rate to hold for a reaction turbine.

- **Constant rated rotational speed;** $\tilde{\omega} = 1$

$$\tilde{\eta}\Big|_{\tilde{\omega}=1} = \frac{m_S - \psi}{\left(\frac{q}{y}\right)^2} = \frac{\xi \frac{q}{y} \left(\cos\alpha_1 + \tan\alpha_{1R} \sin\alpha_1\right) - \psi}{\left(\frac{q}{y}\right)^2} = f(q, y) \quad (3.36)$$

In the Hill diagram, we are moving vertically along the y-axis. This is simply the performance measured on site when runner is kept at synchronous speed and the flow is varied by changing position of the guide vanes. The head may also vary with the flow according to the valve equation when $\tilde{\omega} = 1$:

$$q = y\sqrt{h} \implies h = \left(\frac{q}{y}\right)^2 \quad (3.37)$$

Therefore, at rated speed the efficiency may also be expressed as a function of pressure head h and guide vane angle α_1 or opening y:

$$\tilde{\eta}\Big|_{\tilde{\omega}=1} = \frac{\xi\sqrt{h}\left(\cos\alpha_1 + \tan\alpha_{1R} \sin\alpha_1\right) - \psi}{h} = f(h, \alpha_1) = f(h, y) \quad (3.38)$$

- **Constant rated rotational speed and flow;** $\tilde{\omega} = 1$**,** $q = 1$

For most operation in real plants, runner speed is constant since it is connected to the grid. If also flow rate is kept constant, the efficiency may only vary as a consequence of variation in pressure head. Since $q = 1, \tilde{\omega} = 1$ in the flow equation yields $y = 1/\sqrt{h}$, and $\alpha_1 = \arcsin\left(y \sin\alpha_{1R}\right)$, this can be substituted into the above equation 3.38 to yield:

$$\tilde{\eta}\Big|_{\tilde{\omega}=1,q=1} = \frac{\xi\sqrt{h}\left(\cos\left(\arcsin\left(\frac{\sin\alpha_{1R}}{\sqrt{h}}\right)\right) + \frac{\tan\alpha_{1R} \sin\alpha_{1R}}{\sqrt{h}}\right) - \psi}{h} = f(h) \quad (3.39)$$

The net head available for the turbine is the difference between spiral casing inlet and draft tube outlet energy grade lines (EGL). The inlet EGL is equal to the upper reservoir water level minus head losses in the waterways. These head losses

are typically friction and component single losses, and they are increasing with the flow squared. The outlet EGL is equal to the lower reservoir water level plus head losses due to the (usually low but still finite) water velocity at the draft tube exit [28].

For constant rotational speed and flow thus constant head losses, in theory, only change in upstream or downstream reservoir levels can change the head and therefore change the efficiency. If water levels are constant, we are at a single working point in the Hill diagram and $\tilde{\eta}$ is constant.

- **Constant rated rotational speed and net head;** $\tilde{\omega} = 1$**,** $h = 1$

  If we picture being able to keep the net head approximately constant at its rated value for varying flow by minor changes to the reservoir levels, while also running av constant rated speed, the p.u. efficiency simplifies to:

$$\tilde{\eta}\Big|_{\tilde{\omega}=1,h=1} = m_S - \psi = \xi(\cos\alpha_1 + \tan\alpha_{1R}\sin\alpha_1) - \psi = f(\alpha_1) \tag{3.40}$$

  In 3.40, the only dependency on varying discharge is through the guide vane angle in m_S, as also stated in [5]. Substituting for $\alpha_1 = \arcsin(y\sin\alpha_{1R})$ and using that $q = y$ when $h = 1$ and $\tilde{\omega} = 1$, yields:

$$\tilde{\eta}\Big|_{\tilde{\omega}=1,h=1} = \xi(\cos(\arcsin(q\sin\alpha_{1R})) + q\tan\alpha_{1R}\sin\alpha_{1R}) - \psi = f(q) \tag{3.41}$$

  The dependency upon q in equation 3.41 is present, but very weak. This was illustrated in the project thesis where it was used for plotting efficiency versus flow rate for three different turbines, two Francis of different design heads and one Pelton. The curve was unrealistically flat for the Francis turbines and completely flat for the Pelton, clearly illustrating the need for model improvement on this matter.

The model equations' strengths and weaknesses will be further analysed in Chapter 6 by comparison to experimental data. But in order to understand more about the physics of this issue, some more theoretical background about energy loss in reaction turbines is crucial.

3.5 Energy loss in reaction turbines

In this section, different energy loss contributions in a Francis turbine will be reviewed from a physical and mathematical perspective. The objective is to uncover the most relevant contributions with respect to the considered model drawback. That is, which are more significant when discharge is at off-design conditions. The classification is similar to [4] [6] and several other references cited previously. The different types will be reviewed and common loss model functionalities presented.

3.5.1 Friction loss

Viscous effects of water flowing through different turbine components will result in energy losses in all flow domains, analogous to continuous frictional head losses in a pipe. Some references like [16], refer to these as "hydraulic losses", but this name can become too general. Expressed as a head loss ΔH, frictional losses are typically expressed as [6]:

$$\Delta H \ = \ R_f \ Q^2 \tag{3.42}$$

or even more general:

$$\Delta H \ = \ f[Q^2] \tag{3.43}$$

Thus being non-zero also at optimum design flow, $Q = Q_R$.

Because of the complex geometry of the flow fields, the proportionality constant or function is challenging to determine. For a simple pipe of constant diameter, the proportionality constant becomes $\frac{8fL}{g\pi^2 D^5}$ [28]. For the converging spiral casing, the distributor, the runner flow channel or the diverging and usually elbow-shaped draft tube channel, the calculation becomes far more complex. According to [22], flow friction typically accounts for 2.0 - 2.5 $\%$ loss in a high head Francis.

3.5.2 Incidence loss

Also referred to as shock or impact loss in literature, incidence loss are present whenever there is mismatch between relative flow direction and runner blade orientation at inlet. The incidence angle is defined in 2D as the difference between runner blade angle at inlet (fixed) and relative inflow angle β_1. Incidence losses are assumed to be zero whenever the incidence angle is zero [6]:

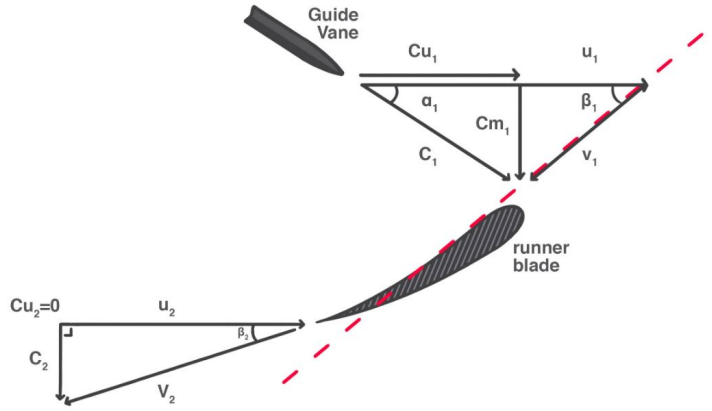

Figure 3.4: Best efficiency operation corresponding to optimal guide vane setting. The optimal velocity diagrams have zero incidence angle at runner inlet (1) and zero rotational component of the absolute velocity at runner outlet (2).

The turbine is designed to have zero incidence angle at optimum operating conditions. Expressed as a head loss ΔH, these losses typically have the following functionality:

$$\Delta H \;=\; R_i \,(Q - Q_R)^2 \tag{3.44}$$

or even more general:

$$\Delta H \;=\; f[(Q - Q_R)^2] \tag{3.45}$$

An alternative is to distinguish between positive and negative incidence angles (depending on definition, corresponding to lower or higher flow than design), and express incidence loss as a function of the difference in tangential components of the relative fluid velocities squared, as in [10]. Alternatively, one can use directly the vector difference between real and optimal relative fluid velocity for the same discharge squared, as in [24].

3.5.3 Residual swirl in draft tube

In some references simply referred to as ”draft tube losses”, but this can become too general. The considered energy loss is due to swirling of the water at runner outlet, that is, the peripheral ’u’ component of the absolute flow velocity is non-zero. The energy lost is the kinetic energy of this velocity vector component, i.e. $\propto \frac{c_{u2}}{2}$.

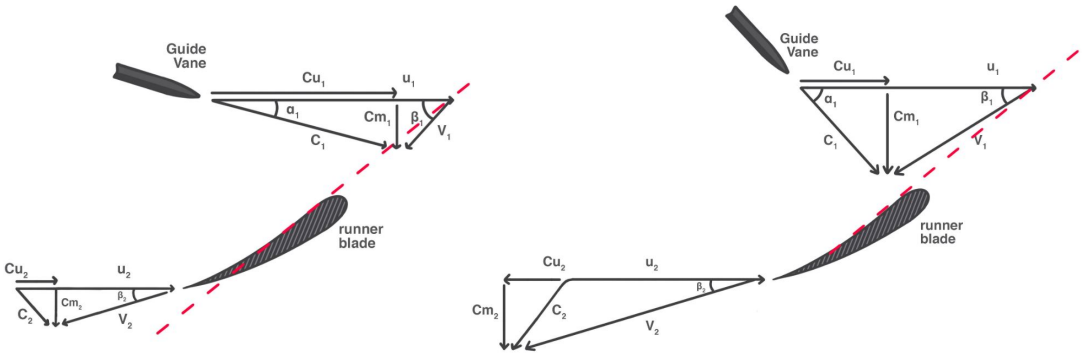

Figure 3.5: (a) Low flow operation (low guide vane angle) and (b) high flow operation (high guide vane angle) velocity diagrams at runner inlet (1) and outlet (2). The former yields positive outlet rotational component, $c_{u2} > 0$, the latter yields negative, $c_{u2} < 0$, defined according to runner rotational direction. There are energy losses associated with both operational modes.

The turbine is designed to have zero swirl at runner outlet such that this loss is zero at optimum conditions, as in figure 3.4. Some references use the peripheral velocity component directly in defining this head loss:

$$\Delta H \;=\; R_d \, c_{u2}^2 \tag{3.46}$$

Others use the flow rate in a similar functionality as for the incidence loss:

$$\Delta H \;=\; R_d \, (Q - Q_R)^2 \tag{3.47}$$

or even more general:

$$\Delta H \;=\; f[(Q - Q_R)^2] \tag{3.48}$$

Some references like [16], choose to categorize the two losses, inlet incidence and outlet swirl, together for modelling purposes. This seems reasonable given that both are present when velocity diagrams deviate from design operation, illustrated in figure 3.5.

3.5.4 Minor kinetic losses

Even for zero incidence angle, there will be *some* wake effect from the guide vanes on the flow. The stay vanes are designed to not disturb the flow (streamlined), but can still impose a minor loss. In the draft tube, diffusion and bend losses will also be present [10]. These are all examples of minor kinetic head losses. They can cause flow separation and mixing, but for a well-designed turbine should be minor compared to other losses like viscosity or mechanical friction. In several reviewed references, minor kinetic losses are completely

neglected in the turbine loss model. If they are included, one may express them as a head loss having similar functionality as single losses in a pipe [28]:

$$\Delta H \;=\; R_k \, Q^2 \tag{3.49}$$

Where the loss coefficient R_k must be specified for each considered component, analogous to K_L in [28] (ch. 8-6).

3.5.5 Leakage loss

Also referred to as volumetric loss, water which escapes the runner wheel will not produce any work. This energy loss cannot be considered as a head loss, but it is directly proportional to the flow rate of water leaked, $Q_{leakage}$, and should be included in the total efficiency. According to [16], it depends on the sealing method, and according to [6], it is also a function of local pressure differences in the turbine. Usually, one can assume leakage loss to only alter the efficiency *level*, not the *shape*, as this loss is relatively constant for varying speed or flow [6]. It appear common to assume a constant percentage for the leakage flow, ranging from about 0.25 $\%$ as in [16] to about 1.5 $\%$ as in [22].

3.5.6 Disk friction loss

Since the runner is installed close to stationary components separated by a layer of water, there will be disk friction present on the outside of the runner, both at the runner band (side) and crown plate (top). According to [22], disk friction typically accounts for 1.5 $\%$ loss in a high head Francis. [29] specifically investigated disk friction and leakage loss in a Francis model turbine using CFD. Among other findings, they found that the total disk friction power loss is proportional to the rotational speed cubed:

$$\Delta P \;=\; R_{df} \, n^3 \tag{3.50}$$

Similar functionality is stated also in [4] [6], but in [6] as a loss in torque and thus $\propto n^2$.

[29] investigated other effects on disk friction and leakage loss as well, including clearance size, rotor-stator interactions, and looked at differences near the runner band versus crown as well as pressure versus suction side of the blade.

3.5.7 Mechanical friction loss

External mechanical friction in bearings, shaft seals and wear rings are also present. [10] provide a simple estimation method for this loss by measuring the power required to spin the runner at synchronous speed in air (drained from water). They assume this power loss ΔP to be constant over its operating range. Other references like [6] [16] assume the mechanical friction to be a function of rotational speed, thus relatively constant for a prototype running at synchronous speed. [6] proposes the same functionality as for disk

friction and categorizes these types together. Expressed as a power loss ΔP, the typical functionality can be:

$$\Delta P \; = \; R_{mf} \; n^3 \tag{3.51}$$

Based on the above presented functional dependencies, hydraulic friction, incidence, draft tube swirl and minor kinetic losses, appear to be most relevant for the subsequent work of improving model efficiency prediction for constant speed varying flow conditions. Leakage, disk and mechanical friction losses appear to be less relevant, as these are relatively constant for constant rotational speed.

Chapter 4

Experimental Data

4.1 Data access and anonymization

Experimental data from Rainpower were accessed to analyze model performance with respect to real turbine performance, and produce incipient efficiency η_i curves based on the data. Rainpower is a specialized hydro power business which deliver different equipment necessary in a hydro power plant, from runners to governor systems. They are a project organization based on Norwegian technology [30]. Aside from thesis supervisor Bjørnar Svingen, the author's contact person in Rainpower was Henning Lysaker, CTO of the turbine laboratory in Trondheim. As requested by Rainpower, their data will be presented anonymously throughout the thesis.

Measurements covering a certain operational range were given for three different model Francis turbines. One designed for very low pressure head operation (high specific speed), one for medium pressure head (medium specific speed) and one for very high pressure head (low specific speed). A "typical" classification for low, medium and high head Francis can be $<$100 m, 100-300 m and $>$300 m, respectively. The accessed data are for the extreme of these heads, but exact values are excluded in the thesis.

Measurements of mechanical efficiency η_m (also referred to as hydraulic efficiency) were given with measurements of flow Q, speed n, mechanical torque T_m and guide vane opening angle α_1. Flow, speed and torque were presented as unit properties, Q_{11}, N_{11} and T_{11}, such that also variation in head is included, as explained in the theory. The design prototype heads were given, and the point of best efficiency was easily located. The best efficiency measurements will not be presented in numerical values, but denoted with subscript 'R'. All data will be presented in per unit with respect to these. Only presented in numerical values are the *estimated* speed numbers since these are highly relevant. They will be presented in Chapter 5.

The design of a Francis runner depends greatly on the ranges of pressure head and

discharge it is suppose to operate within, as well as the synchronous speed. Since experimental data were given for three turbines designed to operate under very different heads, this will allow us to analyze model performance for three very different designs.

4.2 Assumptions and restrictions

When dividing by rated value, notice how the net head ratio H/H_R will appear when using 11-parameters (or any version of reduced parameters):

$$\frac{Q_{11}}{Q_{11R}} = \sqrt{\frac{H_R}{H}}\,\frac{Q}{Q_R} = \frac{q}{\sqrt{h}} \tag{4.1}$$

$$\frac{N_{11}}{N_{11R}} = \sqrt{\frac{H_R}{H}}\,\frac{n}{n_R} = \frac{\tilde{\omega}}{\sqrt{h}} \tag{4.2}$$

$$\frac{T_{11}}{T_{11R}} = \frac{H_R}{H}\,\frac{T_m}{T_{mR}} = \frac{t}{h} \tag{4.3}$$

Where h, q, $\tilde{\omega}$ and t are the "real" p.u. head, flow, speed and torque, respectively, as referred to in previous chapters and the project thesis. If all measurements are assumed at rated head, $h = 1$, p.u. 11-parameters are equivalent to p.u. real parameters. In reality, the pressure head will vary with reservoir levels (less relevant for lab experiments but more relevant for prototypes), with flow rate (head losses are a function of flow) and with runner speed. For measurements in a certain operational range of speed and flow, a certain variation in head will also occur depending on the plant.

When presenting the data graphically, one can choose to keep certain parameters constant while letting others vary. This will be clearly specified. For several of the characteristics of interest, the net head is assumed constant equal to its rated value, $H = H_R$ or $h = 1$, such that p.u. 11-parameters can be used directly. It will also be of interest to present the data for runner speed assumed constant equal to its rated value, $n = n_R$ or $\tilde{\omega} = 1$. The p.u. expression for head then becomes:

$$\frac{N_{11}}{N_{11R}} = \sqrt{\frac{H_R}{H}}\,1 \implies h = \frac{H}{H_R} = \left(\frac{N_{11R}}{N_{11}}\right)^2 \tag{4.4}$$

Equation 4.4 gives a basis for evaluating head variation using unit speed N_{11} values. The p.u. flow and torque becomes:

$$q = \frac{Q}{Q_R} = \sqrt{h}\,\frac{Q_{11}}{Q_{11R}} = \frac{N_{11R}}{N_{11}}\,\frac{Q_{11}}{Q_{11R}} \tag{4.5}$$

$$t = \frac{T_m}{T_{mR}} = h\,\frac{T_{11}}{T_{11R}} = \left(\frac{N_{11R}}{N_{11}}\right)^2\,\frac{T_{11}}{T_{11R}} \tag{4.6}$$

Expressions 4.4 to 4.6 can be used for example when plotting variation in flow q with head h at constant speed for different constant guide vane openings.

It will become evident that these data capture a limited range with respect to high flow operation. For example at rated head and speed, the low head turbine captures up to about 1.20 in p.u. flow, medium head about 1.48 and high head about 1.57, of *useful* data. The relevant speed ranges varied as well. For example, the medium head turbine was tested up to runaway conditions, while these were only partly or not at all covered for the low and high head turbines. Model analysis and improvement based on these data will only be valid in the operational ranges of available data, outside can at best be commented on.

4.3 Data processing

The experimental data were loaded from Excel into MATLAB where most processing took place. Scripts were written to sort appropriately, perform interpolation, produce relevant plots, estimate gradients, and so on. All relevant MATLAB scripts are described in Appendix A3 and the executable m-files given as attachments to the thesis.

Some of the data contained measurements from operation in so-called ”turbine brake mode”. This mode is in the first quadrant of a speed-flow characteristics such that speed and flow are still in the direction corresponding to normal turbine operation. However, operation is below the zero efficiency line and the generator will need to deliver energy to surpass this line in steady state operation. Turbine brake mode is unstable and outside of normal operating procedures for prototypes, but may be deliberately entered in the laboratory [31]. Measurements of negative efficiencies and negative torque are considered irrelevant for the objective of this work (to model normal turbine operation and calculate Hill charts), and therefore filtered out during the processing.

4.4 Characteristic curves

The following performance curves are extracted by interpolation of the experimental data, and will be presented in Chapter 6:

1. Flow - speed at different guide vane openings under constant pressure head
2. Torque - speed at different guide vane openings under constant pressure head
3. Output power - speed at different guide vane openings under constant pressure head
4. Efficiency - speed at different guide vane openings under constant pressure head
5. Torque - guide vane opening at different runner speeds under constant pressure head

6. Efficiency - flow at different runner speeds under constant pressure head
7. Flow - head at different guide vane openings at constant rated speed

Several other combinations can easily be extracted, but the above listed curves were considered the most relevant. Clearly, combining plots 4. and 6. as a contour plot of constant efficiency levels together with the curves of plot 1, yields a Hill diagram. Complete performance diagrams were also generated, but for the purpose of capturing even more details, 2D-plots are very useful. Hill charts from the experimental data will be presented in section 7.4 together with Hill charts calculated by the model equations.

4.5 Characteristic coefficients

The characteristic values of a linear model are the partial derivatives of the dependent variables with respect to the independent variables. To obtain these from measurements involves finding the gradients of different parameters while others are kept constant. This was achieved by interpolation of the data depending on the gradient of interest. Evaluating this interpolant, one can numerically estimate its first derivative in any point in the data.

From Theory section 3.4, the coefficients a_{11}, a_{12} and a_{13} in any working point can be extracted directly from the flow-head, flow-opening and flow-speed diagrams, respectively. From relation 3.27, the coefficient a_{21} must be calculated from the torque-head and flow-head diagrams, coefficient a_{22} from the torque-opening, torque-head, flow-head and flow-opening diagrams, and finally coefficient a_{23} from the torque-speed, torque-head, flow-head and flow-speed diagrams. To summarize, the relevant gradients necessary to obtain all six characteristic coefficients a_{ij} of the linear model in this work, are:

$$\left.\frac{\partial q}{\partial h}\right|_{y,\tilde{\omega}\ const} \quad \left.\frac{\partial q}{\partial y}\right|_{h,\tilde{\omega}\ const} \quad \left.\frac{\partial q}{\partial \tilde{\omega}}\right|_{h,y\ const} \quad \left.\frac{\partial t}{\partial h}\right|_{y,\tilde{\omega}\ const} \quad \left.\frac{\partial t}{\partial y}\right|_{h,\tilde{\omega}\ const} \quad \left.\frac{\partial t}{\partial \tilde{\omega}}\right|_{h,y\ const} \tag{4.7}$$

To check the accuracy of the numerically estimated gradients, the derivative was calculated both using a central difference scheme and the inbuilt function *gradient()*. Prior to the calculations, the data were smoothed using inbuilt function *smooth()* to "filter out" smaller irregularities from general curve shape. The grid size (Δh, Δy or $\Delta\tilde{\omega}$) for the central difference scheme was originally 0.01, but also doubled, tripled and halved, to check for convergence in the numerically estimated gradient. An accuracy of two decimal places was considered adequately.

Because of the extensive analysis for the nonlinear model, only characteristics at BEP were investigated for the linear model. The mathematical expressions for a_{ij} from the model will be compared to numerical gradients from the experiments in section 6.2.

Chapter 5

Model Input

To set up and perform simulations with the model in per unit only requires σ, α_{1R}, ψ and ξ as inputs. σ is the geometrical constant, containing information about self-governing properties (stability) among other things. α_{1R} is the optimal guide vane angle resulting in zero incidence angle at rated speed. Machine constant ψ contains information about the torque-speed relationship, and machine constant ξ is given by ψ and α_{1R} according to relation 3.21. These input parameters are equal for geometrical similar turbines, that is, turbines of equal speed number and reduced velocity diagrams at BEP.

To make the comparison between model and experimental data valid, the correct input values must be given to the model. These correspond to a turbine design geometrical similar to the turbine of the measurements. As explained in Chapter 4, no information about turbine geometry was given by Rainpower. The appropriate designs had to be obtained by ”reverse engineering” based on the best efficiency points from the experimental data.

To design turbines which are geometrical similar to those from the data turned out to be ambiguous and challenging. Two main approaches of ”tuning” the design according to the measurements was performed. Even though one approach resulted in more accurate simulations (characteristics) than the other, better understanding about the model and its inputs was achieved after performing both. Interesting observations on runner design procedure also resulted from the process. For these reasons, both methods will be described in detail and resulting simulations presented and discussed. For clarity, the two approaches will be denoted as ”Method 1” and ”Method 2”.

The model-to-measurement comparison can be performed in p.u. or scaled by rated values when these are known. The former is preferred to maintain generality in the analysis, but also since the data were given in scalable unit parameters.

5.1 Method 1: Turbine design based on measured best efficiency point

The first approach is based on the assumption that measured best efficiency point should correspond to design point. Only knowing the highest efficiency and associated 11-parameters at this point, speed number was calculated by equation 3.8. It was considered essential to design runners with the same speed numbers, and to obtain α_{1R} not too far off the measured ones, as similarity also implies equal velocity diagram angles.

5.1.1 Francis runner design

The Francis runner wheel design recipe used, is according to student courses at NTNU, described for example in [4] [8] [26], developed by professor Hermod Brekke. The procedure was also described in Appendix A4 of the project thesis. It is a simple recipe to obtain approximate main runner dimensions. Knowing the nominal head and flow, one starts with the outlet, making reasonable assumptions about peripheral velocity u_2 and relative flow angle β_2 according to necessary submergence with respect to cavitation limits. Outlet dimensions are calculated and adjustments made to achieve rotational speed synchronous with the grid frequency. Design of runner inlet also begins with empirically based assumptions. The relation between reduced peripheral velocities, $\underline{u}_1$ and $\underline{c}_{u1}$, should minimize shock loss for varying guide vane angle. According to [26], $\underline{u}_1$ should be 0.7 - 0.76. Acceleration of meridional velocity is another necessary assumption. Since slightly accelerated flow reduces the chance of flow separation, 10% acceleration is often assumed in Norway [26]. $\underline{u}_1$ = 0.73 and $c_{m2} = 1.1c_{m1}$ was assumed in this design.

The starting point of "Method 1" is the measured BEP in unit parameters and the assumption that this should equal design point. Prototype design heads were also given, but this is a matter of scaling and we are free to choose different heads. Since the objective is to design three very different runners, design heads should comply reasonably with the usual classification of low, medium and high head Francis. Therefore, the prototype heads were used directly, but will remain anonymous in the thesis.

Different from the recipe, this starting point does not specify a design flow rate or synchronous speed, they are given implicitly in the best point Q_{11} and N_{11} values for a certain outlet diameter and head (a matter of scaling). The scaling is performed firstly by choosing the head, and secondly by choosing the number of pole pairs in the generator. A larger number gives a larger runner and a larger power output. Since these runners are not designed for some expected discharge or desired output, but mainly since the similarity is conserved and the model input parameters are completely unaffected by the scaling, the choice is quite arbitrary in this case. The number of pole pairs gives the synchronous speed which from the best point N_{11} specifies the outlet diameter. The design flow rate and torque are then specified by the best point Q_{11} and T_{11} values.

Following, outlet velocity diagrams can be calculated assuming zero outlet swirl at de-

sign point, $c_{u2} = 0$. This procedure does not guarantee a good design, every value should be compared to empirical values. u_2 should in general be between 35 - 43 m/s however not exceed $\sqrt{2gH}$ by too much (this applies to all velocities). β_2 should in general be between 13 - 22° [26]. The empirical relations for the inlet stated above are used together with measured maximum efficiency to determine inlet velocity diagrams. The resulting design α_1 should not be too far off the measured one, if so, the design should be reviewed. At this point, inlet dimensions D_1 and B_1 can be calculated. One may check the turbine's self-governing properties by comparing inlet and outlet diameters; $D_1 > D_2$ implies stable self-governing and $D_1 < D_2$ implies unstable. Nevertheless, we can expect to achieve $D_1 < D_2$ (considering *average* D_1 as it will vary across inlet section) slightly for a Francis runner operating under a very low pressure head [5].

Finally, σ, ψ and ξ can be calculated from design values, and together with α_{1R}, they make up the model input. Knowing all the rated values makes it possible to convert from non-dimensional to dimensional values later.

An Excel sheet was developed to design according to this recipe, but is not included in the thesis to protect Rainpower's data. MATLAB scripts read from this sheet the relevant model inputs in order to plot characteristics, etc. The Excel sheet 'francis design.xls' provided by Bjørnar Svingen (attachment to the LVTrans Manual [15]) was used to check the design afterwards. This sheet is based on the same procedure, but having nominal head and flow as starting point. There was no problem to design reasonable high and medium head turbines by this approach, but the low head caused some challenges, discussed in the following section.

5.1.2 Discussion of the low head turbine design

Problems arose when trying to design a low head (high Ω) Francis runner by this recipe based on the low head measurements' BEP. The prototype design head is very low, and the speed number calculated from measured best point, about 1.14, is very high. According to references like [4] (p. 29) it is in the "transition region" between Francis and Kaplan. Such combination of nominal head, flow and speed can also belong to a Kaplan turbine, and both choices could be adequate during a design process.

Such low head Francis can have a high inlet height B_1 with varying diameter D_1 across the inlet section. Recall that Kaplan is an axial flow machine while Francis is a mix of radial inflow and axial outflow. In the attempted procedure, the velocity diagrams are based on the assumption of radial inflow and axial outflow, but the former can become inaccurate for a very low head Francis. Further, the outlet diameter can be slightly larger than the average inlet diameter. In the turbine model, this implies a negative σ value.

In general, the design procedure by Hermod Brekke does not work for Kaplan turbines [15], and may not work too well for *very* low head Francis either. It was evident when trying to dimension according to this recipe, that the resulting runner will have a very

unstable design. The ratio of D_1 to D_2 becomes too low (σ too negative) to make much sense when given as input to the model.

One problem is the recommendation of u_2. A very low design head gives a very low corresponding maximum water velocity $\sqrt{2gH}$. The peripheral runner speed is not restricted by this, but exceeding it too much may result in water velocity components exceeding $\sqrt{2gH}$, which is physically impossible. The "quick fix" in 'francis design.xls' is to choose u_2 equal to the lower of 40 m/s and $\sqrt{2gH}$. Violating the recommended speed range can be necessary, but a low u_2 may result in β_2 larger than its recommended range. From the outlet velocity diagram, a larger β_2 will give a larger c_{m2} which will give a larger c_{m1} and thus larger design α_1, which may be far off the one at measured best point. The ratio of D_1 to D_2 becomes more reasonable, i.e. not too unstable, but the velocity diagrams are not optimal anymore, the angles are very large and the design point do not match measured BEP anymore. More reasonable (lower) design angles for β_2 and α_1 will give higher u_2 and more unstable diameter ratio. Based on this reasoning, the recipe appears to not work so well for designing *very* low head Francis runners.

To quickly check the resulting design, the same nominal head, flow and speed were given as inputs to the more sophisticated design software ALAB. The resulting ALAB design actually got D_1 slightly larger than D_2, however also this resulting in somewhat nonphysical simulations with the model equations. This implies that σ alone is not the sole cause for errors, also the other input parameters, α_{1R}, ψ or ξ, can cause trouble. In general, all three becomes larger for low head than high head runners, evident in table 5.1.

5.1.3 Model input values

The model inputs resulting from the three turbine designs, are presented below in table 5.1. The unstable low head runner is included and will also be simulated to demonstrate the challenges of applying this design in the model of the analysis.

	High head Francis	Medium head Francis	Low head Francis
α_{1R} [°]	9.674	12.29	25.47
σ [−]	0.3997	0.1057	-0.6835
ψ [−]	0.2856	0.9029	2.582
ξ [−]	1.267	1.859	3.234

Table 5.1: Model inputs from turbine design based on measured best efficiency point, using the design recipe by Hermod Brekke.

5.1.4 Design speed numbers

Since measured unit values at best point are used directly in this approach, calculating the speed number from equations 3.1 or 3.8 will give equal values, which are presented below in table 5.2.

	High head Francis	Medium head Francis	Low head Francis
$\Omega = \frac{\omega Q^{1/2}}{(2gH)^{3/4}}$ [−]	0.2238	0.4566	1.137

Table 5.2: Speed numbers resulting directly from measured best efficiency point.

5.2 Method 2: Direct tuning input parameters to measured best efficiency point

An alternative approach in order to obtain the necessary model inputs, is tuning them *directly* to the measurements' BEP. This can only be performed when experimental data are available, which implies that the method presented in this section is based on empiricism. To design a runner according to a procedure like Brekke's, has more of a first principles based approach. In fact, his recipe is a mix of first principles and empirical relations. For the objective of model analysis, tuning the inputs directly can obtain a better basis for comparing model to experimental data.

Tuning the input parameters to the experimental data was performed by the following:

1. Optimal guide vane angle α_{1R} is extracted directly from measured BEP.

   According to the turbine model equations, the characteristic coefficients at BEP can be calculated from the four model inputs, demonstrated in Appendix A2. Reversely, we may use measured gradients of flow and torque with respect to runner speed at BEP, to estimate the model input parameters σ and ψ:

2. $\sigma = -\, a_{13} = -\left.\dfrac{\partial q}{\partial \tilde{\omega}}\right|_{BEP}$

3. $\psi = -\, a_{23} = -\left.\dfrac{\partial t}{\partial \tilde{\omega}}\right|_{BEP}$

   The gradients must be estimated by a numerical scheme for the experimental data. The other independent variables must be kept constant in this point.

4. Finally: $\xi = (1 + \psi)\cos\alpha_{1R}$

5.2.1 Francis runner design

Even though redundant for the purpose of modelling, a complete runner design *can* be derived from these input values. It will not be unique, different turbines can be designed from the same values of α_{1R}, σ and ψ. They will however be geometrical similar, that is, equal speed number Ω and reduced velocity diagrams at best efficiency point. The resulting design is a matter of scaling by choosing rated design head H_R and number of pole pairs for the generator Z_p, the same scaling choices as for the procedure in section 5.1.

The chosen Z_p gives the synchronous speed of the unit. The outlet peripheral velocity at this speed is calculated from the definition of machine constant ψ:

$$\psi \;=\; \frac{u_{2R}^2}{\eta_R \; g \; H_R} \;\Longrightarrow\; u_{2R} \;=\; \sqrt{\psi \; \eta_R \; g \; H_R} \tag{5.1}$$

Where rated efficiency η_R is known as the experimental best point at which the gradients defining σ and ψ, was extracted. One should check that the resulting velocity is not too much larger than $\sqrt{2gH_R}$. This should be no problem as long as ψ times η_R is not much larger than 2. Both diameters can now be calculated, firstly D_2 from u_{2R} and ω_R, and secondly D_1 from the definition of geometrical constant σ:

$$\sigma \;=\; \frac{\omega_R^2}{8 \; g \; H_R} \; (D_1^2 - D_2^2) \;\Longrightarrow\; D_1 \;=\; \sqrt{D_2^2 + \sigma \; \frac{8 \; g \; H_R}{\omega_R^2}} \tag{5.2}$$

Inlet peripheral velocity u_{1R} can be calculated from D_1 and ω_R, and one should check its value with respect to $\sqrt{2gH_R}$.

Subsequently, the complete inlet and outlet velocity diagrams at design point can be calculated, but must be based on a few assumptions. First, design point implies zero outlet swirl, and second, 10% acceleration of the meridional velocity through the runner. The first assumption combined with Euler's equation gives:

$$\eta_R \;=\; \frac{u_{1R} \; c_{u1R} - 0}{g \; H_R} \;\Longrightarrow\; c_{u1R} \;=\; \eta_R \; \frac{g \; H_R}{u_{1R}} \tag{5.3}$$

Knowing already α_{1R} from measurements, the remaining velocity components and angles at the inlet can be calculated. The second assumption combined with the first enables calculation of the remaining velocity components and angles at the outlet. β_{2R} should generally be between $13 - 22^\circ$ [4] [26].

Finally, design discharge Q_R is calculated from meridional velocity and outlet cross section area, and inlet height B_1 is calculated from continuity. The speed number Ω is calculated from H_R, Q_R and ω_R. From table 5.4, these values of Ω are not equal to those obtained *directly* from the best point unit values in table 5.2. The two different tuning approaches do not achieve geometrical similar runners, as expected and intended.

Evidently, tuning the inputs directly as described above, avoids the challenges encountered when designing the very low head runner "from scratch", described in section 5.1.2. It is not given that the design resulting from direct tuning is good. After all, it is derived from very little information in only one operational point measured. Empirical relations should be used to check the resulting dimensions and velocity diagrams. That being said, excessive instability of the low head runner is avoided, since σ is extracted directly from the flow-speed measurements, and used when determining the diameter ratio.

5.2.2 Model input values

The model inputs extracted directly from the experimental data are presented below in table 5.3. The gradients are numerical estimates obtained using a central difference scheme on the interpolated data after filtering out smaller deviations from general curve shape. As explained in section 4.5, they were considered accurate with two decimal digits. Since the model inputs (except α_{1R}) are based on four of these gradients, also the inputs and speed numbers (resulting from the design based on the inputs), will be presented with two decimal place accuracy.

	High head Francis	Medium head Francis	Low head Francis
α_{1R} $[^\circ]$	10.52	15.99	27.15
σ $[-]$	0.69	0.46	0.01
ψ $[-]$	0.20	0.45	1.12
ξ $[-]$	1.18	1.39	1.89

Table 5.3: Model inputs tuned directly to measured best efficiency point.

5.2.3 Design speed numbers

The speed numbers resulted from equation 3.1 after performing a runner design as described in section 5.2.1, and are presented below in table 5.4.

	High head Francis	Medium head Francis	Low head Francis
$\Omega = \frac{\omega Q^{1/2}}{(2gH)^{3/4}}$ $[-]$	0.18	0.35	0.78

Table 5.4: Speed numbers resulting from turbine design based on model inputs tuned directly to measured best efficiency point.

Chapter 6

Model Analysis

In this chapter, the mathematical turbine model will be further analyzed.

Two main aspects will be addressed:

- Compare characteristic curves resulting from the governing equations to real turbine behavior extracted from the measurements. Model performance in predicting loss with respect to real efficiency data, will be emphasized.
- Compare characteristic coefficients calculated by the model to gradients numerically estimated from the measurements, at the point of best efficiency.

Keep in mind that the turbine model subjected to analysis consists of the following governing equations presented in Chapter 3, but repeated below for clarity (in per unit):

$$q = y \sqrt{h - \sigma \left(\tilde{\omega}^2 - 1\right)} \tag{6.1}$$

$$t = q \left(m_S - \psi \, \tilde{\omega}\right) \tag{6.2}$$

The mechanical power on the shaft is:

$$p = t \, \tilde{\omega} \tag{6.3}$$

And the hydraulic efficiency of the turbine:

$$\tilde{\eta} = \frac{t \, \tilde{\omega}}{q \, h} \tag{6.4}$$

So far, the model remains completely first principles based and independent from measurements. In this chapter, it will be analyzed before altering it with incipient efficiency $\eta_i(q)$. This improvement will be proposed and implemented in Chapter 7. Relevant simulations will then be repeated and compared to the measurements and to the model without the improvement, to check for better consistency.

6.1 Characteristic curves

In Theory section 3.4.4, the model's efficiency expression was investigated for its functional dependencies and ability to predict general hill shape in a performance diagram. This mathematical analysis of the equations serves as relevant background for this section. Further analysis of the nonlinear model will consist in evaluating how well characteristic curves are predicted. A number of different graphs can be plotted from the equations or extracted from the data. The characteristics considered most interesting were listed in section 4.4. The measurements (raw version without smoothing) and the two different model simulations will be presented in different plots next to each other. The main objective is to compare general curve shapes and the physics they entail. Given the simplicity of the model, and the unknown Rainpower turbine designs, exact agreement between model and experiments is not expected.

Since both runner design approaches (method 1 and 2) are modelled, we may assess how well they match the Rainpower runners, that is, to what extent geometrical similarity seem to have been achieved. It is important to distinguish between behaviour caused by the model inputs, and behaviour caused by the model itself. In other words, certain inaccuracies can be corrected for by correcting the inputs (tuning), while other inaccuracies will exist regardless of input accuracy, simply because of the nature of the governing equations.

Emphasis will be on *general* model behaviour irrespective of input accuracy, but the analysis will also try to distinguish between these two "types" of errors. Surely, since only two different designs and associated input parameters have been produced (per head classification high/ medium/ low), the basis for assessing this is limited. Since the thesis' objective is also model improvement, the analysis part is limited to the described scope.

The MATLAB scripts written to generate turbine characteristics are described in Appendix A3 and the executable m-files given as attachments to the thesis. Each script generates 14 different plots, only the 7 considered most relevant are included here (figures 1, 2, 3, 4, 6, 11 and 12 in the scripts). Adjustments in legends and axes were made to enhance comparison to the experimental data. For plotting the model equations, correct inputs must be specified. The code reads from the turbine design Excel sheet, but the user must specify which column to read according to what design to model. Plots (b) and (c) were produced by the same scripts reading different model inputs.

By accessing the model inputs (σ, α_{1R}, ψ and ξ) together with these scripts, the reader can easily generate the exact same model characteristic curves as presented in this section. Including all plots in full size would make the thesis extensively long, which is why they are presented in a compact 3x3 layout. Clearly, the measured characteristics in plots (a) are not possible nor intended for the reader to be able to generate on his or her own.

6.1.1 High head Francis turbine

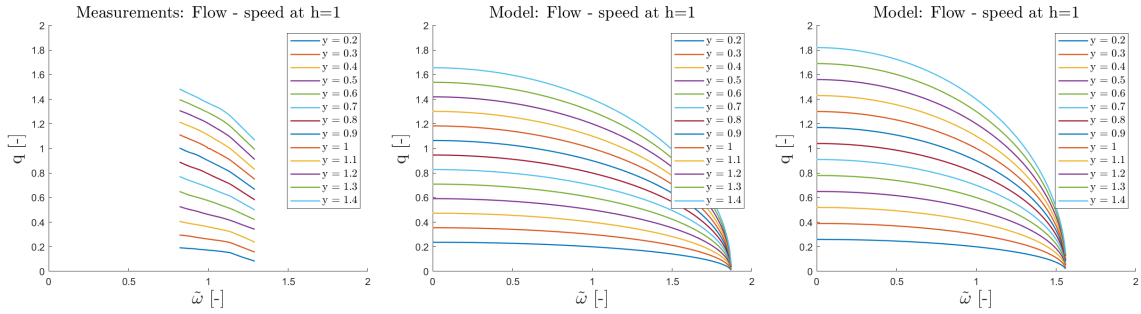

Figure 6.1: Flow - speed characteristics for different guide vane openings under constant rated head for high head turbine. From the left: (a) measurements, (b) simulations for model inputs by method 1 and (c) simulations for model inputs by method 2.

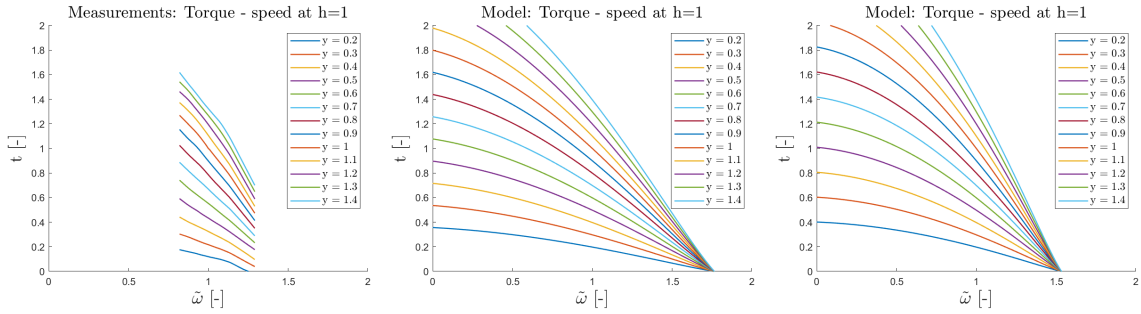

Figure 6.2: Torque - speed characteristics for different guide vane openings under constant rated head for high head turbine. From the left: (a) measurements, (b) simulations for model inputs by method 1 and (c) simulations for model inputs by method 2.

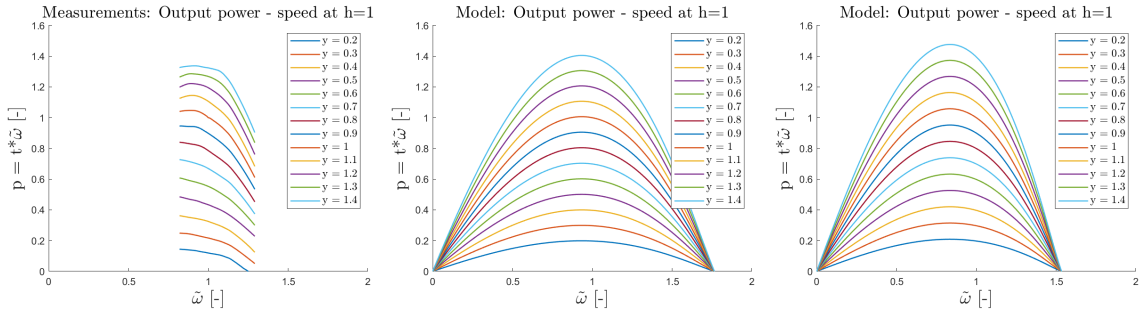

Figure 6.3: Mechanical power - speed characteristics for different guide vane openings under constant rated head for high head turbine. From the left: (a) measurements, (b) simulations for model inputs by method 1 and (c) simulations for model inputs by method 2.

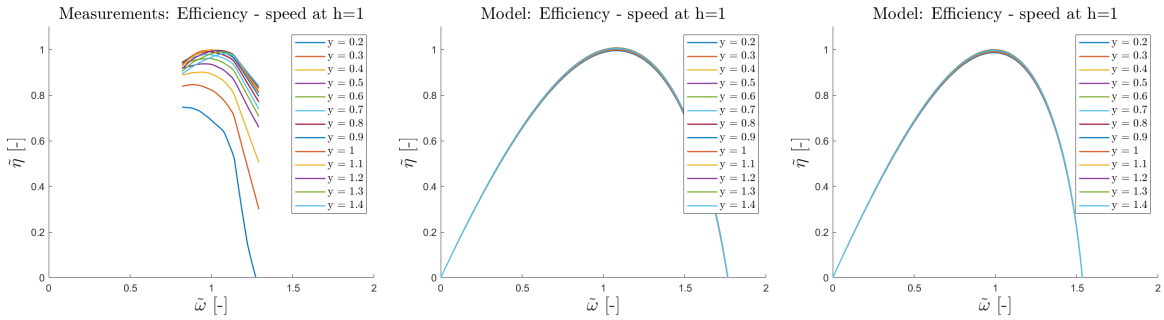

Figure 6.4: Efficiency - speed characteristics for different guide vane openings under constant rated head for high head turbine. From the left: (a) measurements, (b) simulations for model inputs by method 1 and (c) simulations for model inputs by method 2.

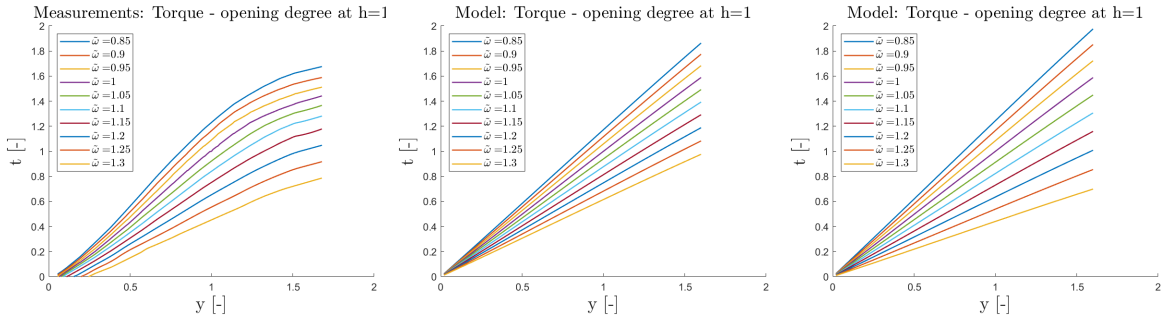

Figure 6.5: Torque - opening degree characteristics for different rotational speeds under constant rated head for high head turbine. From the left: (a) measurements, (b) simulations for model inputs by method 1 and (c) simulations for model inputs by method 2.

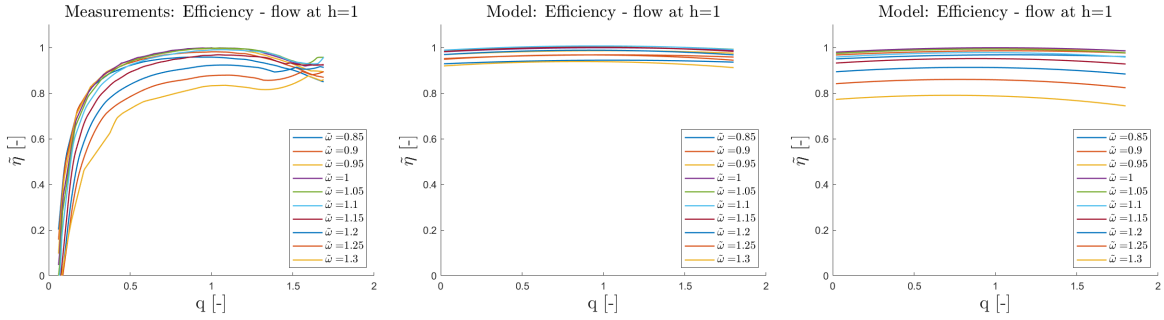

Figure 6.6: Efficiency - flow characteristics for different rotational speeds under constant rated head for high head turbine. From the left: (a) measurements, (b) simulations for model inputs by method 1 and (c) simulations for model inputs by method 2.

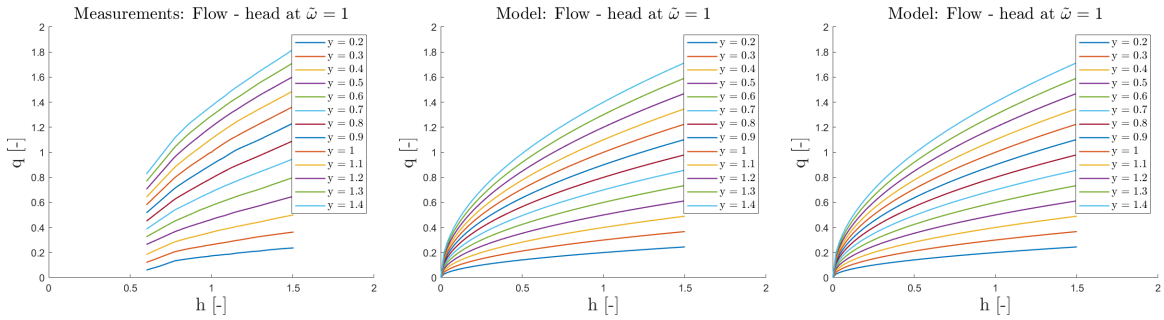

Figure 6.7: Flow - net head characteristics for different guide vane openings at constant rated speed for high head turbine. From the left: (a) measurements, (b) simulations for model inputs by method 1 and (c) simulations for model inputs by method 2.

Discussion

Firstly, comparing plots (b) and (c) of the above figures together with the high head columns of the input tables in Chapter 5, these two runner designs are somewhat similar. This is confirmed by their speed numbers, approximately 0.22 and 0.18 by method 1 and 2, respectively. Minor differences are observed, for example in predicted self-governing properties (steepness of flow-speed characteristics) and predicted runaway speeds (high speed values at which zero efficiency is reached). For the self-governing, the lower speed number yields the steeper characteristics, as expected. The σ values or diameter ratios confirm so. For the runaway speeds, experiments were not performed at high enough speeds to assess which design is more correct.

On a general note, the model seem to predict flow-speed characteristics quite well in the data range as both show stable (decreasing) behaviour. Curve shape of the torque-speed and the mechanical power-speed characteristics also seem to agree quite well with the measurements. Model behaviour outside of the given data range will generally not be commented on.

From figure 6.4, efficiency variation with runner speed is predicted well at rated guide vane opening ($y = 1$ curves), as expected from works like [7] [8], and from Theory section 3.4.4. Losses caused by the runner spinning too slow or too fast with respect to nominal speed are mostly kinetic energy not utilized by the turbine [8]. Since Euler's equation was the starting point for derivation, such losses are expected to be captured quite well. However, the model fails to predict variations in curve shape and especially levels for different constant openings. Different levels when taking different 2D sections along constant y lines in a Hill chart, are not accurately captured by the equations. The drawback appear to be independent from model input accuracy. Even so, general curve shape is not too bad.

To plot the mechanical torque from the water on the shaft against the guide vane open-

ing as in figure 6.5, is perhaps very intuitive, but can be a good check. Obviously, the torque should increase with the flow allowed through the runner when spinning at constant speed. The model captures this as a linear relation even though the measurements show less linearity, especially at higher openings. At α_1 higher than optimal, incidence loss will increase because of flow separation from the runner blades since the relative inlet flow does not match the (fixed) orientation of the runner blade at inlet. At $h = 1$ and constant $\tilde{\omega}$, according to equation 6.4, a deviation from linearity between t and q (which is linear with y at constant head and speed) will decrease the efficiency, that is, capture the presence of energy losses. The fact that the model predicts the torque to increase linearly with opening degree for all opening degrees (or with the flow for all flows, this plot is excluded but looks very similar), implies that it struggles to capture such incidence losses. This seem to exist independent of input values, it is a weakness of the torque equation.

The next figure 6.6 confirms this implication. The equations struggle to capture losses for constant speed, varying flow rate. The Euler efficiency of the model predicts unrealistically flat behaviour for all speeds included, and does not at all tend to zero for low and high flow, as expected. However, for the runner design by method 2 (plot (c)), there is a certain band around $q = 1$ where the efficiency is predicted quite well as the model manages to capture different levels for different speeds. Nevertheless, the prediction when flow tend to zero is highly erroneous regardless of model input. Accurate input values might improve the "level error" around $q = 1$, but the "shape error" for varying q appears to originate from the equations, as was demonstrated mathematically in section 3.4.4.

Finally, flow-head characteristics at $\tilde{\omega} = 1$ for different constant guide vane openings are included in figure 6.7. This plot might also seem intuitive - a larger available net pressure head will increase the discharge through the runner when spinning at constant speed. Nevertheless, the plot is included to check that the model predicts this behaviour correctly. At $\tilde{\omega} = 1$, the flow equation reduces to the valve equation; $q = y\sqrt{h}$. Since the effect of σ disappears, the flow equation becomes equal for all turbines regardless of geometry. This is why the two simulations of figure 6.7 are identical, and also equal to the lower head turbines, as will be illustrated. In fact, the flow equation at $\tilde{\omega} = 1$ predicts the same behaviour as for a Pelton turbine at any rotational speed. Comparing simulations to measurements of figure 6.7, flow-head characteristics for the high head turbine are modelled well by the valve equation in the given region.

6.1.2 Medium head Francis turbine

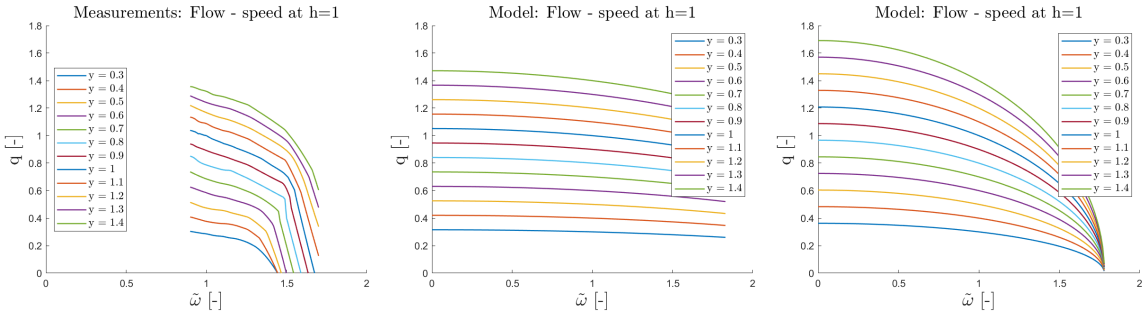

Figure 6.8: Flow - speed characteristics for different guide vane openings under constant rated head for medium head turbine. From the left: (a) measurements, (b) simulations for model inputs by method 1 and (c) simulations for model inputs by method 2.

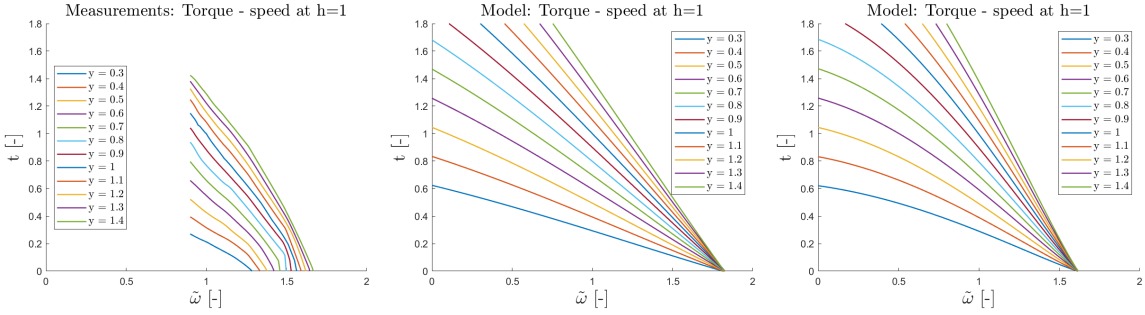

Figure 6.9: Torque - speed characteristics for different guide vane openings under constant rated head for medium head turbine. From the left: (a) measurements, (b) simulations for model inputs by method 1 and (c) simulations for model inputs by method 2.

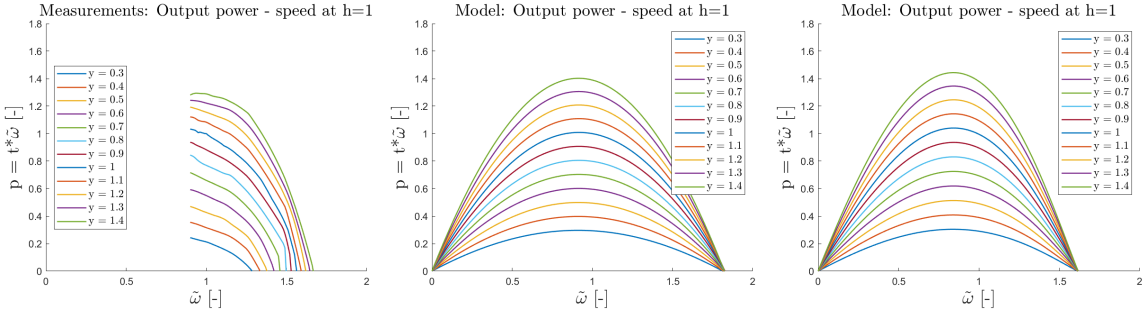

Figure 6.10: Mechanical power - speed characteristics for different guide vane openings under constant rated head for medium head turbine. From the left: (a) measurements, (b) simulations for model inputs by method 1 and (c) simulations for model inputs by method 2.

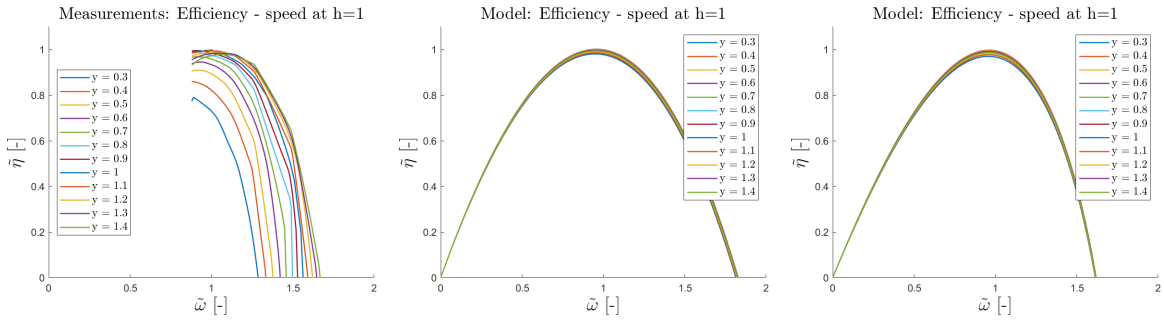

Figure 6.11: Efficiency - speed characteristics for different guide vane openings under constant rated head for medium head turbine. From the left: (a) measurements, (b) simulations for model inputs by method 1 and (c) simulations for model inputs by method 2.

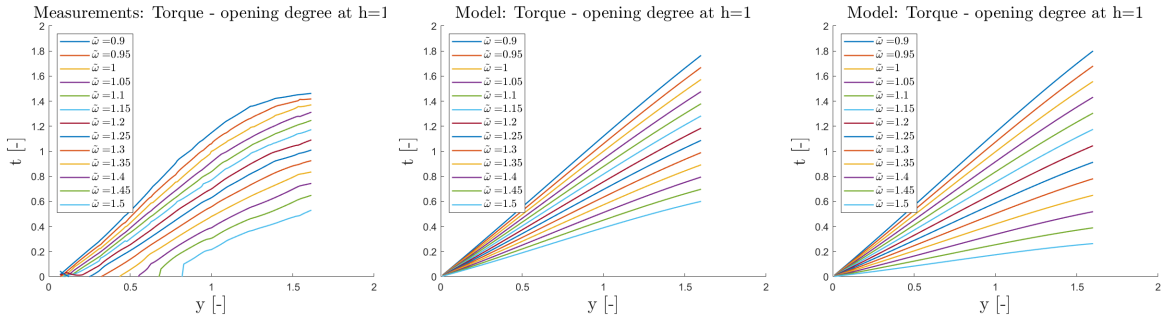

Figure 6.12: Torque - opening degree characteristics for different rotational speeds under constant rated head for medium head turbine. From the left: (a) measurements, (b) simulations for model inputs by method 1 and (c) simulations for model inputs by method 2.

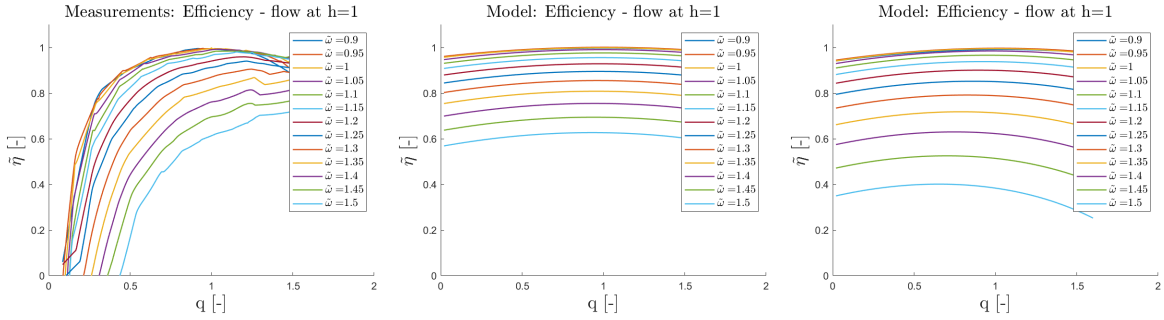

Figure 6.13: Efficiency - flow characteristics for different rotational speeds under constant rated head for medium head turbine. From the left: (a) measurements, (b) simulations for model inputs by method 1 and (c) simulations for model inputs by method 2.

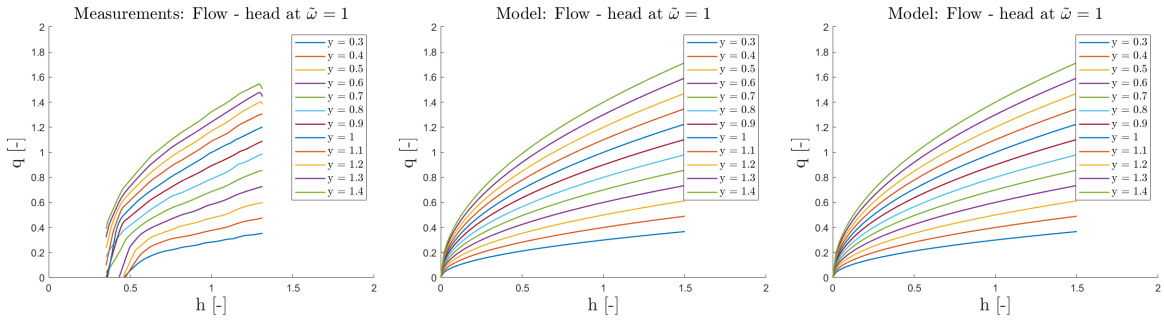

Figure 6.14: Flow - net head characteristics for different guide vane openings at constant rated speed for medium head turbine. From the left: (a) measurements, (b) simulations for model inputs by method 1 and (c) simulations for model inputs by method 2.

Discussion

Differences in the two runner wheel designs are more evident for the medium head turbine. This is concluded from comparing plots (b) and (c) of the above figures, together with the medium head columns of the input tables in Chapter 5.

Comparing simulations to measurements in figure 6.8, the gradient of the decreasing flow-speed characteristics especially at faster rotation, is highly under-predicted for the design by method 1. The model will predict decreasing curves as long as $\sigma > 0$ (which both designs got), but the steepness is determined by the *value* of σ. Recall its definition:

$$\sigma = \frac{\omega_R^2}{8\, g\, H_R} (D_1^2 - D_2^2) \tag{6.5}$$

Not only the difference in main diameters determines its value, also rated speed and design head. As explained in section 5.1, the design by method 1 was based on the assumption that measured BEP is also the design point of the model turbines, thus our design point as well. Comparing plots (a) and (b) in figure 6.8 indicates that this design is not geometrical similar to the turbine of the experiments. The larger σ and associated larger diameter ratio in the design by method 2 in plot (c), seem to match significantly better.

From figure 6.9, measured torque reaches zero (which gives zero power output and zero efficiency) at different speeds for different guide vane openings. In a Hill diagram, this is where the flow-speed characteristics reach the zero efficiency line (runaway curve). Also referred to as runaway speed, it is the speed for a certain guide vane opening which the runner will accelerate towards if the generator "falls off" the grid (load rejection). At this speed, efficiency is zero since all the power from water to runner is balancing losses in the unit - all the energy is transformed to heat. Runaway operation may damage the generator and one will usually begin to shut down the turbine after such incident [4].

From figures 6.9, 6.10 or 6.11, the model predicts *the same* runaway speed for all guide vane openings, which contradicts the measurements. This applies to both runner designs, indicating it to be a general weakness of the model and not an issue solved by correct inputs. This was also observed when modelling the high head Francis, but it was not measured to its runaway conditions. For the medium head design by method 1, the single modelled runaway speed is higher than those in the measurements, but for the design by method 2, the runaway speed for $y = 1$ agrees quite well with the experimental one.

Similar as for high head, the model seem to predict efficiency-speed shape well enough especially for $y = 1$, but different levels for different openings are not captured. This is evident when comparing simulations to measurements of figure 6.11. The real performance curves are slightly steeper than modelled, this indicates again that the second runner is more similar to the experiments. It also indicates that curve shape can - to a certain extent - be corrected for by correcting the inputs, but the level inaccuracy cannot.

Already discussed for the high head, hydraulic losses related to a sub-optimal inflow angles can result in deviation from linearity between torque and opening degree when spinning at constant speed. This is evident from the measurements of figure 6.12. Such losses are poorly captured by the model as is, which is why the torque equation predicts linear $t - y$ relation. Also observable in figure 6.12, is that the model fails to predict the different guide vane openings for different rotational speeds at which torque is zero. It simply predicts zero torque at zero opening for all speeds regardless model inputs, which contradicts the measurements. This is related to the shape of the runaway curve, and indicates that the model predicts it imprecisely.

From the efficiency-flow plots of figure 6.13, the model captures different levels, but the second design overpredicts the losses around $q = 1$, especially at higher speeds. Further, the shape of the curves does not at all match the experimental ones, nor the expectations that efficiency will tend to zero when flow tend to zero. These results confirm previous observations about this being a general model drawback.

Finally, comparing simulations to measurements in figure 6.14, the valve equation seems to model the flow-head relation quite well also for the medium head turbine, at least for about $h > 0.6$.

6.1.3 Low head Francis turbine

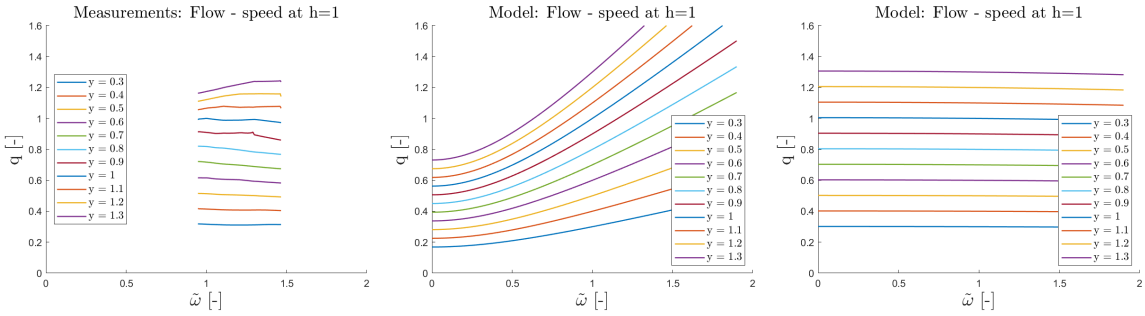

Figure 6.15: Flow - speed characteristics for different guide vane openings under constant rated head for low head turbine. From the left: (a) measurements, (b) simulations for model inputs by method 1 and (c) simulations for model inputs by method 2.

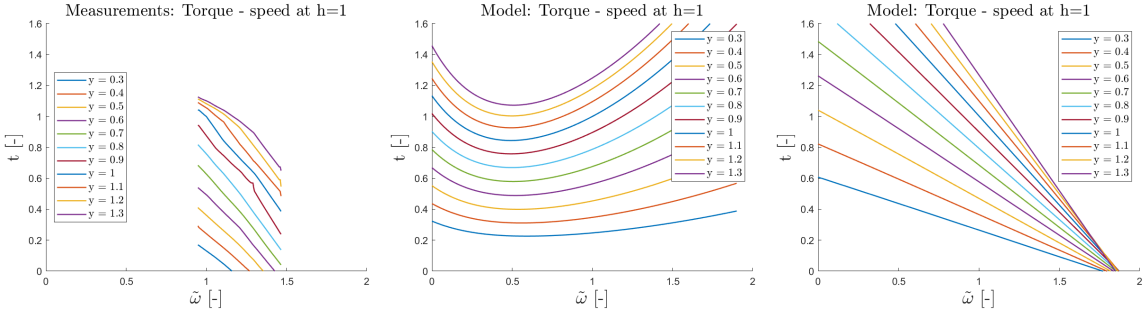

Figure 6.16: Torque - speed characteristics for different guide vane openings under constant rated head for low head turbine. From the left: (a) measurements, (b) simulations for model inputs by method 1 and (c) simulations for model inputs by method 2.

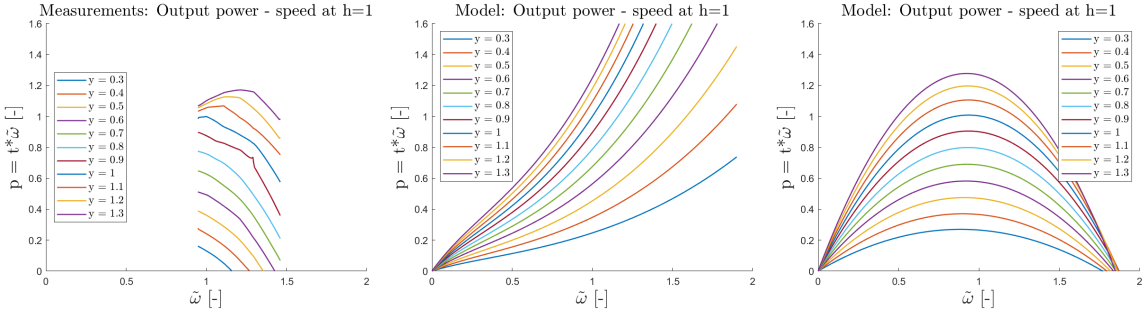

Figure 6.17: Mechanical power - speed characteristics for different guide vane openings under constant rated head for low head turbine. From the left: (a) measurements, (b) simulations for model inputs by method 1 and (c) simulations for model inputs by method 2.

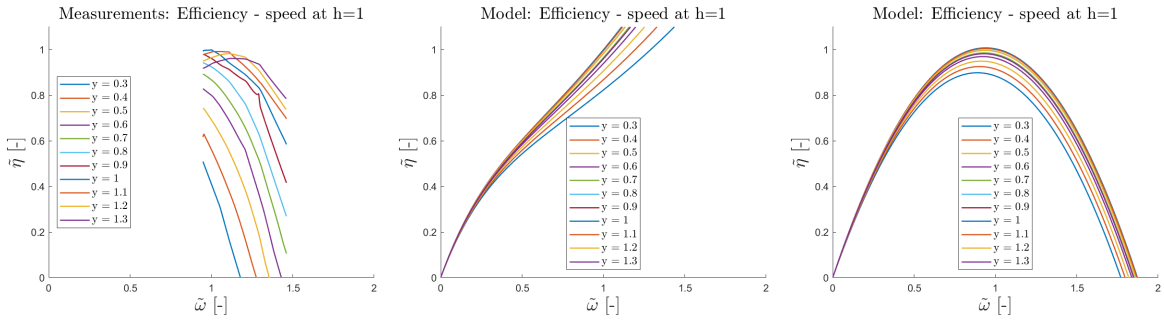

Figure 6.18: Efficiency - speed characteristics for different guide vane openings under constant rated head for low head turbine. From the left: (a) measurements, (b) simulations for model inputs by method 1 and (c) simulations for model inputs by method 2.

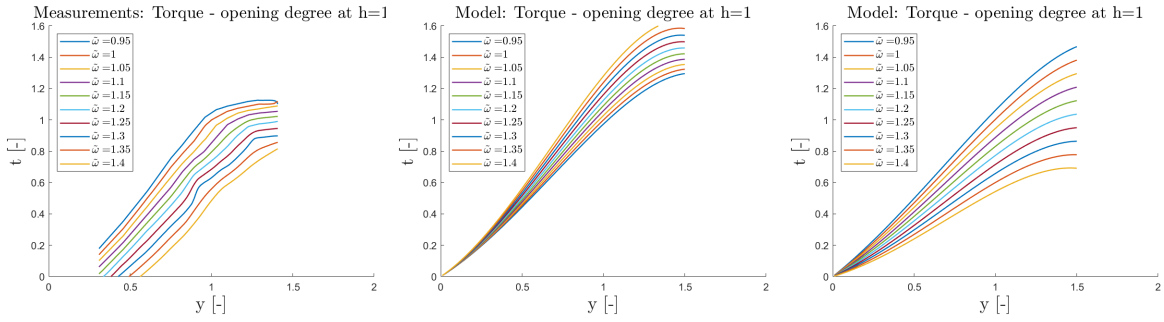

Figure 6.19: Torque - opening degree characteristics for different rotational speeds under constant rated head for low head turbine. From the left: (a) measurements, (b) simulations for model inputs by method 1 and (c) simulations for model inputs by method 2.

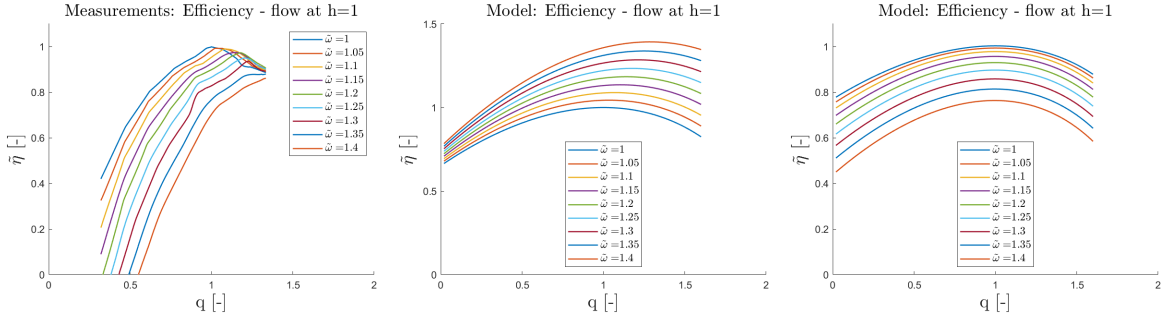

Figure 6.20: Efficiency - flow characteristics for different rotational speeds under constant rated head for low head turbine. From the left: (a) measurements, (b) simulations for model inputs by method 1 and (c) simulations for model inputs by method 2.

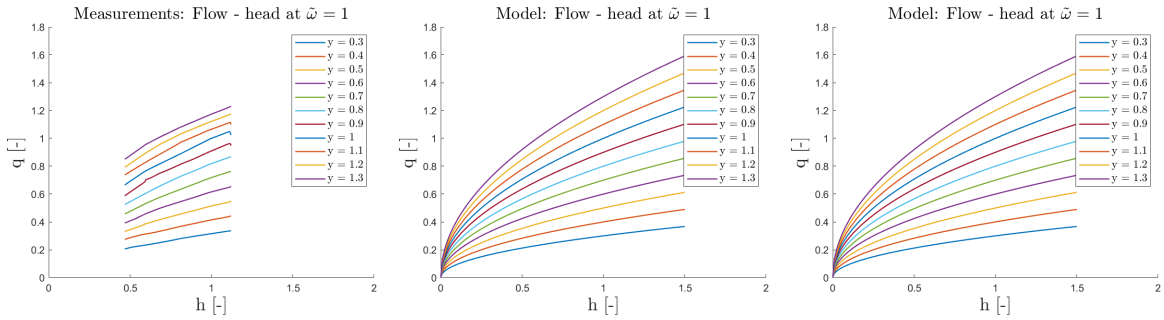

Figure 6.21: Flow - net head characteristics for different guide vane openings at constant rated speed for low head turbine. From the left: (a) measurements, (b) simulations for model inputs by method 1 and (c) simulations for model inputs by method 2.

Discussion

From Chapter 5, the two tuning approaches to the low head measurements resulted in very different turbine designs. As explained in section 5.1.2, setting design point equal to measured BEP and performing a design according Brekke's recipe (method 1) yielded highly unstable diameter ratio and large values for the machine constants, compared to what was achieved by direct tuning (method 2). The design differences are confirmed by comparing plots (b) and (c) of the above characteristics. The design by method 1 appears to be so off target that the equations sometimes struggle to predict reasonable behaviour, observed in several of the (b) plots above.

Looking at the measurements from Rainpower in figure 6.15, in the given range their low head runner is only slightly unstable for higher openings and quite unaffected or slightly stable for lower openings. The flatness of these curves indicates that the inlet and outlet diameters are almost equal. Clearly, the design by method 2 is much more similar to Rainpower's than the design by method 1. Based on this and the flow-speed curves of the higher head turbines, the model can be tuned by σ to predict quite accurate flow-speed characteristics.

From the torque equation 6.2, wrongly estimated machine constants ψ and ξ can cause large errors in predicting the torque. Since ξ is calculated from ψ and α_{1R}, ψ is the root of the issue. In reality, change of torque with increasing speed is always negative, the turbine will resist changes in rotational speed (negative feedback) [8]. If torque is wrong, consequential errors in predicting power and efficiency, will follow. The erroneous torque characteristics in plot (b) in figure 6.16 propagates to plots (b) of figures 6.17 and 6.18.

Comparing only (a) and (c) of figures 6.16, 6.17 and 6.18, clearly, model performance can be much better for a better design. Its weakness in predicting runaway speeds accurately, is present also for the low head Francis. The equations manage to capture different

efficiency levels for different guide vane openings slightly better for the low head turbine than they did for the higher head ones. This is evident comparing plots (a) and (c) in figure 6.18, and looking at the corresponding figures 6.4 and 6.11 for the high and medium head ones, respectively.

From figure 6.19, the torque-opening graphs are not too bad as they include some non-linearity for higher y. However, the curves for low to slightly past rated speed are quite overpredicting. As discussed, the model's struggle to predict runaway curve shape correctly, is seen as the torque reaching zero at exactly closed guide vane position (zero flow) for all speeds. This is simply a drawback to the torque equation.

Looking at figure 6.20 plot (b), the turbine p.u. efficiency exceeds 1.0 when $\tilde{\omega} > 1$. This confirms previous statements about the method 1 design being unreasonable. For the design in plot (c), the curvature is better than for the higher head turbines, yet still far from the sharp curvature of the experiments. The model still struggles to capture irreversible hydraulic losses for high and low flow conditions.

The model's efficiency-speed curves of figure 6.18 can be related to its efficiency-flow curves of figure 6.20. Since the curvature of the latter are more apparent for the low head turbine, this will appear as more distinguished levels for the former curves, compared to the higher head turbines. In fact, the "level error" in the $\tilde{\eta} - \tilde{\omega}$ plots, is the *same* problem as the "shape error" in the $\tilde{\eta} - q$ plots. Therefore, it is reasonable to believe that improving model accuracy of efficiency-flow will also improve model accuracy of efficiency-speed. This will be checked in Chapter 7.

Comparing figure 6.21 to 6.14 and 6.7, accuracy of the $q - h$ relation at $\tilde{\omega} = 1$ seem slightly better for the higher head than for the low head turbine. Even so, the general curve shape in figure 6.21 is still well captured in the given region.

6.2 Characteristic coefficients

Analysis of the linear turbine model will consist in analysing the characteristic coefficients a_{ij}, $i = 1, 2$, $j = 1, 2, 3$. The mathematical formulas presented in Appendix A2 are physically correct as they are simply partial derivations of the physics-based model equations. This does not guarantee that they are accurate for any turbine, and in this section, they will be compared to gradients from the experiments. It is difficult to extract from measurements the change in torque for changing guide vanes while flow is held constant, or vice versa. Therefore, the reformulations presented in equations 3.26 and 3.27 in the theory, are convenient when estimating a_{2j} from measurements.

The accuracy of the numerical gradients was achieved to about two decimal places by smoothing the data before applying a central difference scheme. In this way, smaller irregularities were eliminated from the general curve shape, and more precise gradients were obtained. Since the experimental coefficients will be presented with only two decimal digits, so will the analytically calculated ones and their absolute error with respect to the experimental.

The gradients of q, a_{11}, a_{12} and a_{13}, are independent of any η_i as it will be inserted into the torque equation. However the gradients of t, a_{21}, a_{22} and a_{23}, will change for linearization points other than best efficiency. At BEP, any η_i curve should be 1 and have first derivative 0, thus the coefficients are unaffected by the presence or choice of curve. As for the nonlinear model in section 6.1, the linear model will be analyzed without any incipient efficiency in this section, therefore, only the coefficients at BEP will be presented here.

From Appendix A2, the general functionalities for the q and t characteristics at an arbitrary linearization point are:

$$a_{1j} = f(h, y, \tilde{\omega}, \sigma), \qquad a_{2j} = g(q, y, \tilde{\omega}, \eta_i, \xi, \psi, \alpha_{1R}, \sigma) \tag{6.6}$$

Where h, q, y, $\tilde{\omega}$, η_i depends on the point of linearization, and ξ, ψ, α_{1R}, σ depends on turbine geometry and optimal design point (closely related). At linearization point equal to design point, the general functionality becomes:

$$a_{ij} = h(\xi, \psi, \alpha_{1R}, \sigma) \tag{6.7}$$

for all six coefficients. This is demonstrated mathematically in Appendix A2.

Since model input parameters were tuned by two different methods and the characteristic coefficients depend on these, obviously different values will result from these methods. Based on the improvement of the simulations observed for method 2 compared to method 1, one may assume that the second turbine design is more similar to Rainpower's model turbines than the first. For this reason, only coefficients from this approach will be presented and compared to measurements. Recall that method 2 was based on choosing σ and ψ equal to (negative) a_{13} and a_{23} experimental gradients at BEP. This is why the absolute error between model and experiments at this point is exactly zero.

6.2.1 High head Francis turbine

	a_{11}	a_{12}	a_{13}	a_{21}	a_{22}	a_{23}	Check: $a_{21} + a_{22}$
Measurements	0.87	1.01	-0.69	2.15	-1.17	-0.20	0.98
Mathematical model	0.50	1.00	-0.69	2.20	-1.20	-0.20	1.00
Absolute error	0.37	0.01	0	0.05	0.03	0	0.02

Table 6.1: Numerical gradients from measurements versus mathematical characteristic values from model for the high head turbine, both at BEP.

Discussion

The error in a_{11} can be observed in figure 6.7. The measurements have slightly steeper slope than the model predicts around $h = 1$ for the $y = 1$ graph. Apart from this, the model predicts characteristic values for a_{12}, a_{21} and a_{22} at BEP very well compared to the experimental gradients.

6.2.2 Medium head Francis turbine

	a_{11}	a_{12}	a_{13}	a_{21}	a_{22}	a_{23}	Check: $a_{21} + a_{22}$
Measurements	0.74	1.03	-0.46	2.40	-1.42	-0.45	0.98
Mathematical model	0.50	1.00	-0.46	2.44	-1.45	-0.45	0.99
Absolute error	0.24	0.03	0	0.04	0.03	0	0.01

Table 6.2: Numerical gradients from measurements versus mathematical characteristic values from model for the medium head turbine, both at BEP.

Discussion

These coefficients show similar agreement with measured gradients as was observed for the high head turbine. The slope of the flow-head relation at BEP, a_{11}, is still underpredicted by the model, but slightly less for this turbine.

6.2.3 Low head Francis turbine

	a_{11}	a_{12}	a_{13}	a_{21}	a_{22}	a_{23}	Check: $a_{21} + a_{22}$
Measurements	0.49	0.73	-0.01	3.20	-1.59	-1.12	1.61
Mathematical model	0.50	1.00	-0.01	3.13	-2.12	-1.12	1.01
Absolute error	0.01	0.27	0	0.07	0.53	0	0.60

Table 6.3: Numerical gradients from measurements versus mathematical characteristic values from model for the low head turbine, both at BEP.

Discussion

For the low head turbine, the model predicts flow-head slope at BEP much better than for the higher head ones, as a_{11} matches significantly better. Details like exact gradients are hard to detect simply be looking at the curves in section 6.1. From the graphs, the valve equation seemed to be a slightly better fit for high head than low head Francis, but the above a_{11} gradients suggests the opposite around the point of best efficiency.

The flow-opening degree slope in BEP, a_{12}, is overpredicted by the model, this was not an issue for the higher head turbines. The torque-opening degree slope, a_{22}, is overpredicted in absolute value as well. This cannot be observed from the experiments since torque versus opening degree for constant flow is nonphysical. As explained in section 4.5, a_{22} is calculated from flow-head, torque-head (constant opening and speed) and flow-opening, torque-opening (constant head and speed) gradients. The accuracy of a_{22} is therefore difficult to anticipate simply by looking at the curves. This goes for a_{21} as well, and is why the $a_{21} + a_{22}$ check is more ”physically applicable”. Collectively (added),

a_{21} and a_{22} shows how well the torque can be controlled by opening or closing the guide vanes [8]. For this low head turbine, the model underpredicts this relation with respect to the experiments. The response in torque for increasing the flow by opening the guide vanes is in fact steeper than predicted.

Clearly, the error in a_{12} and a_{22} is connected. The response in flow for increasing guide vane opening impacts the response in torque for increasing flow (by increasing guide vane opening). This comes from the model equations' functional dependencies.

Chapter 7

Model Improvement

In this chapter, improvement of how the model predicts the hydraulic efficiency is presented, implemented and discussed. The concept of this improvement has already been mentioned several times in the master thesis, and was briefly investigated in the project thesis. It is the incipient efficiency η_i function, intended to collectively include all the irreversible hydraulic losses that the governing equations do not include on their own.

Chapter 7 contains six main parts. In section 7.1, the concept including assumptions and restrictions will be presented. In section 7.2, different curve models are fitted to the experimental data and three specific curves proposed for the three turbines. Model implementation is demonstrated in sections 7.3 and 7.4 for these turbines. First, by presenting relevant characteristic curves, and second, by presenting the complete Hill diagrams. Following this, section 7.5 investigates the idea of generalizing η_i curves with respect to turbine speed number Ω. Finally, section 7.6 will provide a general discussion of the incipient efficiency improvement based on all the findings of this chapter.

7.1 Incipient efficiency concept

The idea is the same as in [5] [7] [8], but using experimental data as in the project thesis. A much larger quantity of data was accessed and several different curve fitting models investigated this time. The turbine model strive to capture real behaviour as precise as possible. Observed in Chapter 6, the efficiency is in general predicted better along the speed axis than along the flow axis. Energy losses caused by irreversible flow phenomena like friction, flow separation, turbulence, etc., were not fully captured. The incipient efficiency curve, $\eta_i(q)$, intend to correct the model efficiency for this inaccuracy, that is:

$$\tilde{\eta}_{model}\ \eta_i(q)\ \approx\ \tilde{\eta}_{real} \tag{7.1}$$

The p.u. model efficiency is as stated before; $\tilde{\eta} = t\tilde{\omega}/qh$, inserted for t and h. $\tilde{\eta}_{real}$ represents the efficiency measurements on site or in the lab at constant runner speed for varying flow, non-dimensionalized by the measured maximum efficiency.

The intention of this work is to develop new $\eta_i(q)$ curves more accurate than the second degree polynomial, $\eta_i(q) = q(2-q)$, proposed in [5], based on Rainpower's data. The curves should remain simple to implement, and can be applied in the model for turbines similar to those from the measurements.

7.1.1 Requirements

The proposed $\eta_i(q)$ curves should fulfill the following:

1. Be equal to 1 in $q = 1$
2. Have first derivative in (1,1) equal to 0 (maximum at BEP)
3. Be differentiable in its entire q range (have continuous first derivatives)
4. Adapt to the data points smoothly with as little oscillatory behaviour as possible
5. Monotonically decrease for $q < 1$ and $q > 1$

The first two requirements are self-explanatory given the definition of BEP and per unit values. The first derivative of $\eta_i(q)$ appears in a_{21} of the linear model, thus needs to be defined for all working points one may want to linearize around. Number 4. and 5. are based on expected behaviour from real efficiency curves.

7.1.2 Influence of varying speed

The η_i implementation as in [5] [7] [8] is independent of speed and depends solely on flow rate. Efficiency-speed curves at different openings might not be symmetric around rated speed, or the shape of efficiency-flow curves at different speeds can be very dissimilar. Such tendencies are seen as asymmetry of contour lines in the performance diagram, such that implementing the same $\eta_i(q)$ curve for different speeds might be inaccurate.

The measurement plots in section 6.1 featuring efficiency versus flow at constant rated head for different speeds, figures 6.6, 6.13 and 6.20, can indicate the accuracy of implementing only one curve. The three sets of turbine data were given for quite different speed ranges, i.e. the width of the region in a Hill chart in which performance data were given, differs. Taking this into account when studying the plots, one may argue that for each turbine the general *shape* of the efficiency curves are somewhat similar in a certain band around rated speed.

To not over-complicate the implementation in this work, only one $\eta_i(q)$ curve will be constructed for each turbine. The curve will be fitted to the $\tilde{\eta} - q$ data corresponding to rated speed and head, i.e. $N_{11}/N_{11R} = \tilde{\omega}/\sqrt{h} = 1.0$. These are the $\tilde{\omega} = 1$ curves of the measurements in figures 6.6, 6.13 and 6.20. The accuracy of this simplification will be further discussed after implementation into the model, mainly in section 7.6.

7.1.3 Flow rate validity range

The $\eta_i(q)$ curve needs only to be defined between the low flow value corresponding to zero efficiency and some high flow value limiting the area of interest. The data for curve fitting are extracted at $\tilde{\omega} = 1$, assuming $h = 1$. At constant speed and head, flow is increased by opening the guide vanes. There is a physical limit to the flow area defined by the guide vanes. When in their maximum opening position, from continuity, further flow rate increase must be due to flow velocity in the meridional direction increasing further. However, such acceleration would require a larger pressure head in order to "push" the water through the runner at a faster rate. From this reasoning, at constant speed and head, the upper flow is restricted by the upper guide vane opening.

The parabola from [5], $q(2 - q)$, is 0 in $q = 0$ and $q = 2$. Multiplied into the torque equation, also output power and turbine efficiency will become 0 at these p.u. flow values, irrespective of runner speed.

To study a couple of Francis Hill diagrams can be useful, for example those depicting low and high specific speeds in figure 3.2, presented again in figure 7.1 below [27]. The 2D section of relevance going through BEP is drawn as a red line in the diagram.

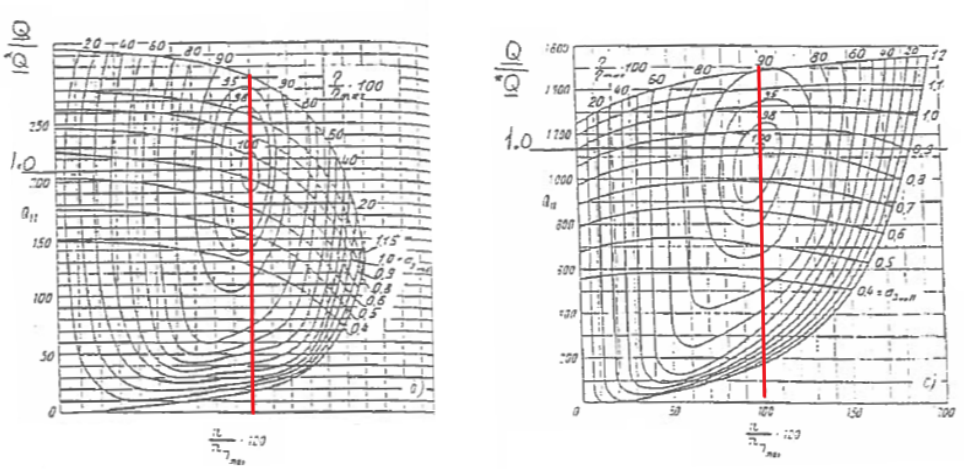

Figure 7.1: (a) Low specific speed (high head) and (b) high specific speed (low head) Francis runner's typical Hill charts.

First of all, it appears that zero efficiency (the runaway curve) is reached for a flow rate slightly higher than zero, exactly how much depending on turbine type and the speed of investigation. The gradient of the runaway curve varies, but the steeper, the larger error to assume $\eta = 0$ in $q = 0$ also at higher speeds.

Second, the diagrams are only constructed up to a certain high flow value associated with a certain high guide vane opening (perhaps the maximum setting), and do not reach any line of zero efficiency at some high flow value. We cannot know exactly what behaviour to expect for discharges larger than restricted by the maximum guide vane opening (neither is it particularly interesting). We can only assume that the efficiency would continue to decrease, i.e. the hill slope to proceed.

In theory, an $\eta_i(q)$ curve can only be valid in the flow range it was based on. Approximate data ranges given in p.u. flow q were for the high head turbine 0.06 - 1.57, for the medium head 0.08 - 1.48 and for the low head 0.32 - 1.20. Measurements were not given all the way down to zero efficiency for the low head, which is a weakness of these data.

To summarize, we will expect the $\eta_i(q)$ curve to reach zero for some low flow value. Also for the low head turbine without knowing exactly at what value this should happen. Based on figure 7.1, we do however expect it to happen at a slightly higher flow than for the high head one. Since we expect the efficiency curve slope to proceed for flow rates past the upper limit of the data, we will require the $\eta_i(q)$ curve to do the same. To make sure it continues to decrease monotonically for flow rates slightly past the upper flow limit, an extra data point, for example (2,0), can be added before curve fitting. This point of zero is not justified by physics, it is included solely to achieve good curve behaviour. When included, it will be stated clearly.

7.1.4 Curve fitting models

In the project thesis, prototype efficiency data were given at synchronous speed, there were only nine data points and no interpolation were performed. Cubic splines in each interval were created in MATLAB, resulting in a smooth curve. Although easily evaluated using computer programs, analytically, it is easier to handle only one, or perhaps two, curves on the entire flow range. This was the main reason for looking into the following types, which are all applicable for curve fitting in MATLAB:

1. Polynomial function of degree n

$$y(x) = p_1\, x^n + p_2\, x^{n-1} + p_3\, x^{n-2} + ... + p_n\, x + p_{n+1} \tag{7.2}$$

2. Fourier series function of n harmonics

$$y(x) = a_0 + a_1 \cos(\omega_0 x) + b_1 \sin(\omega_0 x) + ... + a_n \cos(n\omega_0 x) + b_n \sin(n\omega_0 x) \tag{7.3}$$

3. Two-term exponential functions, one for $x \leq 1$ and one for $x \geq 1$

$$y(x) = a\, e^{bx} + c\, e^{dx} \tag{7.4}$$

4. Two-term power functions, one for $x \leq 1$ and one for $x \geq 1$

$$y(x) = a\, x^b + c \tag{7.5}$$

Restricted by MATLAB's library models for curve fitting, a polynomial could at most have degree 9, and a Fourier series could at most have 8 number of harmonics. For the last two options, it is required that the two functions meet in (1,1) and for their first derivative to be equal or close to 0 in that point. The latter goes for all models.

Different curve fittings were performed using the inbuilt MATLAB function *fit()*, specifying model type and optionally using weights to emphasize the best point (1,1). There was no inbuilt functionality to guarantee that the constraints $f(1) = 1$, $f'(1) = 0$ are met when fitting to the data. In stead, the data point (1,1) was heavily weighted, and the first derivative of the resulting curve in this point was checked afterwards. In addition for the curve to be continuous with continuous first derivatives, we want the curve fitting to disregard minor irregularities in the data. The inbuilt MATLAB function *smooth()* was used to filter out smaller discrepancies before curve fitting.

The polynomial or Fourier series can have various degree of oscillatory behaviour. Testing up to the respective limits, increasing the degree for the polynomial or the number of harmonics for the Fourier, tend to increase accuracy and reduce oscillatory behaviour. Even so, there is a trade-off between curve complexity and desired accuracy. By complexity is meant the number of terms and corresponding coefficients (for Fourier series also the fundamental frequency ω_0) which are necessary to determine.

The scripts testing these curve fitting models are described in Appendix A3 and the executable m-files given as attachments to the thesis.

7.2 Curve proposals

In this section, the fitted curves will be discussed with respect to each other, and to the following options:

1. No model modification (equivalent to $\eta_i = 1$), presented already in section 6.1

2. The parabola in [5]: $\eta_i(q) = q(2 - q)$

From section 7.1, there are several criteria for evaluating and not necessarily only one best fitted curve for each data set. Discussion of curve options for the different turbines will be included, before stating final choice including function coefficients at the end of each section.

In order for η_i to *correct* the model with respect to the measurements, the curve fitting should be based on measured efficiency divided by model efficiency (both in p.u.). Even though model efficiency turned out quite flat for all designs, it must be clarified that the inputs by tuning method 2 were used for this purpose. These turbine designs were more similar to the experiments, and caused no issues like we saw for the low head runner by method 1. From this point on, only the designs by method 2 are given as model inputs.

7.2.1 High head Francis turbine

Efficiency measurements from the high head turbine were quite flat for a wide flow range, with a steep decrease towards zero efficiency at very low flow. This made the model efficiency surprisingly accurate in a certain band around $q = 1$. Implementing $q(2-q)$ would correct for its lack of reaching zero efficiency, but for a certain flow range, may give a larger overprediction of hydraulic losses than the underprediction occurring without η_i. The parabola is simply too steep compared to the measurements.

Obtaining curves with first derivative close to zero in (1,1) was not a problem for this turbine because of the flatness around BEP. All four proposed curve types could easily achieve derivative in absolute value less than 0.1. Fitting by a polynomial or Fourier series can have some oscillatory behaviour. Trial-and-error resulted in minimum oscillations achieved by choosing degree 8 or higher for the polynomial, or number of harmonics 5 or higher for the Fourier series.

Both a higher degree polynomial (like $n = 8$) or a Fourier series of a larger number of harmonics (like $n = 6$) were very good fits in the flow range of the data. Curve behaviour for even higher q varied. To demonstrate, below are presented curve fits by 7th, 8th and 9th degree polynomials, respectively, including their behaviour outside the data range.

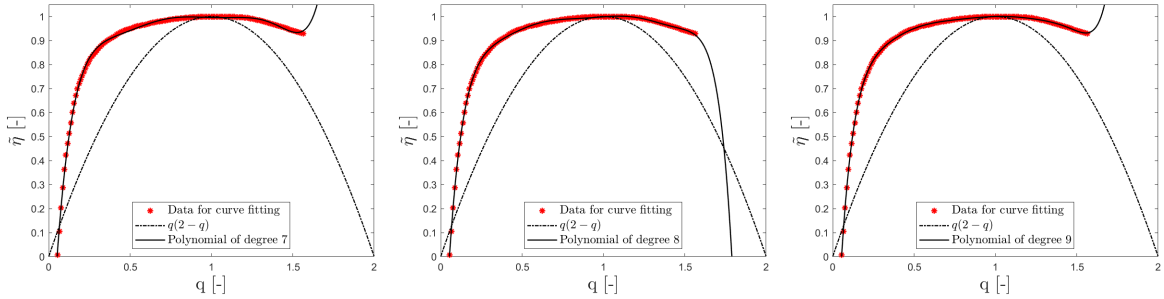

Figure 7.2: Polynomial curve fittings for the high head turbine having (a) degree 7, (b) degree 8 and (c) degree 9. Curves (a) and (c) struggle to predict reasonable behaviour for high flow.

Since we require the curve to decrease monotonically for flow rates slightly past the upper limit of the data, the 8th degree polynomial must be chosen. Alternatively, to add the data point (2,0) can remarkably improve the 9th degree polynomial:

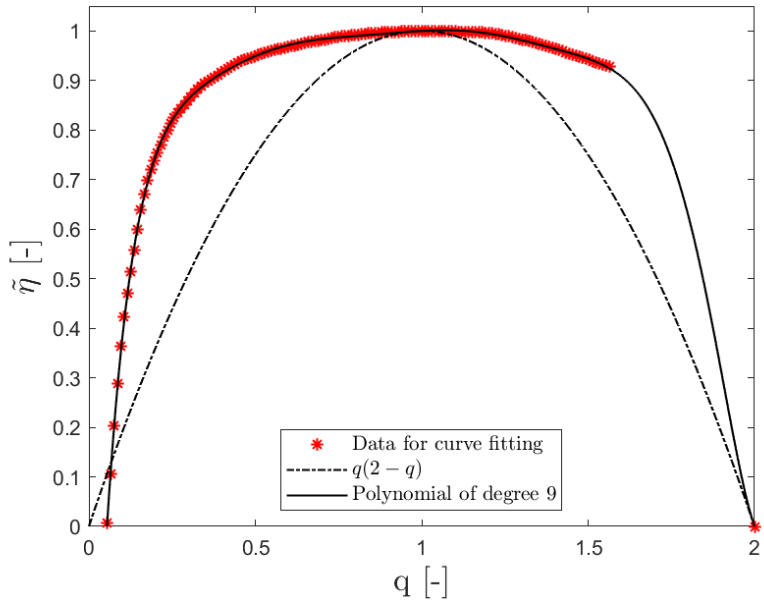

Figure 7.3: Polynomial curve fitting for the high head turbine having degree 9. The extra data point of (2,0) is included to improve curve behaviour outside the data flow range.

Below are presented curve fits by Fourier series of 5, 6 and 7 harmonics, respectively, including their behaviour outside the data range.

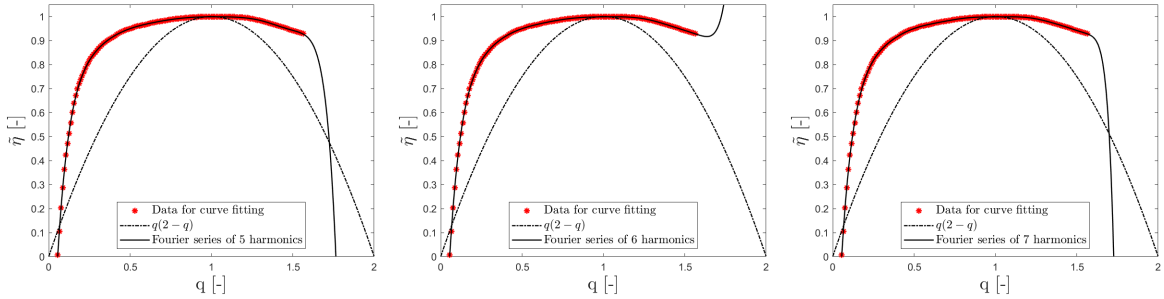

Figure 7.4: Fourier series curve fittings for the high head turbine having (a) 5 harmonics, (b) 6 harmonics and (c) 7 harmonics. Curve (b) struggles to predict reasonable behaviour for high flow.

The Fourier series of 5 and 7 harmonics are basically identical and decrease rapidly outside the data range. For simplicity, the 5 harmonics is preferred. Adding the extra data point (2,0) will improve these Fourier curves, exemplified below for the 6 harmonics one:

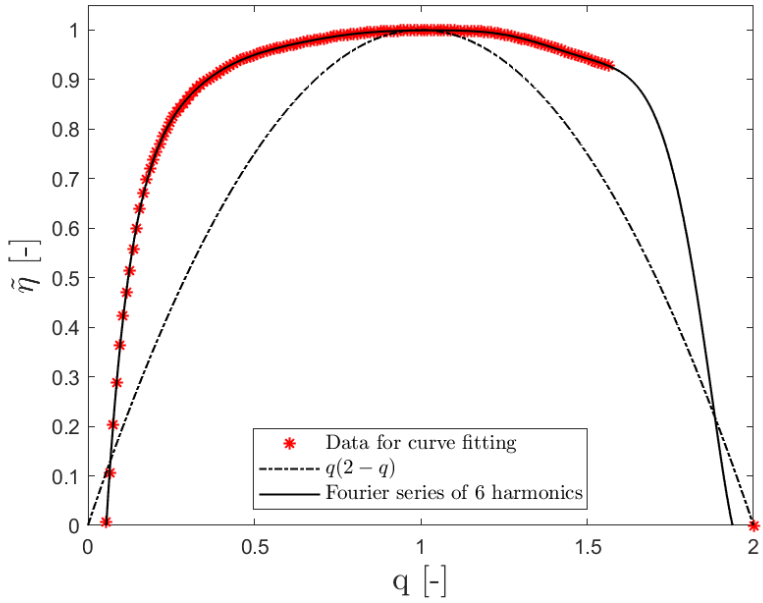

Figure 7.5: Fourier series curve fitting for the high head turbine having 6 harmonics. The extra data point of (2,0) is included to improve curve behaviour outside the data flow range.

The power function for $q \leq 1$ resulted in slightly overprediction of loss, while the exponential function had both a range of slightly over- and underprediction in the same region, observed in figure 7.6. The power and exponential functions for $q \geq 1$ were

similar and both good fits in the valid high flow region. Outside, their decrease towards zero efficiency is quite slow.

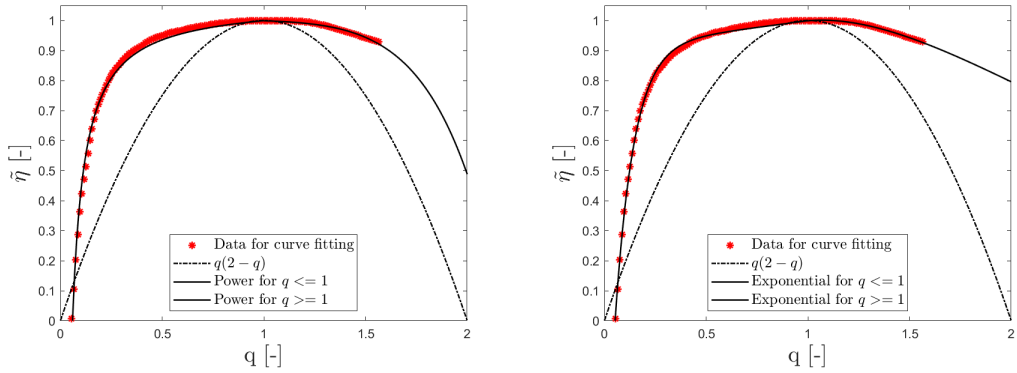

Figure 7.6: (a) Power and (b) exponential curve fittings for the high head turbine.

Also for these, the additional data point (2,0) can change the $q \geq 1$ curve behaviour.

Both the polynomial in figure 7.3 and the Fourier series in figure 7.5, are considered very good options for the high head turbine. The two-term power or exponential functions in figure 7.6 are considered quite good options as well.

For the purpose of demonstrating the model improvement, a choice was made. The 9th degree polynomial is on the following form:

$$\eta_i(q) = p_1 q^9 + p_2 q^8 + p_3 q^7 + p_4 q^6 + p_5 q^5 + p_6 q^4 + p_7 q^3 + p_8 q^2 + p_9 q + p_{10} \quad (7.6)$$

Having the following coefficients:

p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}
5.6718	-52.528	207.27	-456.12	615.25	-526.61	286.34	-96.278	18.782	-0.7765

Table 7.1: Coefficients for a 9th degree polynomial fitted to the high head turbine measurements.

7.2.2 Medium head Francis turbine

The medium head turbine for $\tilde{\omega} = 1$ also has a wide flow region of relatively flat efficiency data, and a rapid decrease towards zero for low flow. After filtering out minor irregularities, the general curve shape is more similar to the curve of the high head rather than the low head runner. From their speed numbers and geometrical similarity, this resemblance is expected.

Tendencies of oscillating behaviour for the polynomials were slightly more evident for the medium head turbine. Similar trends as for the high head were observed when testing: Higher degrees will reduce oscillations, and odd degrees will give unreasonable curve behaviour for flow rates higher than the given range. For Fourier series curves having a larger number of harmonics, less oscillation was achieved. When testing different number of harmonics, certain numbers gave unreasonable behaviour for higher flow rates. A Fourier series of 5 harmonics seem to be a good fit with minimum oscillatory behaviour, and it decrease monotonically (however very fast) for flow past the upper limit of the data:

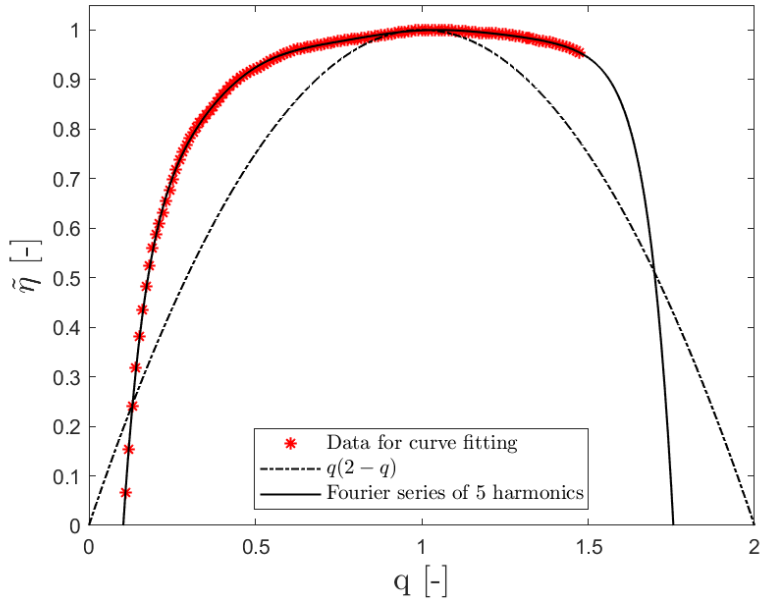

Figure 7.7: Fourier series curve fitting for the medium head turbine having 5 harmonics.

In general for the different curve models, adding the extra data point (2,0) did not improve curve behaviour as much as it did for the high head turbine. The only improvement worth including was for a 9th degree polynomial:

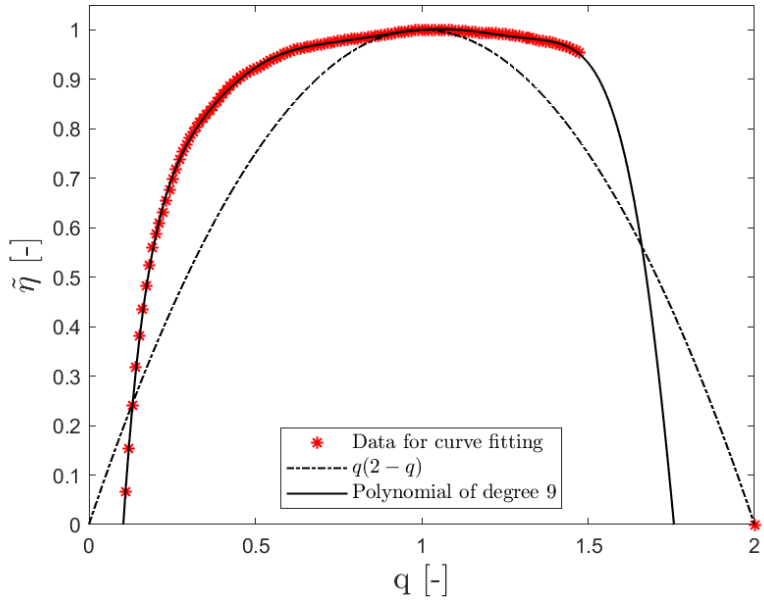

Figure 7.8: Polynomial curve fitting for the medium head turbine having degree 9. The extra data point of (2,0) is included to improve curve behaviour outside the data flow range.

Strengths and weaknesses of having two different exponential or power functions are similar as observed for the high head. For $q \geq 1$, these curves continue to decrease monotonically also past the upper flow of the data:

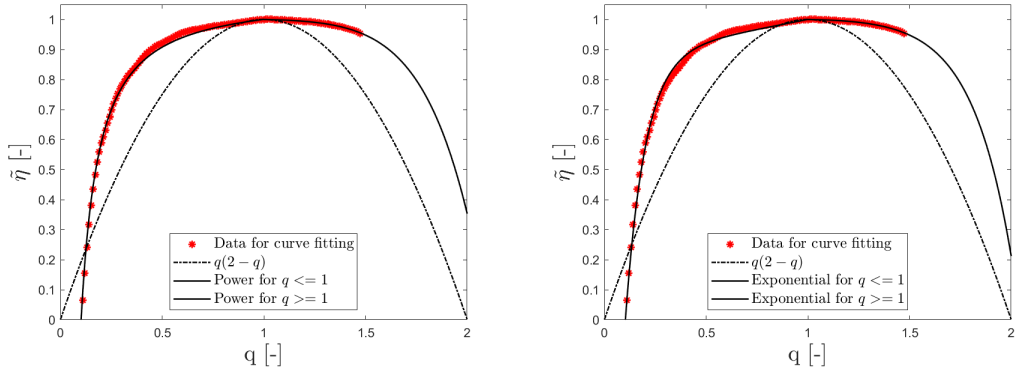

Figure 7.9: (a) Power and (b) exponential curve fittings for the medium head turbine.

All of the above presented curves in figures 7.7, 7.8 and 7.9, are considered good options for the medium head turbine.

Choosing the 5 harmonics Fourier series, the function is on the following form:

$$\eta_i(q) = a_0 + a_1\cos(\omega_0 q) + b_1\sin(\omega_0 q) + a_2\cos(2\omega_0 q) + b_2\sin(2\omega_0 q) + a_3\cos(3\omega_0 q) + b_3\sin(3\omega_0 q) + a_4\cos(4\omega_0 q) + b_4\sin(4\omega_0 q) + a_5\cos(5\omega_0 q) + b_5\sin(5\omega_0 q) \tag{7.7}$$

Having the following coefficients and fundamental frequency:

ω_0	a_0	a_1	b_1	a_2	b_2
0.8655	-22977	26587	27975	1165.1	-22545
a_3	b_3	a_4	b_4	a_5	b_5
-6702.8	5712.9	2058.5	227.92	-133.35	-179.00

Table 7.2: Coefficients for a 5 harmonics Fourier series fitted to the medium head turbine measurements.

7.2.3 Low head Francis turbine

The steep efficiency data with a sharper peak was a little more challenging to fit a curve, especially since we want the first derivative close to zero at BEP. Additionally, the flow range of the given data was very limited for this turbine. Both a higher degree polynomial or a Fourier series of a larger number of harmonics were good fits in the considered range, but curve behaviour outside varied a lot. Below are presented curve fits by 7th, 8th and 9th degree polynomials, respectively, including their behaviour outside the data range.

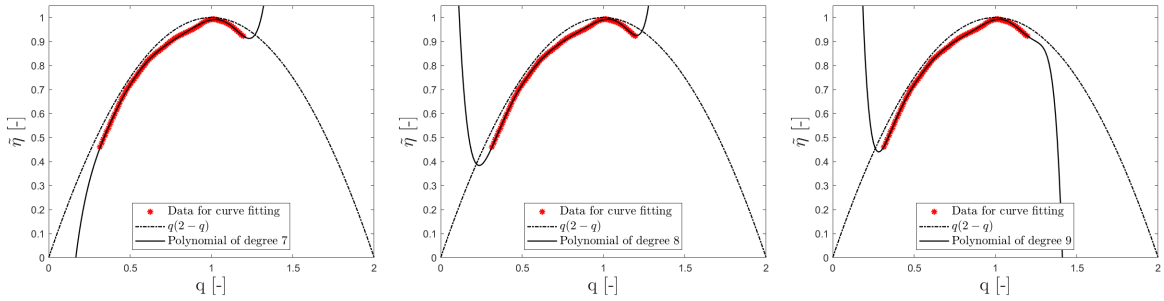

Figure 7.10: Polynomial curve fittings for the low head turbine having (a) degree 7, (b) degree 8 and (c) degree 9. They all struggle to predict reasonable behaviour outside the data range.

None of the above polynomials fulfill the requirements of reaching zero for low flow and monotonically decrease for flow rates higher than the upper limit of the data.

Below are presented curve fits by Fourier series of 5, 6 and 7 harmonics, respectively, including their behaviour outside the data range.

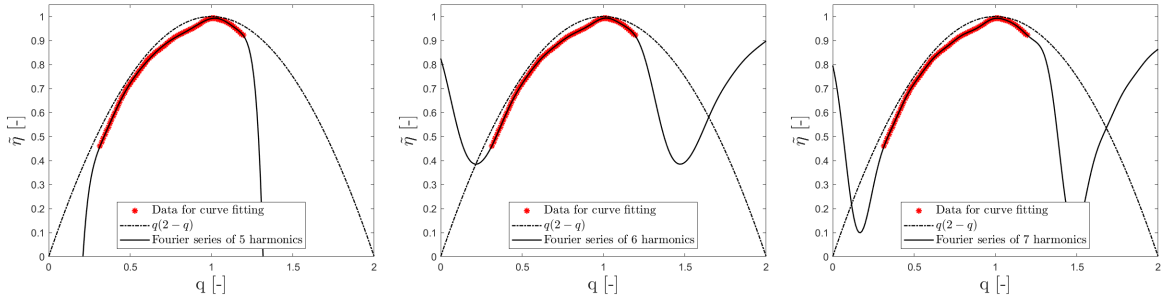

Figure 7.11: Fourier series curve fittings for the low head turbine having (a) 5 harmonics, (b) 6 harmonics and (c) 7 harmonics. Curves (b) and (c) struggle to predict reasonable behaviour outside the data range.

The Fourier series of 7 harmonics works better than the 6 harmonics, but since we require the curve to reach zero for some low flow, the 5 harmonics curve must be chosen.

Because of the shape of the experimental data, it was difficult to improve curve behaviour outside of given flow ranges simply by adding (2,0), or any other data point. This was evident from testing different curve models. The curves still have the tendency of ”turning” at locations where we expect the real efficiency to monotonically decrease, similar to the behaviour of the above graphs.

Because of the sharp and asymmetrical BEP, it was challenging to make two curves meet in (1,1) with first derivatives equal to zero, while still providing good fits for the remaining data. The advantage of exponential or power functions is that they both monotonically decrease outside the data range. The exponential was considered a better fit:

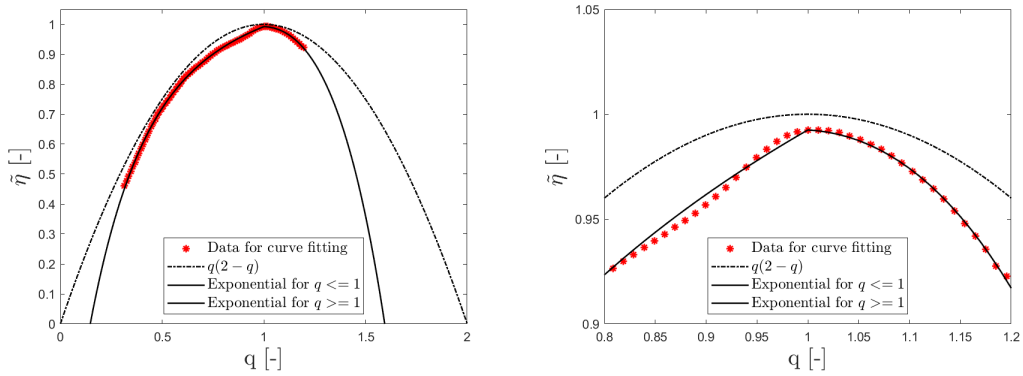

Figure 7.12: Exponential curve fitting for the low head turbine. The right figure provides a zoom-in of the best efficiency peak from the left figure.

For the low head turbine, the model efficiency was only slightly curved, while the real efficiency decreases rapidly at off-design conditions. Also included in the above figures is the parabola from [5], $q(2-q)$, which in fact is more accurate for this turbine than it was for the higher head ones. The low head runner is the only case where the parabola actually underpredicts the losses (overpredicts the efficiency) slightly. For both high and medium head runners, the parabola lies *below* the data for most of the flow range. A curve fitted to measurements is the most accurate option, but when such data are unavailable, $q(2-q)$ seem to agree quite well for a low head turbine like this one.

Based on this observation, polynomials of lower degree were investigated for fitting to these data. Below are presented curve fits by 3rd, 4th and 5th degree polynomials, respectively, including their behaviour outside the data range.

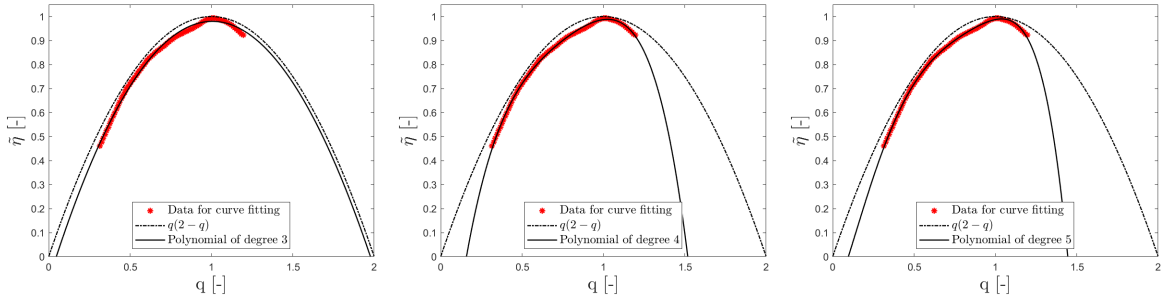

Figure 7.13: Polynomial curve fittings for the low head turbine having (a) degree 3, (b) degree 4 and (c) degree 5. They are all remarkably better behaved outside the data range.

The 4th degree polynomial in plot (b) above is considered a surprisingly good fit, especially considering how simple it is. Also the 5 harmonics Fourier series or the two-term exponential functions are considered good alternatives for the low head turbine.

Choosing the 4th degree polynomial, the function is on the following form:

$$\eta_i(q) = p_1 q^4 + p_2 q^3 + p_3 q^2 + p_4 q + p_5 \tag{7.8}$$

Having the following coefficients:

p_1	p_2	p_3	p_4	p_5
-2.9752	9.0639	-10.912	6.6182	-0.8079

Table 7.3: Coefficients for a 4th degree polynomial fitted to the low head turbine measurements.

7.3 Implementation into the model

The main purpose of $\eta_i(q)$ is to improve model accuracy in capturing irreversible hydraulic losses along the q-axis. As first performed in [5], implementation is simply to multiply the efficiency with the incipient efficiency expression. As performed in [7] [8], this is ensured by inserting it into the torque equation:

$$t \;=\; \eta_i(q)\; q\; (m_S - \psi\; \tilde{\omega}) \tag{7.9}$$

Consequently, it will appear in the nominator of the efficiency expression, as desired:

$$\tilde{\eta} \;=\; \frac{\eta_i(q)\; t\; \tilde{\omega}}{q\; h} \;=\; \frac{\eta_i(q)\; (m_S - \psi\; \tilde{\omega})\; \tilde{\omega}}{\left(\frac{q}{y}\right)^2 \;+\; \sigma\; (\tilde{\omega}^2 - 1)} \tag{7.10}$$

Where $\eta_i(q)$ can be any suitable function. According to [7], inserting it into the torque equation is mathematically equivalent to inserting it into the denominator of the head equation. Expression 7.10 will be the same.

Demonstration of the the η_i implementation will consist in presenting characteristic curves in this section, but also in presenting full Hill charts in the following section 7.4.

η_i will alter the modelling of torque, power and efficiency, the flow model remains unchanged. From the 7 different characteristic curves presented in section 6.1, the torque-speed, power-speed, efficiency-speed, torque-opening degree and efficiency-flow diagrams, will be affected. Since η_i depends on q, its impact on plots having $\tilde{\omega}$ on the x-axis is minor compared to those having y or q on the x-axis. When including η_i, the $t - \tilde{\omega}$ and $p - \tilde{\omega}$ characteristics only show minor changes for low or high opening degree curves, and will therefore be omitted in this section. Given the close coupling between efficiency-speed and efficiency-flow, it is highly relevant to include both plots. The hypothesis was that implementing $\eta_i(q)$ will improve $\tilde{\eta} - \omega$ in addition to $\tilde{\eta} - q$. Finally, the $t - y$ plots are also included mainly to look for nonlinearity.

As in section 6.1, experimental and simulated turbine characteristics will be presented in different plots next to each other. Simulations without the improvement already presented in the analysis will be included for comparison. All relevant MATLAB scripts are described in Appendix A3 and the executable m-files given as attachments to the thesis.

7.3.1 High head Francis turbine

The $\eta_i(q)$ curve implemented in the characteristics of plots (c) below, is the 9th degree polynomial presented already in section 7.2.1. The graphics demonstrate the model improvement, similar results can be achieved with other curves as well.

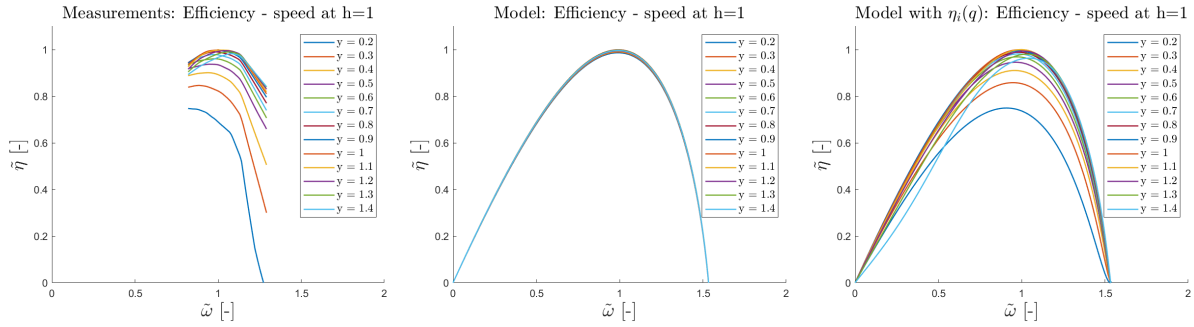

Figure 7.14: Efficiency - speed characteristics for different guide vane openings under constant rated head for high head turbine. From the left: (a) measurements, (b) simulations without η_i and (c) simulations with η_i (both for model inputs by method 2).

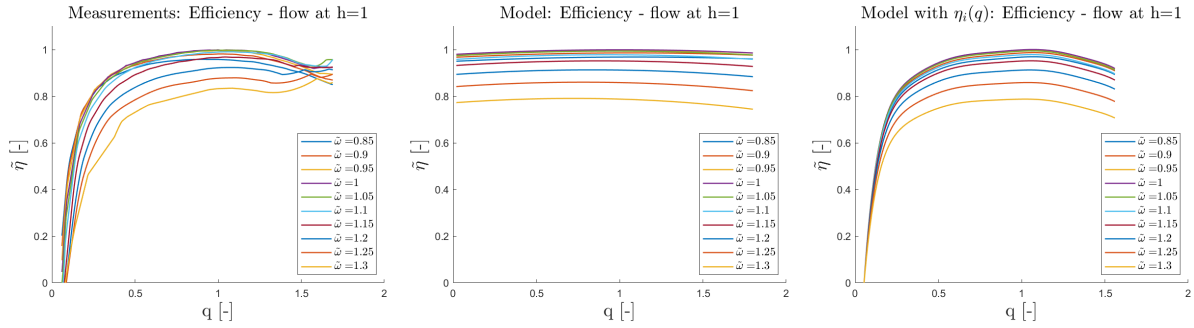

Figure 7.15: Efficiency - flow characteristics for different rotational speeds under constant rated head for high head turbine. From the left: (a) measurements, (b) simulations without η_i and (c) simulations with η_i (both for model inputs by method 2).

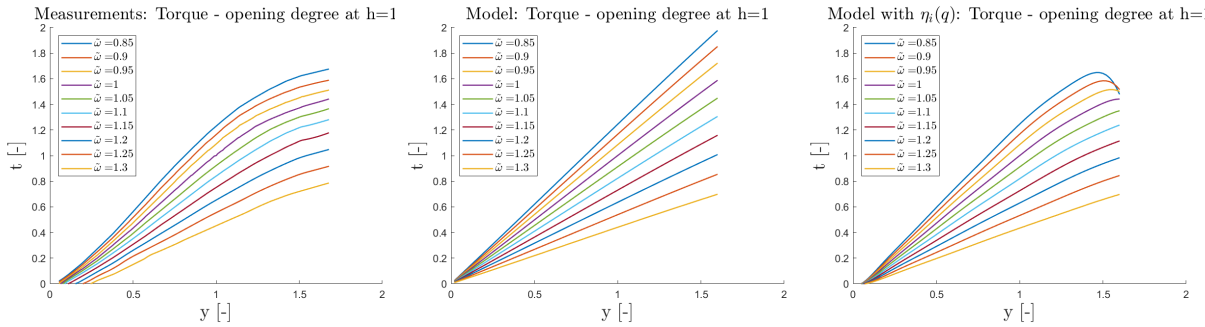

Figure 7.16: Torque - opening degree characteristics for different rotational speeds under constant rated head for high head turbine. From the left: (a) measurements, (b) simulations without η_i and (c) simulations with η_i (both for model inputs by method 2).

Discussion

The improvement in curve shape for the efficiency-flow characteristics in figure 7.15 is expected, and the $\tilde{\omega} = 1$ curve should correspond almost completely. The analysis in section 6.1 showed that correct model inputs can improve accuracy of efficiency *levels* for different speeds, but that that efficiency *shape* for varying flow was predicted poorly by the equations. It becomes the opposite issue in the efficiency-speed diagram. Correct model inputs can improve accuracy of efficiency *shape* for varying speed, but the efficiency *levels* for different guide vane openings was predicted poorly by the equations. This is *the same* problem: hill shape for vertical movement in the diagram is predicted poorly by the equations. Studying plots (c) of figures 7.14 and 7.15 above, $\eta_i(q)$ improves this issue.

The high head runner data were given for a narrow speed range not including runaway operation. Even so, we do expect $q - \tilde{\omega}$ characteristics to intersect the runaway curve in different points. Since only one η_i curve was constructed and implemented for all speeds, the zero efficiency intersection point remains the same. This can be seen in figure 7.14, where both plots (b) and (c) intersect the $\tilde{\omega}$-axis for only one value irrespective of guide vane opening. In a Hill diagram, this implies that the runaway curve is vertical in the region where these $q - \tilde{\omega}$ curves intersect it. It can also be seen in figure 7.15, where plot (c) intersect the q-axis for only one value irrespective of runner speed. In a Hill diagram, this implies that the runaway curve is horizontal in the region containing these speeds.

For the torque-opening characteristics in figure 7.16, deviation from linearity is better captured by the model with η_i than without. There is less impact of η_i on the high speed curves, these did already agree quite well with the measurements, but the curves for low to slightly past rated speed are affected more. Since $\eta_i(q)$ was multiplied into the torque equation, obviously its curve shape will impact curve shape of torque characteristics.

7.3.2 Medium head Francis turbine

The $\eta_i(q)$ curve implemented in the characteristics of plots (c) below, is the 5 harmonics Fourier series presented already in section 7.2.2. The graphics demonstrate the model improvement, similar results can be achieved with other curves as well.

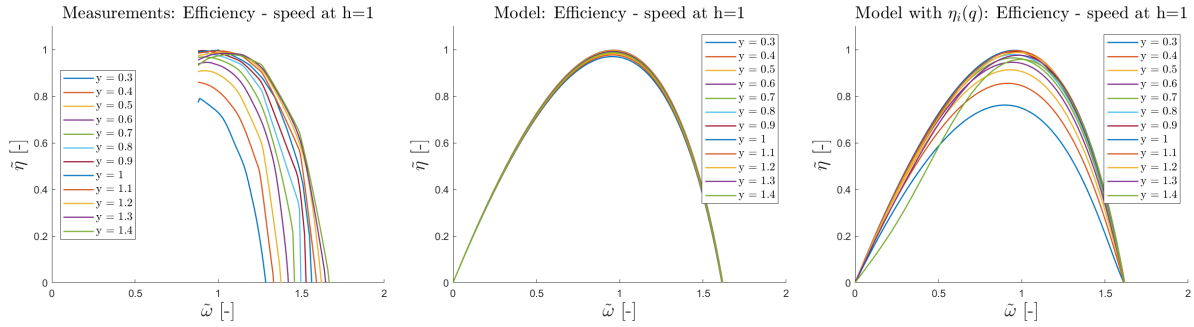

Figure 7.17: Efficiency - speed characteristics for different guide vane openings under constant rated head for medium head turbine. From the left: (a) measurements, (b) simulations without η_i and (c) simulations with η_i (both for model inputs by method 2).

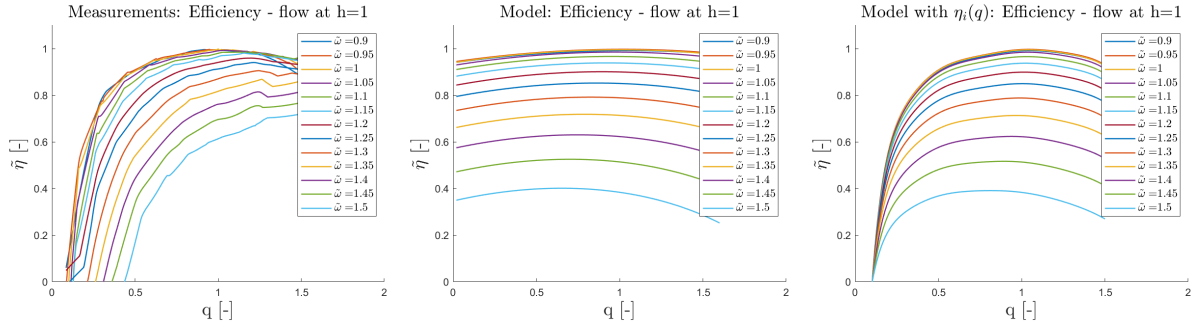

Figure 7.18: Efficiency - flow characteristics for different rotational speeds under constant rated head for medium head turbine. From the left: (a) measurements, (b) simulations without η_i and (c) simulations with η_i (both for model inputs by method 2).

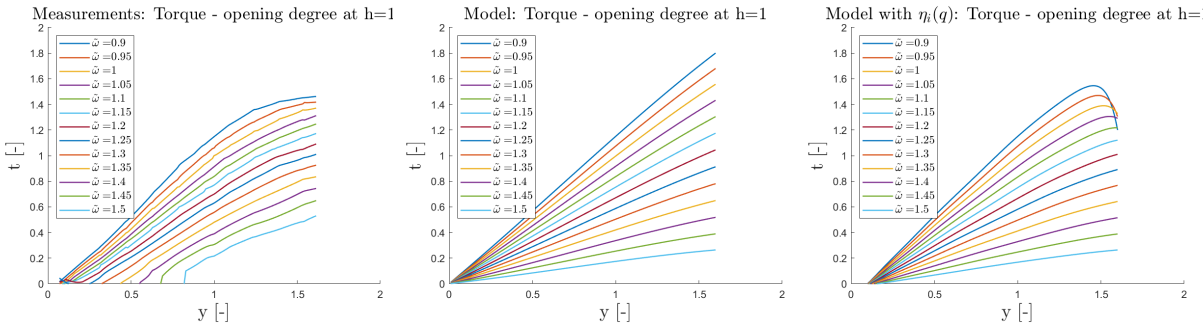

Figure 7.19: Torque - opening degree characteristics for different rotational speeds under constant rated head for medium head turbine. From the left: (a) measurements, (b) simulations without η_i and (c) simulations with η_i (both for model inputs by method 2).

Discussion

Some of the same observations as for the high head turbine can be made also for the medium head. Its data were given in a much larger operational area, including runaway operation. Since its speed range is so wide, the error of implementing only one η_i curve for all speeds becomes more evident. The efficiency-flow curve *shape* in plot (a) of figure 7.18 is similar for different speeds, but higher speeds includes higher flow value for which efficiency is zero (horizontal shift). This implies that the runaway curve has a finite positive slope in the Hill chart, as expected. By implementing only one $\eta_i(q)$ curve going through zero for only one low flow value, this is not modelled.

It was observed in the analysis how the model fails to predict the different guide vane openings for different speeds at which torque is zero, and how this is also related to the shape of the runaway curve. From figure 7.19, including η_i in the torque equation shifts the intersection slightly, but variations are not captured.

As for the high head turbine, the $t - y$ curves for low to slightly past rated speed are more affected by η_i than the high speed curves. Tendencies of overprediction of curvature is observed. Considering that η_i is fitted to $\tilde{\eta} - q$ data, multiplying it into the torque equation can generate some errors. For improvement of torque prediction solely, a curve should be fitted to $t - q$ or $t - y$ data in stead. This is basically direct tuning of model to measurements, and not really the objective here. The intention is to investigate what only *one* curve can do to overall model performance for varying operating conditions.

7.3.3 Low head Francis turbine

The $\eta_i(q)$ curve implemented in the characteristics of plots (c) below, is the 4th degree polynomial presented already in section 7.2.3. The graphics demonstrate the model improvement, similar results can be achieved with other curves as well.

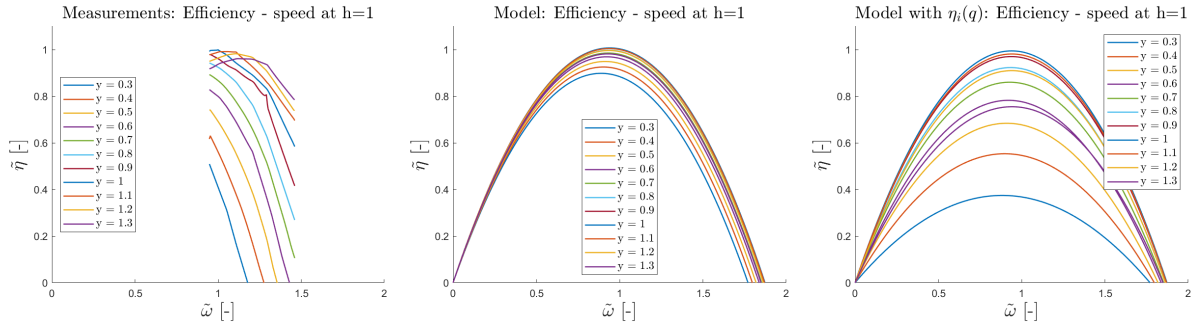

Figure 7.20: Efficiency - speed characteristics for different guide vane openings under constant rated head for low head turbine. From the left: (a) measurements, (b) simulations without η_i and (c) simulations with η_i (both for model inputs by method 2).

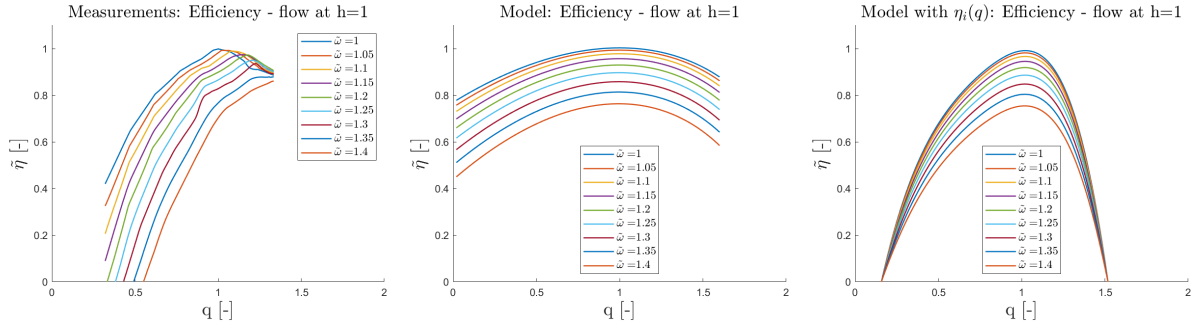

Figure 7.21: Efficiency - flow characteristics for different rotational speeds under constant rated head for low head turbine. From the left: (a) measurements, (b) simulations without η_i and (c) simulations with η_i (both for model inputs by method 2).

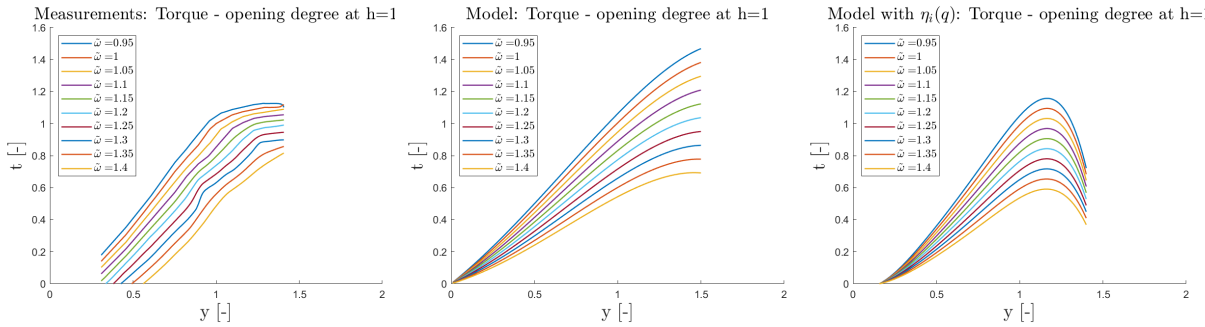

Figure 7.22: Torque - opening degree characteristics for different rotational speeds under constant rated head for low head turbine. From the left: (a) measurements, (b) simulations without η_i and (c) simulations with η_i (both for model inputs by method 2).

Discussion

In the analysis, model behaviour for the low head turbine differed from the higher head ones. The biggest difference was the efficiency-flow curvature and the small level differences for efficiency-speed curves. The model equations capture *some* losses along the q-axis, yet not enough. Studying plots (c) of figures 7.20 and 7.21, including a suitable η_i can improve this. Similar to the preceding turbines, different runaway speeds are not captured.

The proposed 4th degree polynomial for $\eta_i(q)$, is sharp and decreases monotonically and rapidly for flow rates past the upper limit of the data. Studying the $t-y$ characteristics of figure 7.22, the aggressive η_i shape results in overprediction of curvature for increasing guide vane openings. The model characteristics with η_i are improved up to about $y = 1.2$, but become more inaccurate for higher openings. Choosing a less steep η_i curve, for example the parabola by Nielsen [5], may improve this. It is a matter of curve choice and area of interest i.e. how high flow rate intended to model.

7.4 Hill diagrams

Complete Hill diagrams from the measurements, the turbine model without any efficiency improvements, as well as with the η_i curve implementations, will finally be presented in this section. They do not contribute any new information, model performance compared to the experimental data have already been discussed based on the 2D plots. The diagrams are a compact and convenient presentation format, and are mainly included to confirm and summarize previous findings.

As discussed in section 7.1.3, the η_i curve was required to decrease monotonically slightly past the upper flow range limit, but we cannot know for sure what behaviour to expect in this area. Depending on the turbine there is an upper limit to the guide vane setting, such that flow rates corresponding to openings past this limit are nonphysical and unnecessary to include in a Hill chart. The dashed rectangle defines the operating area where experimental data were given. Evaluating the model's ability to predict hill shape can be done in the entire diagram, but validation against the measurements can only be done inside the dashed rectangle (the "validation area").

Separate MATLAB scripts were written to generate Hill charts. These are described in Appendix A3 and the executable m-files given as attachments to the thesis.

7.4.1 High head Francis turbine

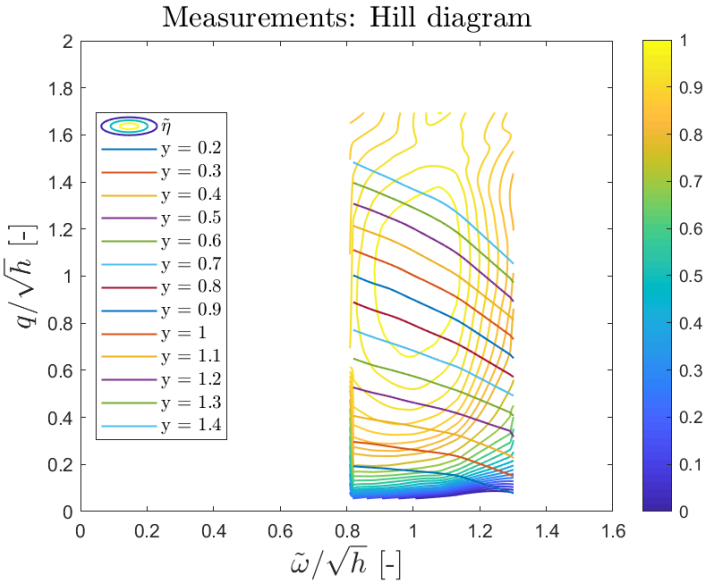

Figure 7.23: Hill diagram from the measurements for high head turbine.

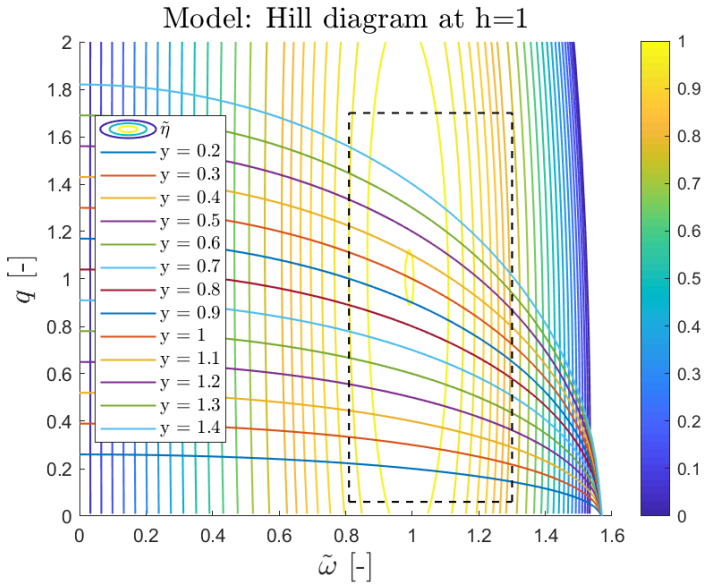

Figure 7.24: Hill diagram from the model without η_i for high head turbine. Dashed rectangle defines the area where measurements were given (validation area).

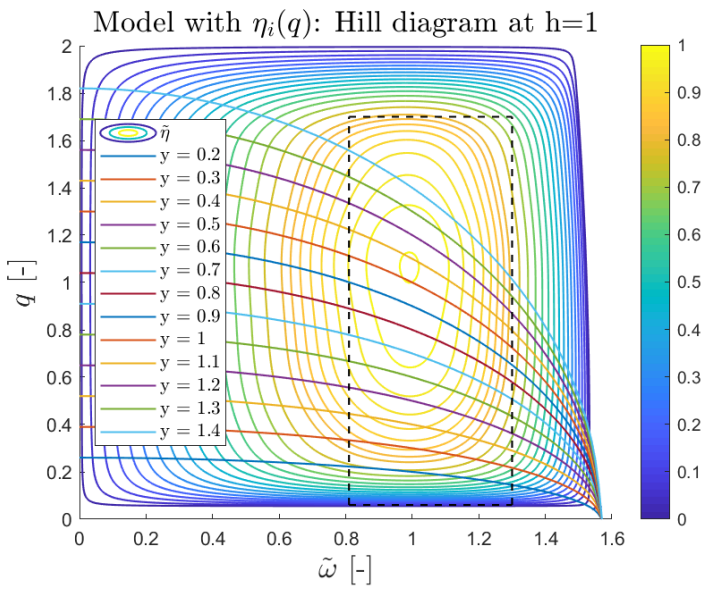

Figure 7.25: Hill diagram from the model with η_i for high head turbine. Dashed rectangle defines the area where measurements were given (validation area).

Discussion

Comparing figure 7.23 to 7.25, inside the validation area, the Hill diagram is predicted very well by the model including η_i. The model without η_i in figure 7.24, does not capture much slope for vertical movement in the diagram. These results are quite as expected.

Clearly, the issues of modelling the runaway curve remain also with the improvement. The speeds where the $q - \tilde{\omega}$ characteristics intersect the zero efficiency line are the same, since the runaway curve is vertical in this region. Its horizontal region results from implementing only one η_i curve going through zero in one low flow value, for all speeds. These vertical and horizontal regions were anticipated already in section 7.3 (the vertical region already in section 6.1) based on the 2D plots. The rectangular curve shape does not resemble runaway curves of real Francis turbines.

7.4.2 Medium head Francis turbine

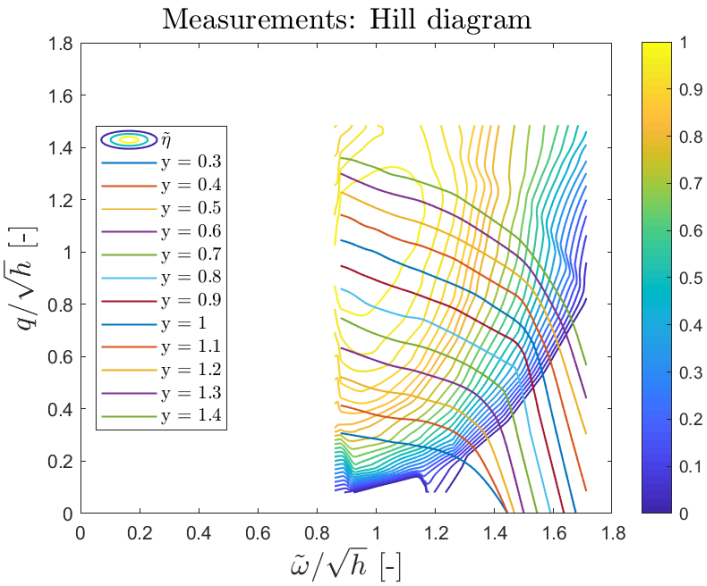

Figure 7.26: Hill diagram from the measurements for medium head turbine.

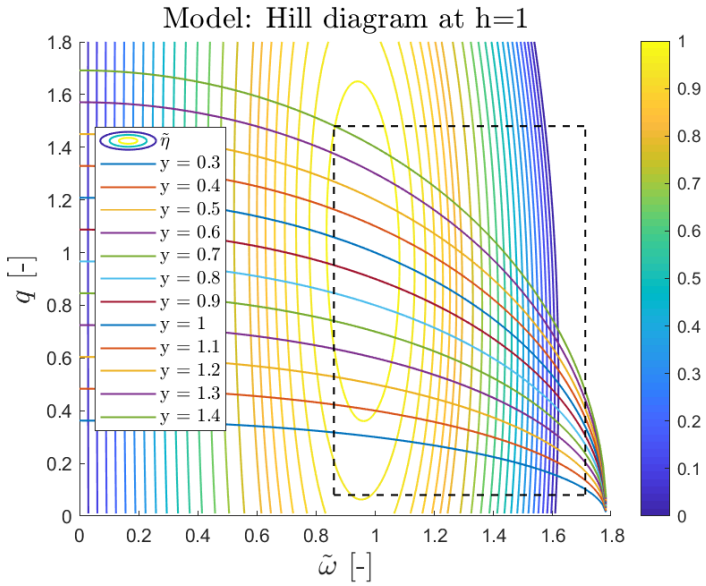

Figure 7.27: Hill diagram from the model without η_i for medium head turbine. Dashed rectangle defines the area where measurements were given (validation area).

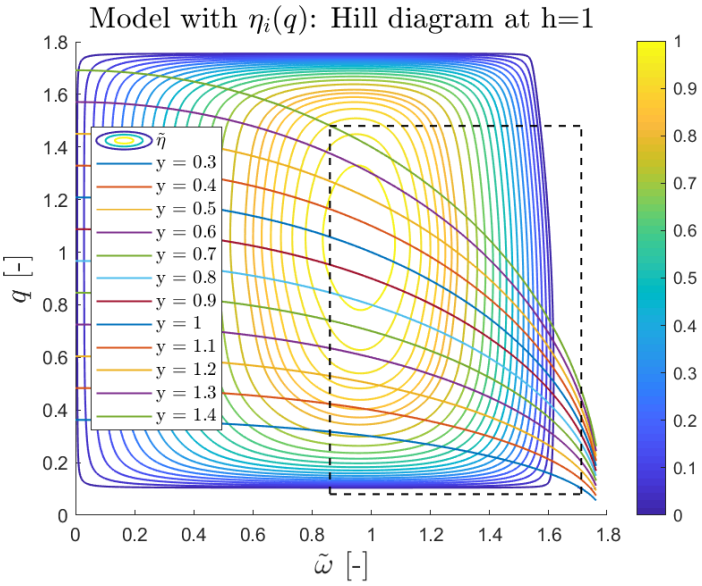

Figure 7.28: Hill diagram from the model with η_i for medium head turbine. Dashed rectangle defines the area where measurements were given (validation area).

Discussion

Comparing figure 7.26 to 7.28, inside the validation area and in proximity to the efficiency peak, the Hill diagram is predicted quite well by the model including η_i. The farther away from the peak, the more imprecise is the model with respect to these experimental data. Nevertheless, figure 7.28 is clearly more similar to figure 7.26 than what 7.27 is.

Runaway operation was well included in the experimental data. From rated guide vane opening characteristic, $y = 1.0$ (dark blue), the measured runaway operation assuming constant rated head $h = 1$, is at $\tilde{\omega} \simeq 1.56$, $q \simeq 0.53$. The same runaway operation is predicted by the model (with or without η_i, it does not affect) to be at $\tilde{\omega} \simeq 1.62$, $q \simeq 0.51$, for rated head $h = 1$. The model predicts reasonable runaway operation at $y = 1$, yet not for other openings. Also in the area around $\tilde{\omega} = 1$ for very low q or y, the runaway curve agrees quite well. It is mainly the shape in the region between these two areas which does not resemble the real runaway curve.

7.4.3 Low head Francis turbine

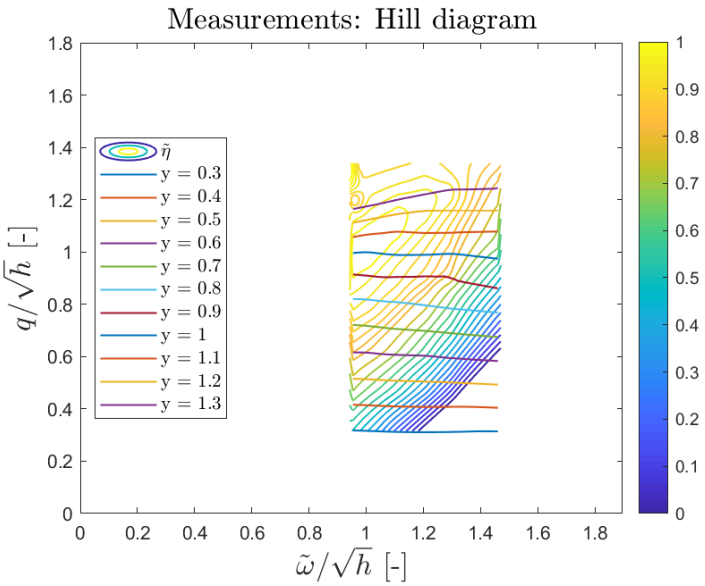

Figure 7.29: Hill diagram from the measurements for low head turbine.

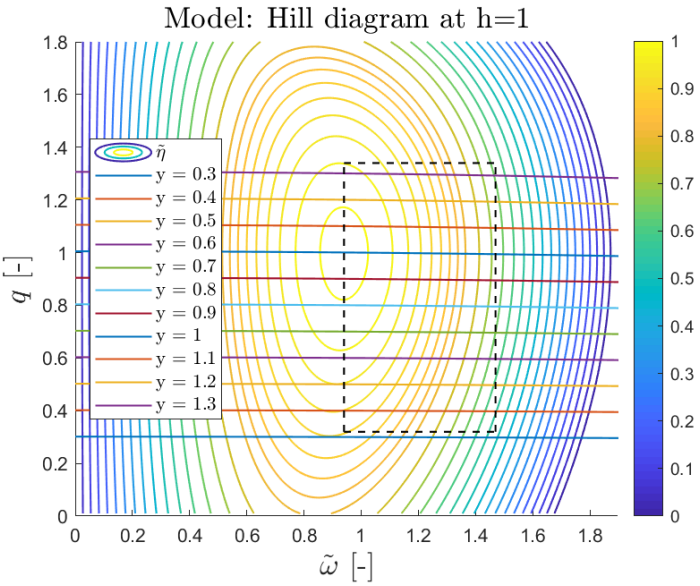

Figure 7.30: Hill diagram from the model without η_i for low head turbine. Dashed rectangle defines the area where measurements were given (validation area).

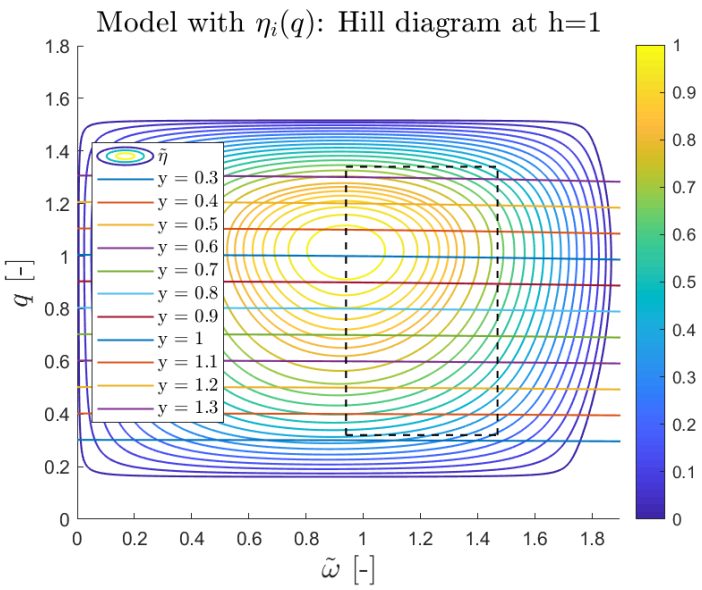

Figure 7.31: Hill diagram from the model with η_i for low head turbine. Dashed rectangle defines the area where measurements were given (validation area).

Discussion

From figure 7.29, the area of experimental data is rather limited. Runaway operation is only included for a few low opening values, but in the measured speed-flow range, the runaway curve increases with a slope of about 1. Since only one η_i curve was implemented for all speeds in figure 7.31, the runaway curve is modelled as horizontal in the same speed range. From the model without η_i in figure 7.30, there were already tendencies of curvature along the q-axis, just not enough. Figure 7.31 shows how including η_i increases the slope drastically.

Investigating the measured Hill chart closely, in the upper left corner there are tendencies of efficiency increase when moving away from BEP. Since this happens around $\tilde{\omega}/\sqrt{h} = 1.0$, a section of data for $q/\sqrt{h} > 1.20$ were filtered out before curve fitting. Looking at the hill slope slightly to the right indicates that in reality, it is less steep than what we anticipated based solely on the q range $1.00 - 1.20$. As a result, the chosen η_i curve is too steep for higher flow values. Comparing figure 7.29 to 7.31 and disregarding for a while that the runaway curve is not reached. Inside the validation area, the Hill diagram is predicted relatively good for lower flow rates by the model with η_i, but less good for higher flow rates. This results from choosing a too steep incipient efficiency curve.

If we were to perform the curve fitting for this low head runner again, a few extra data points to ensure its efficiency slope for $q > 1.0$ is less steep, would have been performed.

7.5 Generalization of the model improvement

The improved agreements in characteristic curves in section 7.3, or the changes in hill shapes in section 7.4, are expected. It is interesting to investigate whether the model can be improved also for *other* Francis turbines, by producing general incipient efficiency $\eta_i(q)$ curves. That is, propose the improvement without loss of generality for the model.

In this section, the concept of generalizing the curve, as well as a specific approach to do so, is described.

7.5.1 Dependency on speed number

Presented below are the p.u. measured efficiency divided by model efficiency versus p.u. flow rate, which was the basis for the η_i curve fittings. The figure is included to emphasize expected η_i curve shapes for different Francis turbines; the low head should have a much sharper curve than the high head, which should have a wider shape with a larger flat region. This generalization is as also consistent with references like [25] [32]. In [32], normalized efficiency-flow curves at constant rotational speed and head (as was our case), show sharper curves for higher specific speeds.

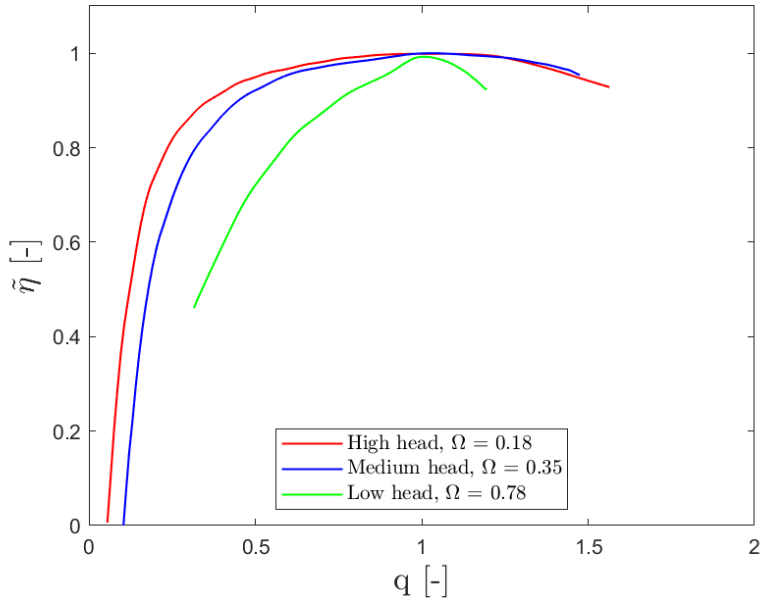

Figure 7.32: Measured efficiency data divided by model efficiency at nominal speed and head, $\tilde{\omega} = 1$ and $h = 1$, the basis for fitting $\eta_i(q)$ curves. The speed numbers, Ω, are estimates from the design by tuning method 2.

The speed numbers from the design by method 2 are included in figure 7.32 to indicate Francis turbine type. The absolute values are less relevant than the values relative to each other. For generalization of $\eta_i(q)$ curves based on turbine speed number Ω, increasing Ω should "sharpen" the curve (green), while decreasing Ω should "widen" it (red). Consequently, increasing Ω should slightly increase the zero efficiency flow rate, while decreasing Ω should slightly decrease it. This is the basic idea we want a general $\eta_i(q)$ curve dependent on speed number, to fulfill.

Coefficients of the different curve fitting models were plotted against Ω for the turbines to investigate any correlations. With only three turbines, a correlation can at best be found as linear. However, the correlations turned out rather weak. Only the polynomials showed some tendencies of linearity between coefficients and speed number when fitting by the same degree polynomial. To obtain curve coefficients by linear interpolation for a certain Ω did not generate any nice curves, and after several attempts, the idea was rejected. If more experimental data had been available, perhaps correlations (not necessarily linear ones) could have been found.

7.5.2 Linear interpolation between functions

A first order approximation to obtain the η_i curve for a given speed number, is to linearly interpolate between two curves corresponding to a lower and a higher speed number. That is, the resulting curve function is a combination of two curve functions, where the weighting is based on Ω:

$$\eta_i(q) \;=\; (1-x)\; f_1(q) \;+\; x\; f_2(q) \;=\; f_1(q) + x\;(f_2(q) - f_1(q)) \tag{7.11}$$

Where

$$x(\Omega) \;=\; \frac{\Omega - \Omega_1}{\Omega_2 - \Omega_1} \tag{7.12}$$

Ω_1 and $f_1(q)$ correspond to the lower speed number, i.e. the high head turbine, and Ω_2 and $f_2(q)$ correspond to the higher speed number, i.e. the low head turbine. For any speed number in this range, $\Omega_1 \leq \Omega \leq \Omega_2$, the resulting $\eta_i(q)$ curve will be a linear combination of the two curves. The $f_1(q)$ and $f_2(q)$ functions do not have to be of similar type. Any suitable mathematical curve will do, but the resulting function becomes a simpler expression if they are.

Taking the high head runner's 9th degree polynomial and the low head runner's 4th degree polynomial presented in equations 7.6 and 7.8, together with their estimated speed numbers of 0.18 and 0.78, respectively, as an example. For any speed number in this range, a curve can be obtained as:

$$\begin{aligned}\eta_i(q) \;=\; (1-x)\,(p_1q^9 + p_2q^8 + p_3q^7 + p_4q^6 + p_5q^5 + p_6q^4 + p_7q^3 \\ +p_8q^2 + p_9q + p_{10}) \;+\; x\,(p_1q^4 + p_2q^3 + p_3q^2 + p_4q + p_5)\end{aligned} \tag{7.13}$$

Where the coefficients p_j for the 9th and the 4th degree polynomials were given in tables 7.1 and 7.3, respectively. x is determined by the speed number:

$$x(\Omega) \;=\; \frac{\Omega - 0.18}{0.78 - 0.18}, \qquad 0.18 \leq \Omega \leq 0.78 \tag{7.14}$$

The resulting incipient efficiency curve from relation 7.13 will be a 9th degree polynomial (unless $\Omega = 0.78$ which gives the 4th degree polynomial). Figure 7.33 below is included to illustrate the concept for different speed numbers in this range.

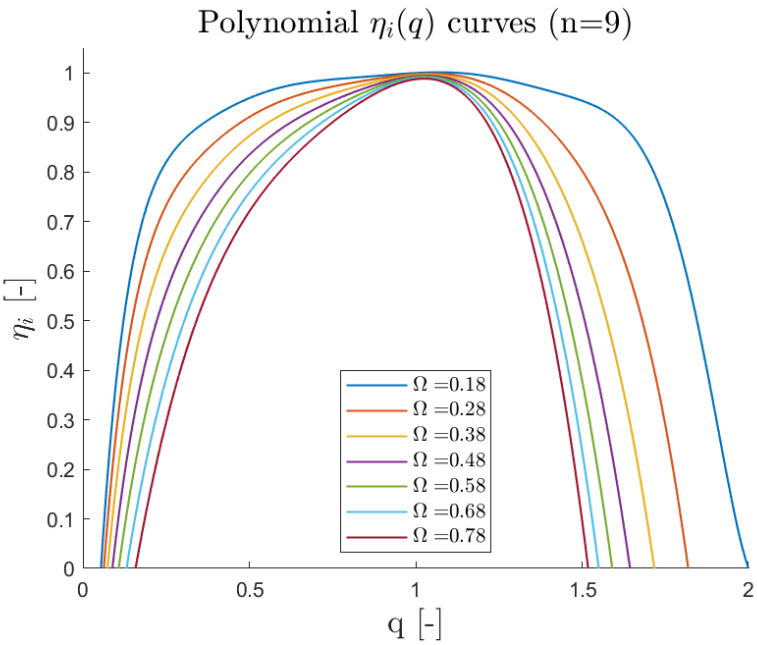

Figure 7.33: Proposed polynomial $\eta_i(q)$ curves for speed numbers in the range $0.18 \leq \Omega \leq 0.78$. The curves are a linear combination of the innermost curve in red (4th degree polynomial) and the outermost curve in blue (9th degree polynomial).

The curves of figure 7.33 can be implemented in the model by multiplication into the torque equation, and will improve performance for the drawbacks addressed.

7.6 General discussion of the incipient efficiency

The model including an $\eta_i(q)$ curve going from 0 through 1 and back to 0, will produce a Hill diagram shaped similar to a cone with a rectangular base. Towards zero efficiency, hill shape is more rectangular, while towards BEP, more oval. In a certain area around the peak, the model appears to predict general shape well, but towards zero efficiency, it is more off target. Such hill shape, especially the charts modelling the medium and high head turbines, resembles the Pelton Hill chart in figure 3.2. This is quite interesting, considering that η_i was fitted with the intention to correct for losses related to varying flow, constant speed operation. A Pelton does not have losses associated with sub-optimal inflow angles when $y \neq 1$ (incidence and draft tube swirl losses, see section 3.5). A Pelton's efficiency-flow curve for constant speed is generally wide and flat, especially for several jets [26].

In the doctoral thesis of professor Torbjørn Nielsen [6], the origin of these model equations, the same hydraulic losses which η_i intend to capture, were modelled as head losses with empirical coefficients R_f, R_c and R_d. These losses are *subtracted from* the torque equation, unlike η_i which is *multiplied into* the torque equation. The purpose of η_i was to avoid empirically based coefficients like the R-terms, and to collectively model irreversible hydraulic losses by only one function.

R_f, R_c and R_d represents viscosity (friction), incidence loss and draft tube swirl loss, respectively. The last two are due to wrong (sub-optimal) inflow angles. His loss models for these two contain $(Q - Q_c)^2$, where Q_c is the discharge when incidence angle is zero. He derives an expression for Q_c depending on guide vane angle α_1 and rotational speed ω. Therefore, Nielsen's loss models related to wrong inflow angles depend on flow, guide vane angle and rotational speed. In comparison, $\eta_i(q)$ as it was produced in this work or in [5], is simplified to completely disregard the effect of varying speed, and was fitted to data for $\tilde{\omega} = 1$ as a function of flow q solely. Based on the shape of the Hill charts, this simplification seems more appropriate for Pelton than for Francis turbines.

When implementing the same $\eta_i(q)$ curve for all speeds, it will not capture any additional losses along the speed axis. The analysis in Chapter 6 showed that the equations manage this quite well on their own. Prediction along the flow axis was the main weakness, which is why η_i is produced to depend solely on flow. The analysis also showed that runaway conditions are not captured too well by the model. In [6], a term representing disk- and mechanical friction losses is also subtracted from the torque equation. Since it depends on runner speed squared, it is constant for $\tilde{\omega} = 1$, but will increase when $\tilde{\omega}$ increase. The loss coefficients included in the torque equation (the R's) are tuned to measurements, more precisely by peak efficiency location, starting torque and runaway speed.

Based on the above discussed factors, neglecting the impact of varying speed on losses due to wrong inflow angles, can possibly be the reason why the model struggles to capture runaway conditions for a Francis turbine accurately. Not including any additional models for disk- and mechanical friction as [6] did, might also have a certain impact. Given the physics of energy losses in a Francis described in section 3.5, only one $\eta_i(q)$ curve for all speeds appears to be too simple to completely correct also the runaway curve.

Chapter 8

Conclusion

The mathematical turbine model studied in this master thesis, was developed from Euler's turbine equation and the opening degree definition. It is based on a first principles approach, and requires very few inputs; σ, ψ and α_{1R}. The input parameters can be found from the main dimensions and nominal operating point of a turbine. This can be obtained by measuring the geometry, or by performing a simplified design "from scratch", as conducted in section 5.1 by Brekke's method. An alternative approach to obtain model inputs, given that the Hill chart is known around the point of best efficiency, was developed in section 5.2. Based solely on measured gradients of flow and torque with respect to speed and head at BEP, σ and ψ can be obtained. Knowing also α_{1R} from measurements, the model is complete. This demonstrated a quick procedure to set up a physical correct mathematical turbine model from only one measured operating point.

Analysis of model performance compared to real Francis behaviour, was conducted *before* and *after* incipient efficiency, η_i, curves were proposed. General hill shape along constant y curves were already quite good, but irreversible hydraulic losses along the q-axis were not captured to the same extent. Even though the basic physics of the governing equations are correct, these losses are not intrinsically a part of the 1D model, and should be modelled separately. For this purpose, an η_i curve depending on q and fitted to data at $\tilde{\omega} = 1$, $h = 1$, was developed and implemented.

The flow equation is unaffected by η_i as it is inserted into the torque equation. Geometrical constant σ proved to be very decisive for predicting flow-speed characteristics. To a certain extent, the model can be tuned by σ to yield quite accurate characteristics. At $\tilde{\omega} = 1$, the effect of σ and thus of runner main dimensions, is lost. The relation between head, flow and guide vane opening is reduced to the valve equation.

This work demonstrates that the general hill shape is highly affected by the choice of $\eta_i(q)$ curve, and that it can, when chosen wisely, be greatly improved. The performance diagram is well predicted around the efficiency peak, but the curves towards zero effi-

ciency tend to be almost rectangular. Prediction of runaway conditions can be relatively good around $\tilde{\omega} = 1$ and y low, and around $y = 1$ and $\tilde{\omega}$ high, but for the region in between, modelled runaway curves deviate from measured ones.

Despite the inaccuracies observed when both $\tilde{\omega}$ and y differ largely from 1, the $\eta_i(q)$ improvement appears to be very promising. For stability- or grid analysis around $\tilde{\omega} = 1$ for any q and y, the model is considerably improved. As shown, it already performs well around $y = 1$ for any $\tilde{\omega}$, as long as inputs are precise. That said, for the purpose of predicting *all parts* of the Hill diagram accurately, for example for general transient analysis including load rejection, start-up or shut-down, further model improvement is necessary.

Chapter 9

Further Work

Since Rainpower did not provide runner geometry with the data, two approaches to obtain reasonable designs based on the data (reversed engineering), were attempted. The first approach caused challenges in obtaining a reasonable low head Francis design. Further work can involve a closer look at the issues discussed in section 5.1.2, and to improve the low head runner design based on measured BEP.

The analysis in Chapter 6 attempted to distinguish between general model behaviour and input behaviour. Future work can be to design more turbines for model application in order to back up the analysis. In regards to distinguish these two types of errors, more simulations could improve the quality of the analysis.

The proposed improvement was based on *model* turbine data, but is intended to be applicable to *prototypes*. Scale effects, especially for frictional losses, may shift efficiency curve and BEP from model to prototype [22]. Future work can include investigating these effects and possibly incorporate the appropriate scaling formulas.

In order to achieve a runaway curve with finite positive slopes, several η_i functions could have been implemented. Developing a few curves for certain values of $\tilde{\omega}$, interpolation can be used in between. This implies that the process of deciding η_i based on Ω should generate preferably three (or at least two) curves, corresponding to lower and higher speed values of interest as well as rated. This can increase accuracy of the runaway curve by avoiding non-physical horizontal and vertical sections. Alternatively, one can develop multivariate incipient efficiency functions; $\eta_i(q, y, \tilde{\omega})$ or $\eta_i(q, \tilde{\omega})$. These should be valid and smooth in the entire operating area. Such functions can be intricate to develop based on experimental data, not to mention difficult to generalize based on Ω. To keep it one-dimensional, $\eta_i(q)$, and develop a few different ones for different $\tilde{\omega}$ and interpolate in between, is perhaps more realistic. Any approach to obtain variation in η_i along also the $\tilde{\omega}$-axis, can be included in further work.

For the purpose of stability analysis along $\tilde{\omega} = 1$ and $y = 1$, the linearized model is convenient. Future work can involve more investigation into this version including η_i. Only characteristic values at BEP were calculated in this work, but these are not affected by η_i. Linear model behaviour at other operating points can also be investigated.

To generalize the improvement with respect to Ω as described in section 7.5, or by any other appropriate method, can also be performed. Following this, software implementation for application to real plants can be the next step. LVTrans already uses the turbine model studied in this work, but including the empirical R-terms which must be tuned. The η_i, as an alternative way of modelling losses, can be implemented into LVTrans.

Last but not least, to develop η_i curves appropriate for other turbine types like Pelton or Kaplan, can also be part of the scope for future work.

Bibliography

[1] Bourmaud J. Girard R. Kariniotakis G. Heggarty, T. Multi-temporal assessment of power system flexibility requirement. *Elsevier: Applied Energy*, 238:1327–1336.

[2] Jaehnert S. Tomasgard A. Midthun K. Skar, C. and M. Fodstad. Norway's role as a flexibility provider in a renewable europe - a position paper prepared by fme censes. 2018.

[3] W. Henley. 2019 hydropower status report. 2019.

[4] H. Brekke. Kompendium: Grunnkurs i hydrauliske strømningsmaskiner, 2002.

[5] T. K. Nielsen. Simulation model for francis and reversible pump turbines. *International Journal of Fluid Machinery and Systems*, 8(3), July-September 2015.

[6] T. K. Nielsen. *Transient Characteristics of High Head Francis Turbines*. PhD thesis, Norges Tekniske Høgskole (NTH), Institutt for Hydro- og Gassdynamikk, Waterpower Laboratory, Trondheim, December 1990.

[7] B. Svingen and T. K. Nielsen. Transient hydraulic calculations with a linear turbine model derived from a nonlinear synthetic model. *IOP Conference Series: Earth and Environmental Science*, 240, March 2019.

[8] B. Svingen and T. K. Nielsen. First principles approach linear model for hydraulic turbines suitable for use in available simulation platforms. November 2018.

[9] E. B. Wylie and V. L. Streeter. *Fluid Transients in Systems*. Prentice Hall, Englewood Cliffs, NJ, 1993.

[10] Henderson A. D. Walker J. M. Giosio, D. R. and P. A. Brandner. Physics-based hydraulic turbine model for system dynamic studies. *IEEE Transactions on Power Systems*, 32(2):1161–1168, March 2017.

[11] Zhang L. Xu T. Zeng, Y. and H. Dong. Building and analysis of hydro turbine dynamic model with elastic water column. *2010 Asia-Pacific Power and Energy Engineering Conference*, pages 1–5, March 2010.

[12] H. Brekke. *The structure matrix method for stability analysis of hydroelectric power plants*. BHRA (Information Services), Cranfield, 1990.

[13] H. Brekke and X. Li. A new approach to the mathematic modelling of hydropower governing systems. In *1988 International Conference on Control - CONTROL 88.*, pages 371–376, April 1988.

[14] B. Svingen. *Fluid Structure Interaction in Piping Systems*. PhD thesis, The Norwegian University of Science and Technology (NTNU), Faculty of Mechanical Engineering, Trondheim, September 1996.

[15] B. Svingen. Lvtrans manual, June 2018.

[16] Guo Y. Zhang L. Xu T. Zeng, Y. and H. Dong. Torque model of hydro turbine with inner energy loss characteristics. *Science China Technological Sciences*, 53(10):2826–2832, October 2010.

[17] Huimin G. and Chao W. Effect of detailed hydro turbine models on power system analysis. pages 1577–1581, 2006.

[18] IEEE Working Group on Prime Mover and Energy Supply Models for System Dynamics Studies. Hydraulic turbine and turbine control models for system dynamic studies. *IEEE Transaction on Power Systems*, 7(1):167–179, February 1992.

[19] Balu N. J. Kundur, P. and M. G. Lauby. *Power System Stability and Control*, volume 7. McGraw-Hill, New York, 1994.

[20] P. Storli and T. K. Nielsen. Simulation and discussion of models for hydraulic francis turbine simulations. *IFAC PapersOnLine*, 51(2):109–114, 2018.

[21] T. Ida. Analysis of scale effects on performance characteristics of hydraulic turbines. *Journal of Hydraulic Research*, 27:809–831, 1989.

[22] H. Brekke. Analysis of losses in hydraulic turbines. In Espert V. Cabrera, E. and F. Martinez, editors, *Hydraulic Machinery and Cavitation*, pages 294–303, Dordrecht, 1996. Springer Netherlands.

[23] Kurokawa J. Matumoto M. Kitahora, T. and R. Suzuki. Prediction of scalable loss in francis runners of different specific speed. In Espert V. Cabrera, E. and F. Martinez, editors, *Hydraulic Machinery and Cavitation*, pages 323–332, Dordrecht, 1996. Springer Netherlands.

[24] Trivedi C. Iliev, I. and O. G. Dahlhaug. Simplified hydrodynamic analysis on the general shape of the hill charts of francis turbines using shroud-streamline modeling. *Journal of Physics: Conference Series*, 1042:012003, June 2018.

[25] H. Brekke. *Hydraulic turbines - Design, Erection and Operation*. 2014.

[26] H. Brekke. Kompendium: Pumper og turbiner, 2003.

[27] J. Raabe. *Hydraulische Maschinen und Anlagen*. VDI-Verlag, Düsseldorf, 2. aufl. der teile 1 bis 4 in einem band. edition, 1989.

[28] Y. A. Çengel and J. M. Cimbala. *Fluid Mechanics: Fundamentals and Applications*. McGraw-Hill, Boston, 2nd, in si units. edition, 2010.

[29] Luo X. Zhu G. Feng, J. and G. Wu. Investigation on disk friction loss and leakage effect on performance in a francis model turbine. *Advances in Mechanical Engineering*, 9, August 2017.

[30] Rainpower homepage: `https://www.rainpower.eu`.

[31] M. F. Svarstad. *Fast Transition between Operational Modes of a Reversible Pump-Turbine*. PhD thesis, Norwegian University of Science and Technology (NTNU), Trondheim, 2019.

[32] Souza Z. Viana A. Villa-Nova H. Rezek Â. Pinto L. Siniscalchi R. Bragança R. Bortoni, E. and J. Bernardes. The benefits of variable speed operation in hydropower plants driven by francis turbines. *Energies*, 12, September 2019.

Appendix

A1 Paper for the IAHR Symposium 2020

Attached is the submitted and accepted paper for the IAHR 30th Symposium on Hydraulic Machinery and Systems 2020, in Lausanne, Switzerland. This version was submitted 14.02.2020.

Theoretical Turbine Model with Hydraulic Losses

B Svingen[1,2], A F Reines[1], T K Nielsen[1], P T Storli[1]

[1] Department of Energy and Process Engineering, NTNU, Norway
[2] Hymatek Controls AS, Oslo, Norway

bjoernar.svingen@hymatek.no, anefr@stud.ntnu.no, torbjorn.nielsen@ntnu.no, paltore.storli@ntnu.no

Abstract. When doing transient and dynamic analysis of powerplants, a mathematical representation of the turbine is needed. These models can be fully based on empirical data, or they can be based on a first principles approach. A first principles approach is practical in many circumstances because any known and unknown turbine can be modelled, and the model can easily be simplified and linearized without losing generality or physical correctness. The model can be used for general transient studies, also where the turbines are not yet specified, or when doing control system analysis and analysis of the grid.

Using the Euler turbine equations together with the definition of the turbine opening degree, one obtains a model that is geometrically and physically correct. The development of this model started several years ago, and this paper presents the current development. The model shows good agreement of the efficiency as a function of speed, as part of the Euler turbine equation. However, the efficiency as a function of flow is over predicted because hydraulic loss phenomenon such as turbulence and dissipation are not inherently included. There are several ways to include the hydraulic losses. This can be done in a simplified manner, or more elaborate, which also could involve empirical relations for exact fit with a measured turbine. This paper discusses these possibilities, along with some examples.

1. Introduction

When doing analysis of hydropower plants, a mathematical model of the turbine is needed. The accuracy and detail of the model needed, varies according to the accuracy and detail required for the results. Typical analysis are analysis of the grid (including hydropower plants); stability analysis of the turbine and turbine governor, and transient analysis of the turbine and waterways. The required detail and accuracy of the turbine model will typically increase accordingly. In later years, the addition of so called *digital twins* and similar concepts, with their own sets of requirement regarding accuracy and detail, have emerged[3][4].

In 1990, Nielsen introduced a new model[6]. This model has been studied and developed ever since, for instance, but not limited to [5][8][9][10][11][12]. This is also the main model used in LVTrans[1][2]. This model is based on a first principles approach using the Euler Turbine equation and the equation for the opening degree. Being a first principles approach, it lends itself naturally to studying and modelling of the details of the physical losses in a turbine.

The Nielsen model includes "Euler losses". These losses are caused by the turbine spinning too fast or too slow in relation to nominal speed. The model also includes effects due to geometry. However, in its most basic form, the model does not include losses caused by irreversible phenomenon, such as turbulent losses, losses in draft tube etc. This makes the basic model able to accurately predict turbine

behaviour along constant opening curves and along the "speed axis", but not the irreversible losses that typically occur along the "flow axis"[8][9][11][12]. Whether these losses are of importance depends on the nature of the analysis. The implication of these losses, as well as modelling alternatives are presented and discusses in this paper. The discussion is mainly about Francis turbines, but is also relevant for Pelton turbines. It does not include Kaplan turbines as of the time being.

.

2. Governing equations

The derivation of the model, which is not trivial, can be found in[6][8][9][12], and in particular[11]. The main starting point is the Euler turbine equation and the related inlet/outlet diagrams, Figure 2.1

$$\eta_h = \frac{1}{gH_e}(c_{u1}u_1 - c_{u2}u_2) = \frac{\omega_t}{gH_e}(c_{u1}r_1 - c_{u2}r_2) \tag{1}$$

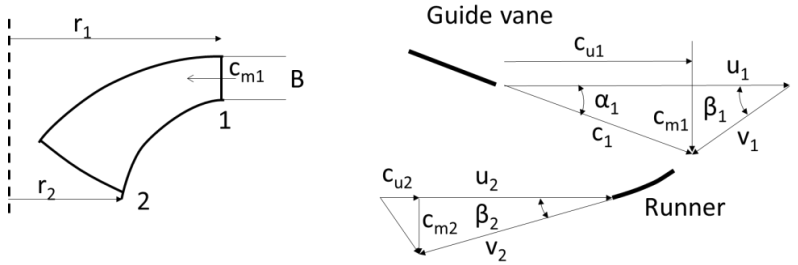

Figure 2.1 Inlet and outlet diagrams for a Francis runner

H_e is the total head between two points where the efficiency, η_h is defined, for instance:

$$H_e = \left(H_0 + \frac{c_0^2}{2g} + Z_0\right) - \left(H_3 + \frac{c_3^2}{2g} + Z_3\right) \tag{2}$$

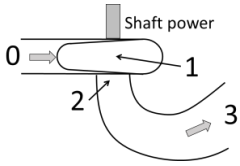

Figure 2.2 Sketch of a Francis turbine roughly showing the numbered positions.

Here η_h is a *general*, but unknown loss function (typically the efficiency). The next starting point is the opening degree, κ, defined as:

$$\kappa = \left(\frac{Q}{\sqrt{2gH_e}}\right) \Big/ \left(\frac{Q_R}{\sqrt{2gH_{eR}}}\right) \quad \xrightarrow{yields} \quad gH_e = gH_{eR}\left(\frac{Q}{\kappa Q_R}\right)^2 \tag{3}$$

Combining the above 3 equations yields one equation for the flow and head (4) and one equation for the torque (5).

$$g(H_0 - H_3) = gH_R\left(\frac{Q}{\kappa Q_R}\right)^2 + s_D(\omega_t^2 - \omega_{tR}^2) \tag{4}$$

$$\begin{aligned} T &= \rho Q(c_{u1}r_1 - c_{u2}r_2) = \rho Q(t_s - r_2^2\omega_t) \\ t_s &= r_1 c_1 cos(\alpha_1) + r_2 A_z cot(\beta_2) c_1 sin(\alpha_1) \end{aligned} \tag{5}$$

Here t_s is the starting torque when $\omega_t = 0$ [7]. A_z is the inlet area divided by the outlet area perpendicular to the shaft.

Equation (4) and (5) are made non-dimensional to form equation (6) and (7) respectively. A term, η_i, called the incipient efficiency, is included[5][8][12]. This incipient efficiency can be regarded as a general term describing all the irreversible losses in the turbine model. This term is the focus of this paper. It's worth noting that equation (6), (7) and (8) are capable of fully describing a turbine hill chart as long as a suitable η_i is constructed and included.

$$t = q\eta_i(m_s - \psi\varpi) \tag{6}$$

$$h_e = \left(\frac{q}{\kappa}\right)^2 + \sigma(\varpi^2 - 1) \tag{7}$$

$$\tilde{\eta}_h = \frac{t\varpi}{qh_e} = \frac{\eta_i(m_s - \psi\varpi)\varpi}{h_e} = \frac{\eta_i(m_s - \psi\varpi)\varpi}{\left(\frac{q}{\kappa}\right)^2 + \sigma(\varpi^2 - 1)} \tag{8}$$

3. Model behaviour with $\eta_i = 1$

When η_i is set to the numerical constant value 1, then no effect of this terms is included. This is useful because it will show what is missing in the basic model.

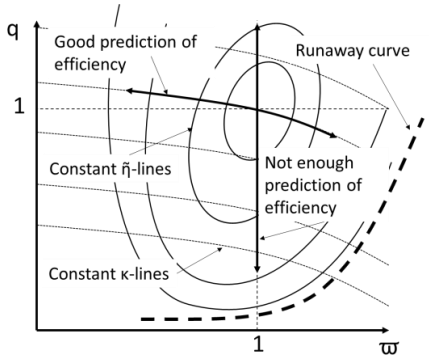

Figure 3.1 Hill chart with constant lines for η̃ and κ, and with ϖ and q as axis. $\eta_i = 1$.

Figure 3.1 shows a sketch of a hill chart. With $\eta_i = 1$, the model is able to give a good prediction of all the variables along the line that represents a constant opening degree, $\kappa = 1$. The reason for this is that along this line, the losses are mostly represented by the turbine running too fast or too slow, and this already included in the basic model from the Euler turbine equation. For a Pelton turbine, this also extends where the flow is different from 1, or $\kappa \neq 1$.

What is missing for Francis turbines, are mainly two different irreversible losses: Losses due to wrong inflow angles, and losses due to spin in the draft tube. Other losses such as friction and turbulence can implicitly be included but are considered minor in comparison.

For a Pelton, the hill chart is already rather good for $\eta_i = 1$, which is not unreasonable knowing that a Pelton does not suffer from wrong inflow angles or swirl in a draft tube. There still are errors compared with measured diagrams, particularly for $q > 1$. This can be explained by too much flow for each bucket to empty before it enters a new jet, and/or the jet becomes too wide for each bucket.

Also worth discussing is when the basic turbine equations in general can be used, even with $\eta_i = 1$. For both Francis and Pelton, the equations can be used "as is", at least when going along the line where $\kappa = 1$. Thus for most grid analysis and stability analysis, the model will be a substantial improvement from using a simple valve equation, as is often used[14][15]. Specifically the model will include the variation of flow with respect to speed, which is important for stability analysis. The model/equations can be used without any knowledge of the actual turbine, other than nominal flow and head[8][9][12].

4. Previous model improvements

Already in Nielsen[6] several first principle improvements of the model were made. Specifically, the losses due to wrong inflow angles and draft tube losses were addressed. Both these losses were modelled as head losses using the same basic equation:

$$h_{loss} = R(Q - Q_c)^2 \tag{9}$$

where

$$Q_c = F(\kappa, \omega) \tag{10}$$

The function, F, itself is an analytical function describing the flow q at BEP. Equation (9) is subtracted from equation (7) to include this in the model. However, several such losses make verifications very difficult, as they are impossible to differentiate from each other. It all becomes very convoluted for practical calculations when model tests or site measurements are needed to find good values for all the Rs, and at the same time they are all related to a head loss.

5. Different forms of η_i

In Nielsen[5] the first use of the *incipient efficiency* was found. This made it possible to clearly divide the "pure" model equations from all the irreversible losses. In addition, the incipient efficiency modified the turbine efficiency instead of the head losses[8][9]. This resulted in the basic equations (6), (7) and (8), and where the incipient efficiency, η_i, is an arbitrary loss function that can take any shape and form. Moreover, it makes the model equations more consistent, in the sense that for any turbine, there should exist an η_i that will make the model equations fit with measurements[9].

As shown in [8][9][11][12], even the simplest and fully generic 1D function of η_i will make huge improvements.

From a more fundamental point of view, this kind of model is interesting. The basic physics is correct, as far as one can be certain, but there are certain loss terms that must be modelled separately because they are not intrinsically a part of the model. This is similar to turbulence modelling in CFD or the Moody diagram in pipe flow. The losses cannot be modelled in the domain of the model, 1D in this case, but they can be approximated by simpler functions that are part of the same domain.

5.1. The simplest η_i

The simplest form of η_i is found in [5][8][11]. It is expressed as:

$$\eta_i = q(2-q) \tag{11}$$

This is a function of the flow, q where: $\eta_i = 0$ for q = 0; $\eta_i = 1$ for q = 1; $\eta_i = 0$ for q = 2. The shape is based on the observation from measurements that the efficiency must go towards zero when q approaches zero[6]. The efficiency must be equal to nominal efficiency when q = 1 (nominal flow), and it should decrease when q > 1. This is shown in the figures below comparing real measurements with the mathematical model.

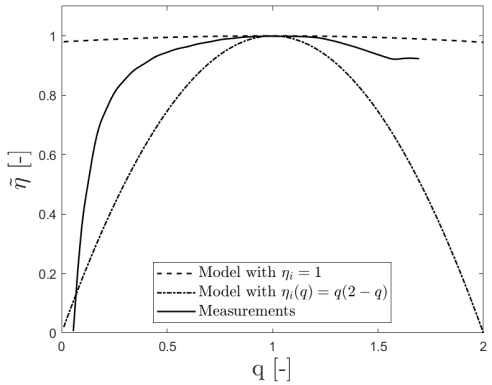

Figure 5.1 Measured vs calculated efficiency for a high head Francis turbine

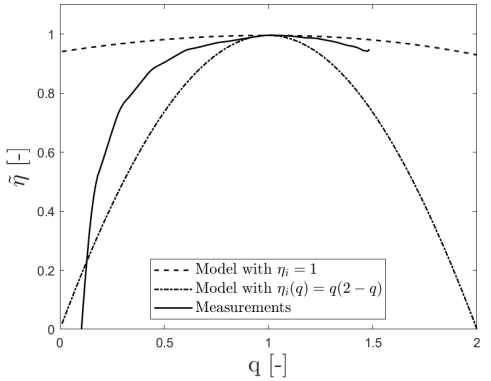

Figure 5.2 Measured vs calculated efficiency for a medium head Francis turbine

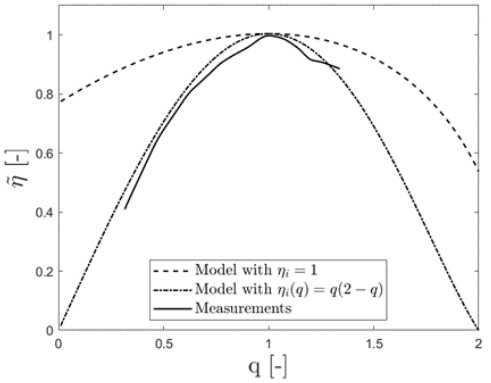

Figure 5.3 Measured vs calculated efficiency for a low head Francis turbine

This very simple generic η_i improves the basic model considerably[8]. However, it is a bit "too generic" for the entire range from low head Francis, Figure 5.3, to high head Francis, Figure 5.1, and further to Pelton[12]. It must be noted that equation (11) is a rather good approximation for low head Francis, Figure 5.2. The curve for $\eta_i = 1$ in the figures above represents the basic model with no irreversible loss terms.

5.2. Best fit incipient efficiency η_i for low head to high head Francis

Assuming that η_i can be approximated by a function of q, then finding the best η_i that fits to several different turbines with different specific speeds, it should be possible to create generic curves that fit with those speeds and interpolate between them.

Further, there are some pure practical considerations that has to be considered. When installing new turbine governors on existing plants, the complete hill chart usually cannot be found. Even if it is found, it is an elaborate process to transfer paper charts to a well-behaved interpolating function for analysis. A more efficient method would be to simply use readily available nominal values, and to build a mathematical model automatically from that.

As shown in [12] this is indeed possible. Measured hill charts from 3 model turbines: high head Francis; medium head Francis; low head Francis, were used to find the best curve for each.

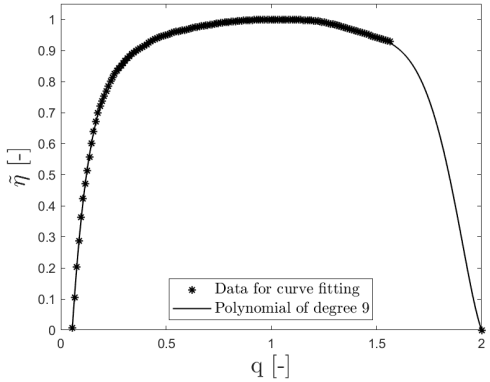

Figure 5.4 High head Francis, best fit of η_i with measured model turbine at nominal speed[12]

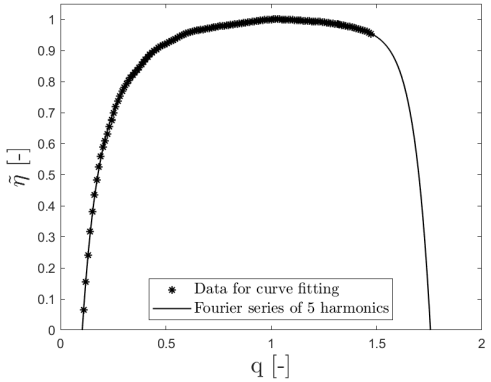

Figure 5.5 Medium head Francis, best fit of η_i with measured model turbine at nominal speed[12]

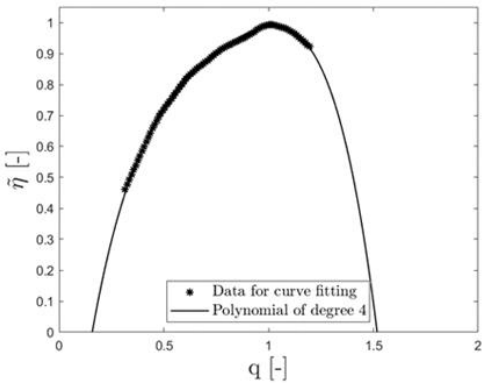

Figure 5.6 Low head Francis, best fit of η_i with measured model turbine at nominal speed[12]

With this method the hill chart will be accurate for all κ and q when $\omega = 1$ and for all ω when $\kappa = 1$. The model can be used for any analysis where $\omega \approx 1$, for instance grid analysis and stability analysis. Since it is also accurate for all ω at $\kappa = 1$, it can be used for simple transient analysis.

The model is slightly inaccurate where both ω and κ are largely different from 1 at the same time. Regarding transient analysis, this would typically be along the curve of the runaway speed, which is important when calculating load rejection. This can be seen comparing Figure 5.7 and Figure 5.8. However, it is worth noting that for κ = 1 or for ω = 1, assuming constant head, h = 1 also for the measured chart, the runaway curve is quite correct in these points.

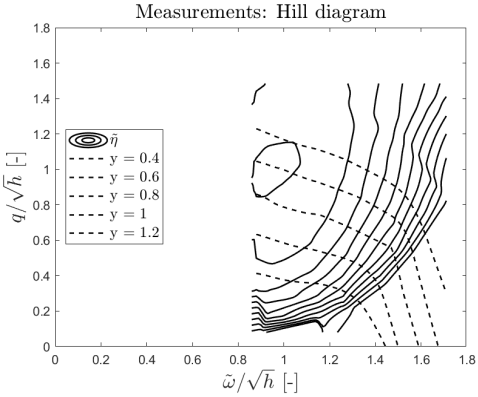

Figure 5.7 Measured hill chart for a medium head Francis

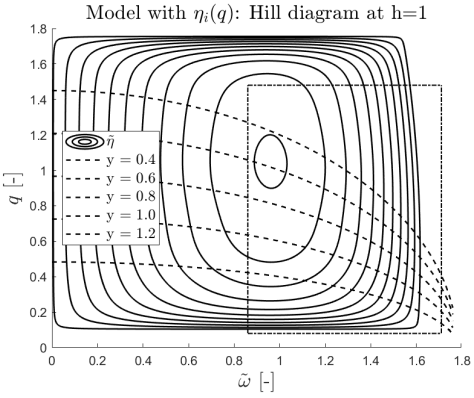

Figure 5.8 Medium head Francis hill chart using the turbine model and η_i from Figure 5.5 and representing the model tests in Figure 5.7

6. Discussion

Using the simplified design procedure as shown in [9] and [12] together with the incipient efficiency as a function of q and speed number, any unknown (Francis) turbine can easily be modelled just using nominal values as flow, speed and head. This is useful for grid analysis and stability analysis of turbine governors, as no additional information is needed[8][12]. The model is easy to linearize, even analytically[9][11].

Data from only 3 model tests were used: low head, medium head and high head Francis. The assumption is therefore that these three specific Francis turbines give representative curves for all other Francis turbines. This may seem as a rather big assumption. However, when studying how turbine characteristics changes with speed number, and noting the fact that the basic equations already include most of these characteristics, this cannot be considered an unreasonable assumption. Still, including many more model tests would of course be one of the next steps further with this model.

More work must be done with the η_i function to make it fully usable for general transient analysis including load rejection, since the runaway curve is not represented well[12].

LVTrans has used the original model[6] with the R functions and Q_c ,Equation (9), with very good results for more than 15 years[1][2][4], although the problem of easily and methodically finding suitable R values for arbitrary turbines do remain. The idea to use an incipient efficiency function of the form:

$$\eta_i = F(q, q_c) = F(q, \kappa, \omega) \tag{12}$$

is therefore tempting and should be investigated. It is believed that simple but effective forms may be found that are adequate for the purpose of anticipating the runaway curve when this is needed.

Linearizing this model is straight forward[9][11]. The block diagram will change slightly in comparison to a traditional diagram. The reason for this can be seen directly from equation (7) and (6) where ω is an independent parameter. This creates the block diagram as shown in Figure 6.1.

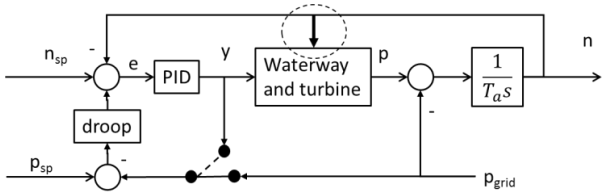

Figure 6.1 Block diagram showing the speed as an input to the "Waterway and turbine" block.

7. Conclusion

The turbine equations with the incipient efficiency seems to be very promising. Most of the shortcomings with the original simple form, equation (11), has been solved by using more elaborate forms as shown in Figure 5.4, Figure 5.5 and Figure 5.6. These new forms do not add any complexity in practical numerical calculations.

This new form of incipient efficiency is accurate around best efficiency point and along the curves of nominal speed and nominal turbine opening in the turbine hill chart. Thus, the turbine equations can be used for grid calculations and stability analysis, as well as simple transient calculations for Francis turbines of literally the full range of speed numbers.

As found in [12] and seen in when comparing Figure 5.7 and Figure 5.8, the new form of incipient efficiency does not give a good representation of the runaway curve for Francis turbines. Further work is needed in this area.

References

[1] Svingen B, 2005, *Application of LabVIEW for dynamic simulation of Hydraulic piping systems,* SIMS 2005 46th Conf. on Simulation and Modeling, 325-333

[2] Svingen B, Brekke T, Skåre P E, Nielsen T, 2006, *Minimizing oscillations and transients for large and rapid changes in power output from hydro power plants by new control algorithms,* 23rd IAHR Symposium

[3] Svingen B, Francke H, 2015, *Large and rapid set-point adjustment of hydro power plants using embedded transient hydraulic simulations of the plant as a model predictive method,* 12th Int. Conf Pressure Surges, BHR Group

[4] Svingen B, 2016, *A predictive controller based on transient simulations for controlling a power plant,* 28th IAHR Symposium on Hydraulic Machinery and Systems

[5] Nielsen T K, 2015, *Simulation model for Francis and Reversible Pump Turbines*, International Journal of Fluid Machinery and Systems Vol. 8 No. 3

[6] Nielsen T K, 1990, *Transient characteristics of hight head Francis turbines*, Dr. Thesis, ISBN 82-7119-242-6

[7] Wylie E B, Streeter, V L, 1993, *Fluid Transients in Systems*, Prentice Hall, ISBN 0-13-322173-3

[8] Svingen B, Nielsen T K, 2018, *First principles approach linear model for hydraulic turbines suitable for use in availale simulation platforms*, 13th International Conference on Pressure Surges, Bordeaux, France, 14th - 16th November 2018

[9] Svingen B, Nielsen TK, *Transient Hydraulic Calculations with a Linear Turbine Model derived from a Nonlinear Synthetic Model*, 29 IAHR Symposium on Hydraulic Machinery and Systems, Kyoto, Japan, November 2018

[10] Storli P, Nielsen T K, *Simulation and discussion of models for hydraulic Francus turbine simulations*, IFAC PapersOnLine, 51(2):109-114, 2018
[11] Reines A F, 2019, *Modelling of hydro power plants*, Project Thesis, Institutt for Termisk Energi og Prosessteknikk. NTNU, Norway
[12] Reines A F, 2020, *Analysis and improvement of mathematical turbine model*, Master Thesis, Institutt for Termisk Energi og Prosessteknikk, NTNU, Norway
[13] Woodward J L, 1968, Hydraulic-turbine transfer function for use in governing studies. Proc. IEE 115, No. 3, 424–426
[14] Wang C, Goa H, 2006, *Effect of detailed hydro turbine models on power system analysis*, PCSE '06, IEEE.
[15] Monuz-Hernandez G A, Mansoor S P, Jones D I, 2013, *Modelling and Controlling Hydropower Plants*, Springer

A2 Mathematical derivation of the characteristic coefficients of the linear model

Model without incipient efficiency

Partial derivation is performed to the following governing equations applicable to a Francis turbine:

$$q \;=\; y\,\sqrt{h-\sigma\,(\tilde{\omega}^2-1)} \;=\; f\,(h,y,\tilde{\omega}) \tag{9.1}$$

$$t \;=\; q\,(m_S-\psi\,\tilde{\omega}) \;=\; g\,(q,y,\tilde{\omega}) \tag{9.2}$$

where

$$m_S \;=\; \xi\,\frac{q}{y}\,(\cos\alpha_1+\tan\alpha_{1R}\;\sin\alpha_1) \;=\; h\,(q,y) \tag{9.3}$$

When differentiating the dependent variables q or t with respect to each of their respective independent variables, the other independent variables are kept constant.

$$a_{11} \;=\; \left.\frac{\partial q}{\partial h}\right|_0 \;=\; \left.\frac{y}{2\,\sqrt{h-\sigma(\tilde{\omega}^2-1)}}\right|_0 \tag{9.4}$$

$$a_{12} \;=\; \left.\frac{\partial q}{\partial y}\right|_0 \;=\; \left.\sqrt{h-\sigma(\tilde{\omega}^2-1)}\right|_0 \tag{9.5}$$

$$a_{13} \;=\; \left.\frac{\partial q}{\partial \tilde{\omega}}\right|_0 \;=\; -\left.\frac{y\,\sigma\,\tilde{\omega}}{\sqrt{h-\sigma(\tilde{\omega}^2-1)}}\right|_0 \tag{9.6}$$

$$a_{21} \;=\; \left.\frac{\partial t}{\partial q}\right|_0 \;=\; (m_S-\psi\,\tilde{\omega}) \;+\; q\,\left.\frac{\partial m_S}{\partial q}\right|_0 \;=\; \left.(2\,m_S-\psi\,\tilde{\omega})\right|_0 \tag{9.7}$$

since

$$\left.\frac{\partial m_S}{\partial q}\right|_0 \;=\; \xi\,\frac{1}{y}\,(\cos\alpha_1 \;+\; \tan\alpha_{1R}\;\sin\alpha_1)\,\Big|_0 \;=\; \left.\frac{m_S}{q}\right|_0 \tag{9.8}$$

$$a_{22} \;=\; \left.\frac{\partial t}{\partial y}\right|_0 \;=\; q\,\left.\frac{\partial m_S}{\partial y}\right|_0 \tag{9.9}$$

where

$$\left.\frac{\partial m_S}{\partial y}\right|_0 = \xi\frac{q}{y^2}\left(-\cos\alpha_1-\tan\alpha_{1R}\sin\alpha_1+y\frac{\partial\alpha_1}{\partial y}(-\sin\alpha_1+\tan\alpha_{1R}\cos\alpha_1)\right)\Bigg|_0 \tag{9.10}$$

where

$$\left.\frac{\partial \alpha_1}{\partial y}\right|_0 = \left.\frac{\partial}{\partial y}\left(\arcsin\left(y\ \sin\alpha_{1R}\right)\right)\right|_0 = \left.\frac{\sin\alpha_{1R}}{\sqrt{1-y^2\ \sin{\alpha_{1R}}^2}}\right|_0 \tag{9.11}$$

$$a_{23} = \left.\frac{\partial t}{\partial \tilde{\omega}}\right|_0 = \left. -\ q\ \psi\ \right|_0 \tag{9.12}$$

Subscript '0' denotes the point of linearization.

Model with general incipient efficiency

Partial derivation is performed to the following governing equations applicable to a Francis turbine:

$$q = y\ \sqrt{h-\sigma\ (\tilde{\omega}^2-1)} = f\ (h, y, \tilde{\omega}) \tag{9.13}$$

$$t = \eta_i(q)\ q\ (m_S - \psi\ \tilde{\omega}) = g\ (q, y, \tilde{\omega}) \tag{9.14}$$

The partial derivatives of the flow equation, a_{11}, a_{12} and a_{13}, are not affected by the implementation of incipient efficiency, only the partial derivatives of the torque equation, a_{21}, a_{22} and a_{23}. The differentiation is performed for an arbitrary incipient efficiency function $\eta_i(q)$, assumed to depend solely on p.u. flow.

$$a_{21} = \left.\frac{\partial t}{\partial q}\right|_0 = \left.\left((m_S - \psi\ \tilde{\omega})\ (q\ \frac{\partial \eta_i}{\partial q} + \eta_i(q)) + q\ \eta_i(q)\frac{\partial m_S}{\partial q}\right)\right|_0 \tag{9.15}$$

$$a_{22} = \left.\frac{\partial t}{\partial y}\right|_0 = \left.\eta_i(q)\ q\ \frac{\partial m_S}{\partial y}\ \right|_0 \tag{9.16}$$

$$a_{23} = \left.\frac{\partial t}{\partial \tilde{\omega}}\right|_0 = \left. -\ \eta_i(q)\ q\ \psi\ \right|_0 \tag{9.17}$$

Subscript '0' denotes the point of linearization.

Characteristic coefficients at best efficiency point (BEP)

At BEP corresponding to rated/ nominal values, $h = q = y = \tilde{\omega} = 1$ holds.
The expressions for m_S, $\frac{\partial m_S}{\partial q}$, $\frac{\partial m_S}{\partial y}$ and $\frac{\partial \alpha_1}{\partial y}$ simplifies to:

$$m_S \Big|_{BEP} = \frac{\xi}{\cos \alpha_{1R}} \tag{9.18}$$

$$\frac{\partial m_S}{\partial q} \Big|_{BEP} = \frac{\xi}{\cos \alpha_{1R}} \tag{9.19}$$

$$\frac{\partial m_S}{\partial y} \Big|_{BEP} = -\frac{\xi}{\cos \alpha_{1R}} \tag{9.20}$$

$$\frac{\partial \alpha_1}{\partial y} \Big|_{BEP} = \tan \alpha_{1R} \tag{9.21}$$

At $q = 1$, any $\eta_i(q)$ expression should be 1 and have first derivative equal to 0. Hence, a_{ij} coefficients will reduce to the same expressions regardless the presence of an η_i curve, the proof is omitted but could easily have been included. For the proposed incipient efficiency functions of Chapter 7, the first derivatives are only approximate equal to 0. This inaccuracy will be disregarded and we may approximate the characteristic values at BEP to be independent of the choice of $\eta_i(q)$ curve. The mathematical expressions for a_{ij} at BEP are summarized below:

	Characteristic values
a_{11}	0.5
a_{12}	1.0
a_{13}	$-\sigma$
a_{21}	$2\,\xi / \cos \alpha_{1R} - \psi$
a_{22}	$-\xi / \cos \alpha_{1R}$
a_{23}	$-\psi$

Table A2.1: Characteristic coefficients of the linear model at best efficiency point.

To obtain numerical values, simply insert σ, ψ, ξ and α_{1R} for the considered turbine into the above expressions.

A3 MATLAB scripts

All relevant MATLAB scripts are delivered as m-files with the master thesis at hand-in. In this way, the reader can run them at his or her own request. As already mentioned, the Excel sheet containing experimental data and model input values from the designs, are not included in order to protect the data from Rainpower. If the reader wants to run any of the model application scripts, just specify the model inputs at the top of the script.

In this Appendix, all script names are presented together with an explanation to what they do. Most scripts were produced in three versions, for the high, medium and low head turbine. These are similar with respect to data handling, inputs/ outputs, calculations, etc., only titles, axis dimensions, legends and query vectors for interpolations may differ. To prevent the Appendix from becoming too long, only scripts which are either general for all turbines, or if such does not exist, specific for the high head one, will be included here. They will be presented in a compact 4x4 format. For more detailed study of any of these MATLAB scripts, the author strongly suggests the reader to open the associated m-file.

Experimental data processing

- Script name: $meas\_characteristics\_highH.m$, $meas\_characteristics\_medH.m$, $meas\_characteristics\_lowH.m$

- Purpose: Plots 14 different characteristic curves from interpolating the Rainpower measurements or any set of experimental data, which are loaded into MATLAB from Excel. Calculates numerically a_{ij} gradients at BEP, and the error of these values with respect to the analytical calculations, which are loaded in from the turbine design Excel sheet. When all *smooth* functions are commented out, "raw" data are being plotted, but the visual difference is minor. However the smoothing is necessary for good accuracy (two decimal places was obtained) of the numerically estimated gradients.

$meas\_characteristics\_highH.m$:

```
clear all
close all
clc

%load data from Rainpower file and sort it
data=table2array(readtable('Hillchart_r3_medBEP.xlsx','Sheet','Høy',...
                           'Range','B2:F257'));    %[etaM Q11 N11 T11
 alfa]
etaM=data(:,1); Q11=data(:,2); N11=data(:,3); T11=data(:,4);
 alfa=data(:,5);

%remove data for negative efficiencies (irrelevant)
positive=find(etaM>=0);
etaM=etaM(positive); Q11=Q11(positive); N11=N11(positive);
 T11=T11(positive); alfa=alfa(positive);

%find rated values according to best efficiency point (BEP)
etaM_R=max(etaM);
pos=find(etaM==etaM_R);
Q11_R=Q11(pos); N11_R=N11(pos); T11_R=T11(pos); alfa_R=alfa(pos);
speednumber=N11_R*(pi/30)*sqrt(Q11_R)/(2*9.82)^(3/4);

%non-dimensionalize by dividing by rated values
eta=etaM./etaM_R; q=Q11./Q11_R; n=N11./N11_R; t=T11./T11_R;
 y=sind(alfa)./sind(alfa_R);

%------------------------------------------------------------------------
% common for plot 1 - 4:
%------------------------------------------------------------------------

y_query=linspace(0.2,1.4,13);               %y=1.0 included
dn=0.01;
nhalf1=linspace(min(n),1,ceil((1-min(n))/dn));
nhalf2=linspace(1,max(n),ceil((max(n)-1)/dn));
n_query=[nhalf1 nhalf2(2:end)];             %n=1.0 included

[Y,N]=meshgrid(y_query,n_query);            %each column in Y const
 opening degree
                                            %each row in N const speed

%------------------------------------------------------------------------
% 1. Flow - speed plot at different opening degrees and const rated
 head
%------------------------------------------------------------------------
F1=scatteredInterpolant(y,n,q,'natural');   %create interpolant that
 fits a surface on the form q=F1(y,n)
Q=F1(Y,N);                                  %evaluate F1 at query
 points specified by Y,N matrices

figure(1)
axis([0 2 0 2])
xlabel('$\tilde{\omega}$ [-]','Interpreter','latex','FontSize',16)
ylabel('q [-]','Interpreter','latex','FontSize',16)
title('Measurements: Flow - speed at
 h=1','Interpreter','latex','FontSize',16)
hold on

%plot one curve per opening degree
for i=1:length(y_query)
    Q(:,i)=smooth(Q(:,i),10);               %comment out for raw data
    txt=['y = ',num2str(y_query(i))];

 plot(N(2:end-1,i),Q(2:end-1,i),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show

% find dq/dn at y=n=q=h=1
pos_n=find(n_query==1);
pos_y=find(y_query==1);
q_test=Q(pos_n,pos_y)                       %should be 1

% using a central difference scheme
dqdn=(Q(pos_n+1,pos_y)-Q(pos_n-1,pos_y))/(N(pos_n+1,pos_y)-
N(pos_n-1,pos_y));
dqdn_2=(Q(pos_n+2,pos_y)-Q(pos_n-2,pos_y))/(N(pos_n+2,pos_y)-
N(pos_n-2,pos_y));
dqdn_3=(Q(pos_n+3,pos_y)-Q(pos_n-3,pos_y))/(N(pos_n+3,pos_y)-
N(pos_n-3,pos_y));

% using inbuildt gradient function
f=gradient(Q(:,pos_y));
dqdn_check=f(pos_n)/(0.5*(n_query(pos_n+1)-n_query(pos_n-1)));

%------------------------------------------------------------------------
% 2. Torque - speed plot at different opening degrees and const rated
 head
%------------------------------------------------------------------------
F2=scatteredInterpolant(y,n,t,'natural');   %create interpolant that
 fits a surface on the form t=F2(y,n)
T=F2(Y,N);                                  %evaluate F2 at query
 points specified by Y,N matrices

figure(2)
axis([0 2 0 2])
xlabel('$\tilde{\omega}$ [-]','Interpreter','latex','FontSize',16)
ylabel('t [-]','Interpreter','latex','FontSize',16)
title('Measurements: Torque - speed at
 h=1','Interpreter','latex','FontSize',16)
hold on

%plot one curve per opening degree
for i=1:length(y_query)
    T(:,i)=smooth(T(:,i),10);               %comment out for raw data
```

```
    txt=['y = ',num2str(y_query(i))];

 plot(N(2:end-1,i),T(2:end-1,i),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show

% find dt/dn at y=n=t=h=1
pos_n=find(n_query==1);
pos_y=find(y_query==1);
t_test=T(pos_n,pos_y)                          %should be 1

% using a central difference scheme
dtdn=(T(pos_n+1,pos_y)-T(pos_n-1,pos_y))/(N(pos_n+1,pos_y)-
N(pos_n-1,pos_y));
dtdn_2=(T(pos_n+2,pos_y)-T(pos_n-2,pos_y))/(N(pos_n+2,pos_y)-
N(pos_n-2,pos_y));
dtdn_3=(T(pos_n+3,pos_y)-T(pos_n-3,pos_y))/(N(pos_n+3,pos_y)-
N(pos_n-3,pos_y));

% using inbuildt gradient function
f=gradient(T(:,pos_y));
dtdn_check=f(pos_n)/(0.5*(n_query(pos_n+1)-n_query(pos_n-1)));

%-------------------------------------------------------------------------
% 3. Mech. power - speed plot at different opening degrees and const
 rated head
%-------------------------------------------------------------------------
P=T.*N;

figure(3)
axis([0 2 0 1.6])
xlabel('$\tilde{\omega}$ [-]','Interpreter','latex','FontSize',16)
ylabel('p = t*$\tilde{\omega}$
 [-]','Interpreter','latex','FontSize',16)
title('Measurements: Output power - speed at
 h=1','Interpreter','latex','FontSize',16)
hold on

%plot one curve per opening degree
for i=1:length(y_query)
    P(:,i)=smooth(P(:,i));                     %comment out for raw data
    txt=['y = ',num2str(y_query(i))];

 plot(N(2:end-1,i),P(2:end-1,i),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show
```

```
%-------------------------------------------------------------------------
% 4. Efficiency - speed plot at different opening degrees and const
 rated head
%-------------------------------------------------------------------------
F4=scatteredInterpolant(y,n,eta,'natural'); %create interpolant that
 fits a surface on the form eta=F4(y,n)
ETA=F4(Y,N);                               %evaluate F4 at query
 points specified by Y,N matrices

figure(4)
axis([0 2 0 1.1])
xlabel('$\tilde{\omega}$ [-]','Interpreter','latex','FontSize',16)
ylabel('$\tilde{\eta}$ [-]','Interpreter','latex','FontSize',16)
title('Measurements: Efficiency - speed at
 h=1','Interpreter','latex','FontSize',16)
hold on

%plot one curve per opening degree
for i=1:length(y_query)
    ETA(:,i)=smooth(ETA(:,i));             %comment out for raw data
    txt=['y = ',num2str(y_query(i))];

 plot(N(2:end-1,i),ETA(2:end-1,i),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show


%-------------------------------------------------------------------------
% common for plot 5 - 8:
%-------------------------------------------------------------------------

n_query=linspace(0.8,1.3,11);              %n=1.0 included
dy=0.01;
yhalf1=linspace(min(y),1,ceil((1-min(y))/dy));
yhalf2=linspace(1,max(y),ceil((max(y)-1)/dy));
y_query=[yhalf1 yhalf2(2:end)];            %y=1.0 included

[N,Y]=meshgrid(n_query,y_query);           %each column in N const
 speed
                                           %each row in Y const
 opening degree

%-------------------------------------------------------------------------
% 5. Flow - opening degree plot at different speeds and const rated
 head
%-------------------------------------------------------------------------
F5=scatteredInterpolant(n,y,q,'natural');  %create interpolant that
 fits a surface on the form q=F5(n,y)
Q=F5(N,Y);                                 %evaluate F5 at query
 points specified by N,Y matrices
```

```
figure(5)
axis([0 2 0 2])
xlabel('y [-]','Interpreter','latex','FontSize',16)
ylabel('q [-]','Interpreter','latex','FontSize',16)
title('Measurements: Flow - opening degree at
 h=1','Interpreter','latex','FontSize',16)
hold on

%plot one curve per speed
for i=2:length(n_query)
    Q(:,i)=smooth(Q(:,i),10);              %comment out for raw data
    txt=['$\tilde{\omega} = $',num2str(n_query(i))];

 plot(Y(2:end-1,i),Q(2:end-1,i),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show

% find dq/dy at n=y=q=h=1
pos_y=find(y_query==1);
pos_n=find(n_query==1);
q_test=Q(pos_y,pos_n)                          %should be 1

% using a central difference scheme
dqdy=(Q(pos_y+1,pos_n)-Q(pos_y-1,pos_n))/(Y(pos_y+1,pos_n)-
Y(pos_y-1,pos_n));
dqdy_2=(Q(pos_y+2,pos_n)-Q(pos_y-2,pos_n))/(Y(pos_y+2,pos_n)-
Y(pos_y-2,pos_n));
dqdy_3=(Q(pos_y+3,pos_n)-Q(pos_y-3,pos_n))/(Y(pos_y+3,pos_n)-
Y(pos_y-3,pos_n));

% using inbuildt gradient function
f=gradient(Q(:,pos_n));
dqdy_check=f(pos_y)/(0.5*(y_query(pos_y+1)-y_query(pos_y-1)));

%-------------------------------------------------------------------------
% 6. Torque - opening degree plot at different speeds and const rated
 head
%-------------------------------------------------------------------------
F6=scatteredInterpolant(n,y,t,'natural');  %create interpolant that
 fits a surface on the form t=F6(n,y)
T=F6(N,Y);                                 %evaluate F6 at query
 points specified by N,Y matrices

figure(6)
axis([0 2 0 2])
xlabel('y [-]','Interpreter','latex','FontSize',16)
ylabel('t [-]','Interpreter','latex','FontSize',16)
title('Measurements: Torque - opening degree at
 h=1','Interpreter','latex','FontSize',16)
hold on
```

```
%plot one curve per speed
for i=2:length(n_query)
    T(:,i)=smooth(T(:,i),8);                   %comment out for raw data
    txt=['$\tilde{\omega} = $',num2str(n_query(i))];

 plot(Y(2:end-2,i),T(2:end-2,i),'-','LineWidth',1,'DisplayName',txt);

 set(legend,'Interpreter','latex','FontSize',10,'Location','northwest')
end

hold off
legend show

% find dt/dy at n=y=t=h=1
pos_y=find(y_query==1);
pos_n=find(n_query==1);
t_test=T(pos_y,pos_n)                          %should be 1

% using a central difference scheme
dtdy=(T(pos_y+1,pos_n)-T(pos_y-1,pos_n))/(Y(pos_y+1,pos_n)-
Y(pos_y-1,pos_n));
dtdy_2=(T(pos_y+2,pos_n)-T(pos_y-2,pos_n))/(Y(pos_y+2,pos_n)-
Y(pos_y-2,pos_n));
dtdy_3=(T(pos_y+3,pos_n)-T(pos_y-3,pos_n))/(Y(pos_y+3,pos_n)-
Y(pos_y-3,pos_n));

% using inbuildt gradient function
f=gradient(T(:,pos_n));
dtdy_check=f(pos_y)/(0.5*(y_query(pos_y+1)-y_query(pos_y-1)));

%-------------------------------------------------------------------------
% 7. Mech. power - opening degree plot at different speeds and const
 rated head
%-------------------------------------------------------------------------
P=T.*N;

figure(7)
axis([0 2 0 2])
xlabel('y [-]','Interpreter','latex','FontSize',16)
ylabel('p = t*$\tilde{\omega}$
 [-]','Interpreter','latex','FontSize',16)
title('Measurements: Output power - opening degree at
 h=1','Interpreter','latex','FontSize',16)
hold on

%plot one curve per speed
for i=2:length(n_query)
    P(:,i)=smooth(P(:,i));                     %comment out for raw data
    txt=['$\tilde{\omega} = $',num2str(n_query(i))];

 plot(Y(2:end-1,i),P(2:end-1,i),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end
```

```
hold off
legend show

%-----------------------------------------------------------------------
% 8. Efficiency - opening degree plot at different speeds and const
 rated head
%-----------------------------------------------------------------------
F8=scatteredInterpolant(n,y,eta,'natural'); %create interpolant that
 fits a surface on the form eta=F8(n,y)
ETA=F8(N,Y);                                %evaluate F8 at query
 points specified by N,Y matrices

figure(8)
axis([0 2 0 1.1])
xlabel('y [-]','Interpreter','latex','FontSize',16)
ylabel('$\tilde{\eta}$ [-]','Interpreter','latex','FontSize',16)
title('Measurements: Efficiency - opening degree at
 h=1','Interpreter','latex','FontSize',16)
hold on

%plot one curve per speed
for i=2:length(n_query)
    ETA(:,i)=smooth(ETA(:,i));              %comment out for raw data
    txt=['$\tilde{\omega} = $',num2str(n_query(i))];

 plot(Y(2:end-1,i),ETA(2:end-1,i),'-','LineWidth',1,'DisplayName',txt);

 set(legend,'Interpreter','latex','FontSize',10,'Location','southeast')
end

hold off
legend show


%-----------------------------------------------------------------------
% common for plot 9 - 11:
%-----------------------------------------------------------------------

n_query=linspace(0.8,1.3,11);               %n=1.0 included
dq=0.005;
qhalf1=linspace(min(q),1,ceil((1-min(q))/dq));
qhalf2=linspace(1,max(q),ceil((max(q)-1)/dq));
q_query=[qhalf1 qhalf2(2:end)];             %q=1.0 included

[N,Q]=meshgrid(n_query,q_query);            %each column in N const
 speed
                                            %each row in Q const flow

%-----------------------------------------------------------------------
% 9. Torque - flow plot at different speeds and const rated head
%-----------------------------------------------------------------------
F9=scatteredInterpolant(n,q,t,'natural');   %create interpolant that
 fits a surface on the form t=F9(n,q)
```

```
T=F9(N,Q);                                  %evaluate F9 at query
 points specified by N,Q matrices

figure(9)
axis([0 2 0 2])
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('t [-]','Interpreter','latex','FontSize',16)
title('Measurements: Torque - flow at
 h=1','Interpreter','latex','FontSize',16)
hold on

%plot one curve per speed
for i=2:length(n_query)
    T(:,i)=smooth(T(:,i));                  %comment out for raw data
    txt=['$\tilde{\omega} = $',num2str(n_query(i))];

 plot(Q(2:end-1,i),T(2:end-1,i),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show

%-----------------------------------------------------------------------
% 10. Mech. power - flow plot at different speeds and const rated head
%-----------------------------------------------------------------------
P=T.*N;

figure(10)
axis([0 2 0 2])
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('p = t*$\tilde{\omega}$
 [-]','Interpreter','latex','FontSize',16)
title('Measurements: Output power - flow at
 h=1','Interpreter','latex','FontSize',16)
hold on

%plot one curve per speed
for i=2:length(n_query)
    P(:,i)=smooth(P(:,i));                  %comment out for raw data
    txt=['$\tilde{\omega} = $',num2str(n_query(i))];

 plot(Q(2:end-1,i),P(2:end-1,i),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show

%-----------------------------------------------------------------------
% 11. Efficiency - flow plot at different speeds and const rated head
%-----------------------------------------------------------------------
F11=scatteredInterpolant(n,q,eta,'natural'); %create interpolant that
 fits a surface on the form eta=F11(n,q)
```

```
ETA=F11(N,Q);                               %evaluate F11 at query
 points specified by N,Q matrices

figure(11)
axis([0 2 0 1.1])
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('$\tilde{\eta}$ [-]','Interpreter','latex','FontSize',16)
title('Measurements: Efficiency - flow at
 h=1','Interpreter','latex','FontSize',16)
hold on

%plot one curve per speed
for i=2:length(n_query)
    ETA(:,i)=smooth(ETA(:,i));              %comment out for raw data
    txt=['$\tilde{\omega} = $',num2str(n_query(i))];

 plot(Q(2:end-1,i),ETA(2:end-1,i),'-','LineWidth',1,'DisplayName',txt);

 set(legend,'Interpreter','latex','FontSize',10,'Location','southeast')
end

hold off
legend show


%-----------------------------------------------------------------------
% common for plot 12 - 14:
%-----------------------------------------------------------------------

h=n.^(-2);                                  %H/H_R = (N_11R/N_11)^2 =
 1/n^2, for n=n_R
q=q./n;                                     %Q/Q_R = sqrt(h)*Q_11/
Q_11R = (N_11R/N_11)*(Q_11/Q_11R) = (1/n)*q, for n=n_R
t=t.*n.^(-2);                               %T/T_R = h*T_11/T_11R =
 (N_11R/N_11)^2*(T_11/T_11R) = (1/n^2)*t, for n=n_R

y_query=linspace(0.2,1.4,13);               %y=1.0 included
dh=0.01;
hhalf1=linspace(min(h),1,ceil((1-min(h))/dh));
hhalf2=linspace(1,max(h),ceil((max(h)-1)/dh));
h_query=[hhalf1 hhalf2(2:end)];             %h=1.0 included

[Y,H]=meshgrid(y_query,h_query);            %each column in Y const
 opening degree
                                            %each row in H const head

%-----------------------------------------------------------------------
% 12. Flow - head plot at different opening degrees and const rated
 speed
%-----------------------------------------------------------------------
F12=scatteredInterpolant(y,h,q,'natural');  %create interpolant that
 fits a surface on the form q=F12(y,h)
Q=F12(Y,H);                                 %evaluate F12 at query
 points specified by Y,H matrices
```

```
figure(12)
axis([0 2 0 2])
xlabel('h [-]','Interpreter','latex','FontSize',16)
ylabel('q [-]','Interpreter','latex','FontSize',16)
title('Measurements: Flow - head at $\tilde{\omega} =
 1$','Interpreter','latex','FontSize',16)
hold on

%plot one curve per opening degree
for i=1:length(y_query)
    Q(:,i)=smooth(Q(:,i),8);                %comment out for raw data
    txt=['y = ',num2str(y_query(i))];

 plot(H(2:end-2,i),Q(2:end-2,i),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show

% find dq/dh at y=h=q=n=1
pos_h=find(h_query==1);
pos_y=find(y_query==1);
q_test=Q(pos_h,pos_y)                       %should be 1

% using a central difference scheme
dqdh=(Q(pos_h+1,pos_y)-Q(pos_h-1,pos_y))/(H(pos_h+1,pos_y)-
H(pos_h-1,pos_y));
dqdh_2=(Q(pos_h+2,pos_y)-Q(pos_h-2,pos_y))/(H(pos_h+2,pos_y)-
H(pos_h-2,pos_y));
dqdh_3=(Q(pos_h+3,pos_y)-Q(pos_h-3,pos_y))/(H(pos_h+3,pos_y)-
H(pos_h-3,pos_y));

% using inbuildt gradient function
f=gradient(Q(:,pos_y));
dqdh_check=f(pos_h)/(0.5*(h_query(pos_h+1)-h_query(pos_h-1)));

%-----------------------------------------------------------------------
% 13. Torque - head plot at different opening degrees and const rated
 speed
%-----------------------------------------------------------------------
F13=scatteredInterpolant(y,h,t,'natural');  %create interpolant that
 fits a surface on the form t=F13(y,h)
T=F13(Y,H);                                 %evaluate F13 at query
 points specified by Y,H matrices

figure(13)
axis([0 2 0 2.5])
xlabel('h [-]','Interpreter','latex','FontSize',16)
ylabel('t [-]','Interpreter','latex','FontSize',16)
title('Measurements: Torque - head at $\tilde{\omega} =
 1$','Interpreter','latex','FontSize',16)
hold on
```

```
%plot one curve per opening degree
for i=1:length(y_query)
    T(:,i)=smooth(T(:,i),0);                %comment out for raw data
    txt=['y = ',num2str(y_query(i))];

 plot(H(2:end-1,i),T(2:end-1,i),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show

% find dt/dh at y=h=t=n=1
pos_h=find(h_query==1);
pos_y=find(y_query==1);
t_test=T(pos_h,pos_y)                       %should be 1

% using a central difference scheme
dtdh=(T(pos_h+1,pos_y)-T(pos_h-1,pos_y))/(H(pos_h+1,pos_y)-
H(pos_h-1,pos_y));
dtdh_2=(T(pos_h+2,pos_y)-T(pos_h-2,pos_y))/(H(pos_h+2,pos_y)-
H(pos_h-2,pos_y));
dtdh_3=(T(pos_h+3,pos_y)-T(pos_h-3,pos_y))/(H(pos_h+3,pos_y)-
H(pos_h-3,pos_y));

% using inbuildt gradient function
f=gradient(T(:,pos_y));
dtdh_check=f(pos_h)/(0.5*(h_query(pos_h+1)-h_query(pos_h-1)));

%-------------------------------------------------------------------------
% 14. Efficiency - head plot at different opening degrees and const
 rated speed
%-------------------------------------------------------------------------
F14=scatteredInterpolant(y,h,eta,'natural'); %create interpolant that
 fits a surface on the form eta=F14(y,h)
ETA=F14(Y,H);                               %evaluate F14 at query
 points specified by Y,H matrices

figure(14)
axis([0 2 0 1.1])
xlabel('h [-]','Interpreter','latex','FontSize',16)
ylabel('$\tilde{\eta}$ [-]','Interpreter','latex','FontSize',16)
title('Measurements: Efficiency - head at $\tilde{\omega} =
 1$','Interpreter','latex','FontSize',16)
hold on

%plot one curve per opening degree
for i=1:length(y_query)
    ETA(:,i)=smooth(ETA(:,i));              %comment out for raw data
    txt=['y = ',num2str(y_query(i))];

 plot(H(2:end-1,i),ETA(2:end-1,i),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show


%-------------------------------------------------------------------------
% Finding the a_ij characteristics at BEP and their error
%-------------------------------------------------------------------------
a_11=dqdh;
a_12=dqdy;
a_13=dqdn;
a_21=dtdh/a_11;
a_22=dtdy - a_21*a_12;
a_23=dtdn - a_21*a_13;

a_11=round(a_11,2); a_12=round(a_12,2); a_13=round(a_13,2);
 a_21=round(a_21,2); a_22=round(a_22,2); a_23=round(a_23,2);

check=a_21+a_22                             %should be -1

% error of analytical a_ij wrt measurements
a_ij=table2array(readtable('Hillchart_r3_medBEP.xlsx','Sheet','Turbin_design',
                           'Range','U68:U74'));
err_a_11=abs(round(a_ij(1),2)-a_11);
err_a_12=abs(round(a_ij(2),2)-a_12);
err_a_13=abs(round(a_ij(3),2)-a_13);
err_a_21=abs(round(a_ij(4),2)-a_21);
err_a_22=abs(round(a_ij(5),2)-a_22);
err_a_23=abs(round(a_ij(6),2)-a_23);
```

Model application

- Script name: $characteristics\_model\_no\_eta\_i.m$
- Purpose: Reads model input column from the turbine design Excel sheet, and plots 14 different characteristic curves using the model equations **without** η_i. It calculates also a_{ij} coefficients at BEP based on gradients from the plots. This script was the starting point (base script) for all model simulations with the nonlinear turbine model **without** η_i.
- Script name: $characteristics\_model\_with\_eta\_i.m$
- Purpose: Reads model input column from the turbine design Excel sheet, and plots 14 different characteristic curves using the model equations **with a general** η_i. The $\eta_i(q)$ function must be specified by commenting out all the other functions one do not want to apply, at the bottom of this script. Before running, the correct $eta\_i\_xxx.m$ script must be run first to set all necessary curve coefficients as global variables. It calculates also a_{ij} coefficients at BEP based on gradients from the plots. This script was the starting point (base script) for all model simulations with the nonlinear turbine model **with a general** η_i.

Based on these two scripts, *specific* versions were constructed. These are custom-made to the turbine of subject, such that legends, axes, query vectors, etc. match and facilitate comparison to the experimental data. The design column input is kept constant.

- Script name: $characteristics\_model\_no\_eta\_i\_highH.m$, $characteristics\_model\_no\_eta\_i\_medH.m$, $characteristics\_model\_no\_eta\_i\_lowH.m$
- Purpose: Specific versions of $characteristics\_model\_no\_eta\_i.m$ for high, medium and low head turbine, respectively.
- Script name: $characteristics\_model\_with\_eta\_i\_highH.m$, $characteristics\_model\_with\_eta\_i\_medH.m$, $characteristics\_model\_with\_eta\_i\_lowH.m$
- Purpose: Specific versions of $characteristics\_model\_with\_eta\_i.m$ for high, medium and low head turbine, respectively.

characteristics_model_no_eta_i.m:

```
% Non-linear turbine model without eta_i implementation - general
 version

clear all
close all
clc

% Design parameters from excel sheet - choose turbine by specifying
 column
 design=table2array(readtable('Hillchart_r3_medBEP.xlsx','Sheet',...
                              'Turbin_design','Range','U9:U80'));

% necessary model input:
 alfa_1R=design(49,1); sigma=design(53,1); psi=design(54,1);
 xi=design(56,1);

%-------------------------------------------------------------------
% common for plot 1 - 4:
%-------------------------------------------------------------------

omega_query=linspace(0,1.8,181);                    %omega=1 included
y_query=linspace(0.5,1.5,11);                       %y=1 included
h=1;

%-------------------------------------------------------------------
% 1. Flow - speed plot at different opening degrees and const rated
 head
%-------------------------------------------------------------------
figure(1)
axis([0 2 0 2])
xlabel('$\tilde{\omega}$ [-]','Interpreter','latex','FontSize',16)
ylabel('q [-]','Interpreter','latex','FontSize',16)
title('Model: Flow - speed at
 h=1','Interpreter','latex','FontSize',16)
hold on

q=zeros(length(y_query),length(omega_query));       %each row const
 opening, each column const speed
for i=1:length(y_query)
    q(i,:)=y_query(i)*sqrt(h-sigma*(omega_query.^2-1));

    %plot one curve per opening degree
    txt=['y = ',num2str(y_query(i))];
    plot(omega_query,q(i,:),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',12)
end

hold off
legend show

% find dq/domega at q=omega=y=h=1
pos_omega=find(omega_query==1);
pos_y=find(y_query==1);
q_test=q(pos_y,pos_omega)                           %should be 1

% using a central difference scheme
dqdomega=(q(pos_y,pos_omega+1)-q(pos_y,pos_omega-1))/
(omega_query(pos_omega+1)-omega_query(pos_omega-1));

% using inbuildt gradient function
f=gradient(q(pos_y,:));
dqdomega_check=f(pos_omega)/(0.5*(omega_query(pos_omega+1)-
omega_query(pos_omega-1)));

%-------------------------------------------------------------------
% 2. Torque - speed plot at different opening degrees and const rated
 head
%-------------------------------------------------------------------
figure(2)
axis([0 2 0 2])
xlabel('$\tilde{\omega}$ [-]','Interpreter','latex','FontSize',16)
ylabel('t [-]','Interpreter','latex','FontSize',16)
title('Model: Torque - speed at
 h=1','Interpreter','latex','FontSize',16)
hold on

t=zeros(length(y_query),length(omega_query));       %each row const
 opening, each column const speed
for i=1:length(y_query)
    alfa_1=asin(y_query(i)*sin(alfa_1R*pi/180));    %radians
    m_S=xi*q(i,:)./
y_query(i)*(cos(alfa_1)+tan(alfa_1R*pi/180)*sin(alfa_1));
    t(i,:)=q(i,:).*(m_S - psi.*omega_query);

    %plot one curve per opening degree
    txt=['y = ',num2str(y_query(i))];
    plot(omega_query,t(i,:),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',12)
end

hold off
legend show

% find dt/domega at t=omega=y=h=1
pos_omega=find(omega_query==1);
pos_y=find(y_query==1);
t_test=t(pos_y,pos_omega)                           %should be 1

% using a central difference scheme
dtdomega=(t(pos_y,pos_omega+1)-t(pos_y,pos_omega-1))/
(omega_query(pos_omega+1)-omega_query(pos_omega-1));

% using inbuildt gradient function
f=gradient(t(pos_y,:));
dtdomega_check=f(pos_omega)/(0.5*(omega_query(pos_omega+1)-
omega_query(pos_omega-1)));
```

```
%-------------------------------------------------------------------
% 3. Mech. power - speed plot at different opening degrees and const
 rated head
%-------------------------------------------------------------------
figure(3)
axis([0 2 0 1.6])
xlabel('$\tilde{\omega}$ [-]','Interpreter','latex','FontSize',16)
ylabel('p = t*$\tilde{\omega}$
 [-]','Interpreter','latex','FontSize',16)
title('Model: Output power - speed at
 h=1','Interpreter','latex','FontSize',16)
hold on

p=zeros(length(y_query),length(omega_query));       %each row const
 opening, each column const speed
for i=1:length(y_query)
    p(i,:)=t(i,:).*omega_query;

    %plot one curve per opening degree
    txt=['y = ',num2str(y_query(i))];
    plot(omega_query,p(i,:),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',12)
end

hold off
legend show

%-------------------------------------------------------------------
% 4. Efficiency - speed plot at different opening degrees and const
 rated head
%-------------------------------------------------------------------
figure(4)
axis([0 2 0 1.1])
xlabel('$\tilde{\omega}$ [-]','Interpreter','latex','FontSize',16)
ylabel('$\tilde{\eta}$ [-]','Interpreter','latex','FontSize',16)
title('Model: Efficiency - speed at
 h=1','Interpreter','latex','FontSize',16)
hold on

eta=zeros(length(y_query),length(omega_query));     %each row const
 opening, each column const speed
for i=1:length(y_query)

    eta(i,:)=(t(i,:).*omega_query)./(q(i,:).*1);    %since h=1

    %plot one curve per opening degree
    txt=['y = ',num2str(y_query(i))];
    plot(omega_query,eta(i,:),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',12)
end

hold off
legend show
```

```
%-------------------------------------------------------------------
% common for plot 5 - 8:
%-------------------------------------------------------------------

y_query=linspace(0,1.6,81);                         %y=1 included
omega_query=linspace(0.5,1.5,11);                   %omega=1 included
h=1;

%-------------------------------------------------------------------
% 5. Flow - opening degree plot at different speeds and const rated
 head
%-------------------------------------------------------------------
figure(5)
axis([0 2 0 2])
xlabel('y [-]','Interpreter','latex','FontSize',16)
ylabel('q [-]','Interpreter','latex','FontSize',16)
title('Model: Flow - opening degree at
 h=1','Interpreter','latex','FontSize',16)
hold on

q=zeros(length(y_query),length(omega_query));       %each row const
 opening, each column const speed
for j=1:length(omega_query)
    q(:,j)=y_query.*sqrt(h-sigma*(omega_query(j)^2-1));

    %plot one curve per speed
    txt=['$\tilde{\omega} = $',num2str(omega_query(j))];
    plot(y_query,q(:,j),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',12)
end

hold off
legend show

% find dq/dy at q=y=omega=h=1
pos_y=find(y_query==1);
pos_omega=find(omega_query==1);
q_test=q(pos_y,pos_omega)                           %should be 1

% using a central difference scheme
dqdy=(q(pos_y+1,pos_omega)-q(pos_y-1,pos_omega))/(y_query(pos_y+1)-
y_query(pos_y-1));

% using inbuildt gradient function
f=gradient(q(:,pos_omega));
dqdy_check=f(pos_y)/(0.5*(y_query(pos_y+1)-y_query(pos_y-1)));

%-------------------------------------------------------------------
% 6. Torque - opening degree plot at different speeds and const rated
 head
%-------------------------------------------------------------------
figure(6)
axis([0 2 0 2])
xlabel('y [-]','Interpreter','latex','FontSize',16)
```

```
ylabel('t [-]','Interpreter','latex','FontSize',16)
title('Model: Torque - opening degree at
 h=1','Interpreter','latex','FontSize',16)
hold on

t=zeros(length(y_query),length(omega_query));        %each row const
 opening, each column const speed
alfa_1=asin(y_query.*sin(alfa_1R*pi/180));           %radians

for j=1:length(omega_query)
    m_S=xi*((q(:,j)').^2
y_query).*(cos(alfa_1)+tan(alfa_1R*pi/180)*sin(alfa_1));
    t(:,j)=(q(:,j)').*(m_S - psi.*omega_query(j));

    %plot one curve per speed
    txt=['$\tilde{\omega} = $',num2str(omega_query(j))];
    plot(y_query,t(:,j),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',12)
end

hold off
legend show

% find dt/dy at t=y=omega=h=1
pos_omega=find(omega_query==1);
pos_y=find(y_query==1);
t_test=t(pos_y,pos_omega)                            %should be 1

% using a central difference scheme
dtdy=(t(pos_y+1,pos_omega)-t(pos_y-1,pos_omega))/(y_query(pos_y+1)-
y_query(pos_y-1));

% using inbuildt gradient function
f=gradient(t(:,pos_omega));
dtdy_check=f(pos_y)/(0.5*(y_query(pos_y+1)-y_query(pos_y-1)));

%-------------------------------------------------------------------------
% 7. Mech. power - opening degree plot at different speeds and const
 rated head
%-------------------------------------------------------------------------
figure(7)
axis([0 2 0 2])
xlabel('y [-]','Interpreter','latex','FontSize',16)
ylabel('p = t*$\tilde{\omega}$
 [-]','Interpreter','latex','FontSize',16)
title('Model: Output power - opening degree at
 h=1','Interpreter','latex','FontSize',16)
hold on

p=zeros(length(y_query),length(omega_query));        %each row const
 opening, each column const speed

for j=1:length(omega_query)
    p(:,j)=t(:,j).*omega_query(j);

    %plot one curve per speed
    txt=['$\tilde{\omega} = $',num2str(omega_query(j))];
    plot(y_query,p(:,j),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',12)
end

hold off
legend show

%-------------------------------------------------------------------------
% 8. Efficiency - opening degree plot at different speeds and const
 rated head
%-------------------------------------------------------------------------
figure(8)
axis([0 2 0 1.1])
xlabel('y [-]','Interpreter','latex','FontSize',16)
ylabel('$\tilde{\eta}$ [-]','Interpreter','latex','FontSize',16)
title('Model: Efficiency - opening degree at
 h=1','Interpreter','latex','FontSize',16)
hold on

eta=zeros(length(y_query),length(omega_query));      %each row const
 opening, each column const speed
for j=1:length(omega_query)

    eta(:,j)=(t(:,j).*omega_query(j))./(q(:,j).*1); %since h=1

    %plot one curve per speed
    txt=['$\tilde{\omega} = $',num2str(omega_query(j))];
    plot(y_query,eta(:,j),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',12)
end

hold off
legend show

%-------------------------------------------------------------------------
% common for plot 9 - 11:
%-------------------------------------------------------------------------

q_query=linspace(0,1.8,181);                         %q=1 included
omega_query=linspace(0.5,1.5,11);                    %omega=1 included
h=1;

%-------------------------------------------------------------------------
% 9. Torque - flow plot at different speeds and const rated head
%-------------------------------------------------------------------------
figure(9)
axis([0 2 0 2])
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('t [-]','Interpreter','latex','FontSize',16)
title('Model: Torque - flow at
 h=1','Interpreter','latex','FontSize',16)
hold on

t=zeros(length(q_query),length(omega_query));        %each row const
 flow, each column const speed
for j=1:length(omega_query)

    qdivy=sqrt(h-sigma*(omega_query(j)^2-1));
    y=q_query./qdivy;
    alfa_1=asin(y.*sin(alfa_1R*pi/180));             %radians
    m_S=xi*qdivy*(cos(alfa_1)+tan(alfa_1R*pi/180)*sin(alfa_1));

    t(:,j)=q_query.*(m_S - psi.*omega_query(j));

    %plot one curve per speed
    txt=['$\tilde{\omega} = $',num2str(omega_query(j))];
    plot(q_query,t(:,j),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',12)
end

hold off
legend show

%-------------------------------------------------------------------------
% 10. Mech. power - flow plot at different speeds and const rated head
%-------------------------------------------------------------------------
figure(10)
axis([0 2 0 2])
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('p = t*$\tilde{\omega}$
 [-]','Interpreter','latex','FontSize',16)
title('Model: Output power - flow at
 h=1','Interpreter','latex','FontSize',16)
hold on

p=zeros(length(q_query),length(omega_query));        %each row const
 flow, each column const speed
for j=1:length(omega_query)

    p(:,j)=t(:,j).*omega_query(j);

    %plot one curve per speed
    txt=['$\tilde{\omega} = $',num2str(omega_query(j))];
    plot(q_query,p(:,j),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',12)
end

hold off
legend show

%-------------------------------------------------------------------------
% 11. Efficiency - flow plot at different speeds and const rated head
%-------------------------------------------------------------------------
figure(11)
axis([0 2 0 1.1])
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('$\tilde{\eta}$ [-]','Interpreter','latex','FontSize',16)
title('Model: Efficiency - flow at
 h=1','Interpreter','latex','FontSize',16)
hold on

eta=zeros(length(q_query),length(omega_query));      %each row const
 flow, each column const speed
for j=1:length(omega_query)

    eta(:,j)=(t(:,j).*omega_query(j))./(q_query'.*1);%since h=1

    %plot one curve per speed
    txt=['$\tilde{\omega} = $',num2str(omega_query(j))];
    plot(q_query,eta(:,j),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',12)
end

hold off
legend show

%-------------------------------------------------------------------------
% common for plot 12 - 14:
%-------------------------------------------------------------------------

h_query=linspace(0,1.6,161);                         %h=1 included
y_query=linspace(0.5,1.5,11);                        %y=1 included
omega=1;

%-------------------------------------------------------------------------
% 12. Flow - head plot at different opening degrees and const rated
 speed
%-------------------------------------------------------------------------
figure(12)
axis([0 2 0 2])
xlabel('h [-]','Interpreter','latex','FontSize',16)
ylabel('q [-]','Interpreter','latex','FontSize',16)
title('Model: Flow - head at $\tilde{\omega} =
 1$','Interpreter','latex','FontSize',16)
hold on

q=zeros(length(y_query),length(h_query));            %each row const
 opening, each column const head
for i=1:length(y_query)
    q(i,:)=y_query(i)*sqrt(h_query-sigma*(omega^2-1));

    %plot one curve per opening degree
    txt=['y = ',num2str(y_query(i))];
    plot(h_query,q(i,:),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',12)
end

hold off
legend show
```

```
% find dq/dh at q=h=y=omega=1
pos_h=find(h_query==1);
pos_y=find(y_query==1);
q_test=q(pos_y,pos_h)                                   %should be 1

% using a central difference scheme
dqdh=(q(pos_y,pos_h+1)-q(pos_y,pos_h-1))/(h_query(pos_h+1)-
h_query(pos_h-1));

% using inbuildt gradient function
f=gradient(q(pos_y,:));
dqdh_check=f(pos_h)/(0.5*(h_query(pos_h+1)-h_query(pos_h-1)));

%-------------------------------------------------------------------------
% 13. Torque - head plot at different opening degrees and const rated
 speed
%-------------------------------------------------------------------------
figure(13)
axis([0 2 0 2.5])
xlabel('h [-]','Interpreter','latex','FontSize',16)
ylabel('t [-]','Interpreter','latex','FontSize',16)
title('Model: Torque - head at $\tilde{\omega} =
 1$','Interpreter','latex','FontSize',16)
hold on

t=zeros(length(y_query),length(h_query));           %each row const
 opening, each column const head
for i=1:length(y_query)

    qdivy=sqrt(h_query-sigma*(omega^2-1));
    alfa_1=asin(y_query(i)*sin(alfa_1R*pi/180));    %radians
    m_S=xi*qdivy.*(cos(alfa_1)+tan(alfa_1R*pi/180)*sin(alfa_1));

    t(i,:)=q(i,:).*(m_S - psi*omega);

    %plot one curve per opening degree
    txt=['y = ',num2str(y_query(i))];
    plot(h_query,t(i,:),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',12)
end

hold off
legend show

% find dt/dh at q=h=y=omega=1
pos_h=find(h_query==1);
pos_y=find(y_query==1);
t_test=t(pos_y,pos_h)                                   %should be 1

% using a central difference scheme
dtdh=(t(pos_y,pos_h+1)-t(pos_y,pos_h-1))/(h_query(pos_h+1)-
h_query(pos_h-1));

% using inbuildt gradient function
f=gradient(t(pos_y,:));
dtdh_check=f(pos_h)/(0.5*(h_query(pos_h+1)-h_query(pos_h-1)));

%-------------------------------------------------------------------------
% 14. Efficiency - head plot at different opening degrees and const
 rated speed
%-------------------------------------------------------------------------
figure(14)
axis([0 2 0 1.1])
xlabel('h [-]','Interpreter','latex','FontSize',16)
ylabel('$\tilde{\eta}$ [-]','Interpreter','latex','FontSize',16)
title('Model: Efficiency - head at $\tilde{\omega} =
 1$','Interpreter','latex','FontSize',16)
hold on

eta=zeros(length(y_query),length(h_query));         %each row const
 opening, each column const head
for i=1:length(y_query)

    eta(i,:)=(t(i,:).*1)./(q(i,:).*h_query);        %since omega=1

    %plot one curve per opening degree
    txt=['y = ',num2str(y_query(i))];
    plot(h_query,eta(i,:),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',12)
end

hold off
legend show

%-------------------------------------------------------------------------
% Finding the characteristics at BEP
%-------------------------------------------------------------------------
a_11=dqdh;
a_12=dqdy;
a_13=dqdomega;
a_21=dtdh/a_11;
a_22=dtdy-a_21*a_12;
a_23=dtdomega-a_21*a_13;

check=a_21+a_22                                 %should be =1
```

characteristics_model_with_eta_i.m:

```
% Non-linear turbine model with eta_i implementation specified in
% eta_i functions at the bottom of script - general version

% clear all
close all
clc

% Design parameters from excel sheet - choose turbine by specifying
% column, run its 'eta_i_xxx' script before this script
 design=table2array(readtable('Hillchart_r3_medBEP.xlsx','Sheet',...
                              'Turbin_design','Range','U9:U80'));
% necessary model input:
 alfa_1R=design(49,1); sigma=design(53,1); psi=design(54,1);
 xi=design(56,1);

%-------------------------------------------------------------------------
% common for plot 1 - 4:
%-------------------------------------------------------------------------
y_query=linspace(0.2,1.4,13);                   %y=1 included
omega_query=linspace(0,1.56,157);               %omega=1 included
h=1;

%-------------------------------------------------------------------------
% 1. Flow - speed plot at different opening degrees and const rated
 head
%-------------------------------------------------------------------------
figure(1)
axis([0 2 0 2])
xlabel('$\tilde{\omega}$ [-]','Interpreter','latex','FontSize',16)
ylabel('q [-]','Interpreter','latex','FontSize',16)
title('Model with $\eta_i(q)$: Flow - speed at
 h=1','Interpreter','latex','FontSize',16)
hold on

q=zeros(length(y_query),length(omega_query));       %each row const
 opening, each column const speed
for i=1:length(y_query)
    q(i,:)=y_query(i)*sqrt(h-sigma*(omega_query.^2-1));

    %plot one curve per opening degree
    txt=['y = ',num2str(y_query(i))];
    plot(omega_query,q(i,:),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show

% find dq/domega at q=omega=y=h=1
pos_omega=find(omega_query==1);
pos_y=find(y_query==1);
q_test=q(pos_y,pos_omega)                       %should be 1

% using a central difference scheme
dqdomega=(q(pos_y,pos_omega+1)-q(pos_y,pos_omega-1))/
(omega_query(pos_omega+1)-omega_query(pos_omega-1));

% using inbuildt gradient function
f=gradient(q(pos_y,:));
dqdomega_check=f(pos_omega)/(0.5*(omega_query(pos_omega+1)-
omega_query(pos_omega-1)));

%-------------------------------------------------------------------------
% 2. Torque - speed plot at different opening degrees and const rated
 head
%-------------------------------------------------------------------------
figure(2)
axis([0 2 0 2])
xlabel('$\tilde{\omega}$ [-]','Interpreter','latex','FontSize',16)
ylabel('t [-]','Interpreter','latex','FontSize',16)
title('Model with $\eta_i(q)$: Torque - speed at
 h=1','Interpreter','latex','FontSize',16)
hold on

t=zeros(length(y_query),length(omega_query));       %each row const
 opening, each column const speed
for i=1:length(y_query)
    alfa_1=asin(y_query(i)*sin(alfa_1R*pi/180));    %radians
    m_S=xi*q(i,:)./
y_query(i)*(cos(alfa_1)+tan(alfa_1R*pi/180)*sin(alfa_1));
    t(i,:)=eta_i(q(i,:)).*q(i,:).*(m_S - psi.*omega_query);

    %plot one curve per opening degree
    txt=['y = ',num2str(y_query(i))];
    plot(omega_query,t(i,:),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show

% find dt/domega at t=omega=y=h=1
pos_omega=find(omega_query==1);
pos_y=find(y_query==1);
t_test=t(pos_y,pos_omega)                       %should be 1

% using a central difference scheme
dtdomega=(t(pos_y,pos_omega+1)-t(pos_y,pos_omega-1))/
(omega_query(pos_omega+1)-omega_query(pos_omega-1));

% using inbuildt gradient function
f=gradient(t(pos_y,:));
dtdomega_check=f(pos_omega)/(0.5*(omega_query(pos_omega+1)-
omega_query(pos_omega-1)));

%-------------------------------------------------------------------------
```

```
% 3. Mech. power - speed plot at different opening degrees and const
 rated head
%-------------------------------------------------------------------------
figure(3)
axis([0 2 0 1.8])
xlabel('$\tilde{\omega}$ [-]','Interpreter','latex','FontSize',16)
ylabel('p = t*$\tilde{\omega}$
 [-]','Interpreter','latex','FontSize',16)
title('Model with $\eta_i(q)$: Output power - speed at
 h=1','Interpreter','latex','FontSize',16)
hold on

p=zeros(length(y_query),length(omega_query));       %each row const
 opening, each column const speed
for i=1:length(y_query)
    p(i,:)=t(i,:).*omega_query;

    %plot one curve per opening degree
    txt=['y = ',num2str(y_query(i))];
    plot(omega_query,p(i,:),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show

%-------------------------------------------------------------------------
% 4. Efficiency - speed plot at different opening degrees and const
 rated head
%-------------------------------------------------------------------------
figure(4)
axis([0 2 0 1.1])
xlabel('$\tilde{\omega}$ [-]','Interpreter','latex','FontSize',16)
ylabel('$\tilde{\eta}$ [-]','Interpreter','latex','FontSize',16)
title('Model with $\eta_i(q)$: Efficiency - speed at
 h=1','Interpreter','latex','FontSize',16)
hold on

eta=zeros(length(y_query),length(omega_query));     %each row const
 opening, each column const speed
for i=1:length(y_query)

    eta(i,:)=(t(i,:).*omega_query)./(q(i,:).*1);    %since h=1

    %plot one curve per opening degree
    txt=['y = ',num2str(y_query(i))];
    plot(omega_query,eta(i,:),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show
```

```
%-------------------------------------------------------------------------
% common for plot 5 - 8:
%-------------------------------------------------------------------------
y_query=linspace(0.2,1.4,61);                   %y=1 included
omega_query=linspace(0.5,1.5,11);               %omega=1 included
h=1;

%-------------------------------------------------------------------------
% 5. Flow - opening degree plot at different speeds and const rated
 head
%-------------------------------------------------------------------------
figure(5)
axis([0 2 0 2])
xlabel('y [-]','Interpreter','latex','FontSize',16)
ylabel('q [-]','Interpreter','latex','FontSize',16)
title('Model with $\eta_i(q)$: Flow - opening degree at
 h=1','Interpreter','latex','FontSize',16)
hold on

q=zeros(length(y_query),length(omega_query));       %each row const
 opening, each column const speed
for j=1:length(omega_query)
    q(:,j)=y_query.*sqrt(h-sigma*(omega_query(j)^2-1));

    %plot one curve per speed
    txt=['$\tilde{\omega} = $',num2str(omega_query(j))];
    plot(y_query,q(:,j),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show

% find dq/dy at q=y=omega=h=1
pos_y=find(y_query==1);
pos_omega=find(omega_query==1);
q_test=q(pos_y,pos_omega)                       %should be 1

% using a central difference scheme
dqdy=(q(pos_y+1,pos_omega)-q(pos_y-1,pos_omega))/(y_query(pos_y+1)-
y_query(pos_y-1));

% using inbuildt gradient function
f=gradient(q(:,pos_omega));
dqdy_check=f(pos_y)/(0.5*(y_query(pos_y+1)-y_query(pos_y-1)));

%-------------------------------------------------------------------------
% 6. Torque - opening degree plot at different speeds and const rated
 head
%-------------------------------------------------------------------------
figure(6)
axis([0 2 0 2])
xlabel('y [-]','Interpreter','latex','FontSize',16)
ylabel('t [-]','Interpreter','latex','FontSize',16)
```

```
title('Model with $\eta_i(q)$: Torque - opening degree at
 h=1','Interpreter','latex','FontSize',16)
hold on

t=zeros(length(y_query),length(omega_query));       %each row const
 opening, each column const speed
alfa_1=asin(y_query.*sin(alfa_1R*pi/180));          %radians

for j=1:length(omega_query)
    m_S=xi*((q(:,j)').*
y_query).*(cos(alfa_1)+tan(alfa_1R*pi/180)*sin(alfa_1));
    t(:,j)=eta_i(q(:,j)').*(q(:,j)').*(m_S - psi.*omega_query(j));

    %plot one curve per speed
    txt=['$\tilde{\omega} = $',num2str(omega_query(j))];
    plot(y_query,t(:,j),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show

% find dt/dy at t=y=omega=h=1
pos_omega=find(omega_query==1);
pos_y=find(y_query==1);
t_test=t(pos_y,pos_omega)                           %should be 1

% using a central difference scheme
dtdy=(t(pos_y+1,pos_omega)-t(pos_y-1,pos_omega))/(y_query(pos_y+1)-
y_query(pos_y-1));

% using inbuildt gradient function
f=gradient(t(:,pos_omega));
dtdy_check=f(pos_y)/(0.5*(y_query(pos_y+1)-y_query(pos_y-1)));

%-------------------------------------------------------------------------
% 7. Mech. power - opening degree plot at different speeds and const
 rated head
%-------------------------------------------------------------------------
figure(7)
axis([0 2 0 2])
xlabel('y [-]','Interpreter','latex','FontSize',16)
ylabel('p = t*$\tilde{\omega}$
 [-]','Interpreter','latex','FontSize',16)
title('Model with $\eta_i(q)$: Output power - opening degree at
 h=1','Interpreter','latex','FontSize',16)
hold on

p=zeros(length(y_query),length(omega_query));       %each row const
 opening, each column const speed

for j=1:length(omega_query)
    p(:,j)=t(:,j).*omega_query(j);

    %plot one curve per speed
    txt=['$\tilde{\omega} = $',num2str(omega_query(j))];
    plot(y_query,p(:,j),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show

%-------------------------------------------------------------------------
% 8. Efficiency - opening degree plot at different speeds and const
 rated head
%-------------------------------------------------------------------------
figure(8)
axis([0 2 0 1.1])
xlabel('y [-]','Interpreter','latex','FontSize',16)
ylabel('$\tilde{\eta}$ [-]','Interpreter','latex','FontSize',16)
title('Model with $\eta_i(q)$: Efficiency - opening degree at
 h=1','Interpreter','latex','FontSize',16)
hold on

eta=zeros(length(y_query),length(omega_query));     %each row const
 opening, each column const speed
for j=1:length(omega_query)

    eta(:,j)=(t(:,j).*omega_query(j))./(q(:,j).*1); %since h=1

    %plot one curve per speed
    txt=['$\tilde{\omega} = $',num2str(omega_query(j))];
    plot(y_query,eta(:,j),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show

%-------------------------------------------------------------------------
% common for plot 9 - 11:
%-------------------------------------------------------------------------
q_query=linspace(0.06,1.56,76);                     %q=1 included
omega_query=linspace(0.5,1.5,11);                   %omega=1 included
h=1;

%-------------------------------------------------------------------------
% 9. Torque - flow plot at different speeds and const rated head
%-------------------------------------------------------------------------
figure(9)
axis([0 2 0 2])
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('t [-]','Interpreter','latex','FontSize',16)
title('Model with $\eta_i(q)$: Torque - flow at
 h=1','Interpreter','latex','FontSize',16)
hold on

t=zeros(length(q_query),length(omega_query));       %each row const
 flow, each column const speed
for j=1:length(omega_query)

    qdivy=sqrt(h-sigma*(omega_query(j)^2-1));
    y=q_query./qdivy;
    alfa_1=asin(y.*sin(alfa_1R*pi/180));            %radians
    m_S=xi*qdivy*(cos(alfa_1)+tan(alfa_1R*pi/180)*sin(alfa_1));

    t(:,j)=eta_i(q_query).*q_query.*(m_S - psi.*omega_query(j));

    %plot one curve per speed
    txt=['$\tilde{\omega} = $',num2str(omega_query(j))];
    plot(q_query,t(:,j),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show

%-------------------------------------------------------------------------
% 10. Mech. power - flow plot at different speeds and const rated head
%-------------------------------------------------------------------------
figure(10)
axis([0 2 0 2])
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('p = t*$\tilde{\omega}$
 [-]','Interpreter','latex','FontSize',16)
title('Model with $\eta_i(q)$: Output power - flow at
 h=1','Interpreter','latex','FontSize',16)
hold on

p=zeros(length(q_query),length(omega_query));       %each row const
 flow, each column const speed
for j=1:length(omega_query)

    p(:,j)=t(:,j).*omega_query(j);

    %plot one curve per speed
    txt=['$\tilde{\omega} = $',num2str(omega_query(j))];
    plot(q_query,p(:,j),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show

%-------------------------------------------------------------------------
% 11. Efficiency - flow plot at different speeds and const rated head
%-------------------------------------------------------------------------
figure(11)
axis([0 2 0 1.1])
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('$\tilde{\eta}$ [-]','Interpreter','latex','FontSize',16)
title('Model with $\eta_i(q)$: Efficiency - flow at
 h=1','Interpreter','latex','FontSize',16)
hold on

eta=zeros(length(q_query),length(omega_query));     %each row const
 flow, each column const speed
for j=1:length(omega_query)

    eta(:,j)=(t(:,j).*omega_query(j))./(q_query'.*1);%since h=1

    %plot one curve per speed
    txt=['$\tilde{\omega} = $',num2str(omega_query(j))];
    plot(q_query,eta(:,j),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show

%-------------------------------------------------------------------------
% common for plot 12 - 14:
%-------------------------------------------------------------------------
y_query=linspace(0.4,1.3,10);                       %y=1 included
h_query=linspace(0,1.6,81);                         %h=1 included
omega=1;

%-------------------------------------------------------------------------
% 12. Flow - head plot at different opening degrees and const rated
 speed
%-------------------------------------------------------------------------
figure(12)
axis([0 2 0 2])
xlabel('h [-]','Interpreter','latex','FontSize',16)
ylabel('q [-]','Interpreter','latex','FontSize',16)
title('Model with $\eta_i(q)$: Flow - head at $\tilde{\omega} =
 1$','Interpreter','latex','FontSize',16)
hold on

q=zeros(length(y_query),length(h_query));           %each row const
 opening, each column const head
for i=1:length(y_query)
    q(i,:)=y_query(i)*sqrt(h_query-sigma*(omega^2-1));

    %plot one curve per opening degree
    txt=['y = ',num2str(y_query(i))];
    plot(h_query,q(i,:),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show
```

```
% find dq/dh at q=h=y=omega=1
pos_h=find(h_query==1);
pos_y=find(y_query==1);
q_test=q(pos_y,pos_h)                                   %should be 1

% using a central difference scheme
dqdh=(q(pos_y,pos_h+1)-q(pos_y,pos_h-1))/(h_query(pos_h+1)-
h_query(pos_h-1));

% using inbuildt gradient function
f=gradient(q(pos_y,:));
dqdh_check=f(pos_h)/(0.5*(h_query(pos_h+1)-h_query(pos_h-1)));

%-------------------------------------------------------------------------
% 13. Torque - head plot at different opening degrees and const rated
 speed
%-------------------------------------------------------------------------
figure(13)
axis([0 2 0 2.5])
xlabel('h [-]','Interpreter','latex','FontSize',16)
ylabel('t [-]','Interpreter','latex','FontSize',16)
title('Model with $\eta_i(q)$: Torque - head at $\tilde{\omega} =
 1$','Interpreter','latex','FontSize',16)
hold on

t=zeros(length(y_query),length(h_query));           %each row const
 opening, each column const head
for i=1:length(y_query)

    qdivy=sqrt(h_query-sigma*(omega^2-1));
    alfa_1=asin(y_query(i)*sin(alfa_1R*pi/180));    %radians
    m_S=xi*qdivy.*(cos(alfa_1)+tan(alfa_1R*pi/180)*sin(alfa_1));

    t(i,:)=eta_i(q(i,:)).*q(i,:).*(m_S - psi*omega);

    %plot one curve per opening degree
    txt=['y = ',num2str(y_query(i))];
    plot(h_query,t(i,:),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show

% find dt/dh at q=h=y=omega=1
pos_h=find(h_query==1);
pos_y=find(y_query==1);
t_test=t(pos_y,pos_h)                                   %should be 1

% using a central difference scheme
dtdh=(t(pos_y,pos_h+1)-t(pos_y,pos_h-1))/(h_query(pos_h+1)-
h_query(pos_h-1));

% using inbuildt gradient function
f=gradient(t(pos_y,:));
dtdh_check=f(pos_h)/(0.5*(h_query(pos_h+1)-h_query(pos_h-1)));

%-------------------------------------------------------------------------
% 14. Efficiency - head plot at different opening degrees and const
 rated speed
%-------------------------------------------------------------------------
figure(14)
axis([0 2 0 1.1])
xlabel('h [-]','Interpreter','latex','FontSize',16)
ylabel('$\tilde{\eta}$ [-]','Interpreter','latex','FontSize',16)
title('Model with $\eta_i(q)$: Efficiency - head at $\tilde{\omega} =
 1$','Interpreter','latex','FontSize',16)
hold on

eta=zeros(length(y_query),length(h_query));         %each row const
 opening, each column const head
for i=1:length(y_query)

    eta(i,:)=(t(i,:).*1)./(q(i,:).*h_query);        %since omega=1

    %plot one curve per opening degree
    txt=['y = ',num2str(y_query(i))];
    plot(h_query,eta(i,:),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10)
end

hold off
legend show


%-------------------------------------------------------------------------
% Finding the characteristics at BEP
%-------------------------------------------------------------------------
a_11=dqdh;
a_12=dqdy;
a_13=dqdomega;
a_21=dtdh/a_11;
a_22=dtdy-a_21*a_12;
a_23=dtdomega-a_21*a_13;

check=a_21+a_22                                 %should be =1


%-------------------------------------------------------------------------
% uncomment the relevant eta_i(q) function
%eta_i multiplied into torque equation, should affect t-, p- and eta-
plots
%q = single flow value/ vector at constant speed/ opening degree
%-------------------------------------------------------------------------

% 1. POLYNOMIAL FUNCTION
function y=eta_i(q)
    global poly_p1 poly_p2 poly_p3 poly_p4 poly_p5 poly_p6 poly_p7 ...
           poly_p8 poly_p9 poly_p10
    y = poly_p1*q.^9 + poly_p2*q.^8 + poly_p3*q.^7 + poly_p4*q.^6 ...
        + poly_p5*q.^5 + poly_p6*q.^4 + poly_p7*q.^3 ...
        + poly_p8*q.^2 + poly_p9*q + poly_p10;
end

% 2. FOURIER SERIES FUNCTION
function y=eta_i(q)
    global fou_a0 fou_a1 fou_b1 fou_a2 fou_b2 fou_a3 fou_b3 fou_a4 ...
           fou_b4 fou_a5 fou_b5 fou_a6 fou_b6 fou_w
    y = fou_a0 + fou_a1*cos(1*fou_w.*q) + fou_b1*sin(1*fou_w.*q) ...
        + fou_a2*cos(2*fou_w.*q) + fou_b2*sin(2*fou_w.*q) ...
        + fou_a3*cos(3*fou_w.*q) + fou_b3*sin(3*fou_w.*q) ...
        + fou_a4*cos(4*fou_w.*q) + fou_b4*sin(4*fou_w.*q)...
        + fou_a5*cos(5*fou_w.*q) + fou_b5*sin(5*fou_w.*q) ...
        + fou_a6*cos(6*fou_w.*q) + fou_b6*sin(6*fou_w.*q);
end

% 3. TWO-TERM EXPONENTIAL FUNCTIONS
function y=eta_i(q)
    global expo1_a expo1_b expo1_c expo1_d expo2_a expo2_b ...
           expo2_c expo2_d
    y=zeros(size(q));

    for k=1:length(q)
        if q(k)<=1
            y(k)=expo1_a*exp(expo1_b*q(k))+expo1_c*exp(expo1_d*q(k));
        elseif q(k)>1
            y(k)=expo2_a*exp(expo2_b*q(k))+expo2_c*exp(expo2_d*q(k));
        end
    end
end

% 4. TWO-TERM POWER FUNCTIONS
function y=eta_i(q)
    global pow1_a pow1_b pow1_c pow2_a pow2_b pow2_c
    y=zeros(size(q));

    for k=1:length(q)
        if q(k)<=1
            y(k) = pow1_a*q(k)^pow1_b + pow1_c;
        elseif q(k)>1
            y(k) = pow2_a*q(k)^pow2_b + pow2_c;
        end
    end
end

% 5. PARABOLA PROPOSED BY NIELSEN
function y=eta_i(q)
    y=q.*(2-q);
end
```

Model improvement

- Script name: $eta\_i\_highH.m$, $eta\_i\_medH.m$, $eta\_i\_lowH.m$
- Purpose: Scripts for curve fitting $\eta_i(q)$ functions to measurements. Reads both experimental data and turbine design parameters from Excel sheet, and curve fits a polynomial, a Fourier series, two exponential functions and two power functions, to these data (smooth version). One must specify degree for the polynomial and number of harmonics for the Fourier. All $\eta_i(q)$ curves are plotted together with the basis for curve fitting, as well as together with each other, for comparison. The script calculates also first derivative in (1,1) for all functions to check if it is close to 0. Finally, all curve coefficients are set as global variables for η_i implementation into the model by scripts of type $characteristics\_model\_with\_eta\_i.m$.
- Script name: $curve\_relationships.m$
- Purpose: Investigates correlations between coefficients of the four different curve fitting models, and the turbine speed numbers Ω. All the same curve fittings as in $eta\_i\_xxx.m$ are redone to obtain all the coefficients. It also compares curve fitting *basis* of the three turbines, that is, general $\eta_i(q)$ shapes, in relation to speed number. The approach to linearly interpolate between curve functions to obtain η_i based on Ω, is demonstrated at the end of this script .

$eta\_i\_highH.m$:

```
clear all
close all
clc

%load data from Rainpower file and sort it
data=table2array(readtable('Hillchart_r3_medBEP.xlsx','Sheet','Høy',...
                 'Range','B2:F257'));    %[etaM Q11 N11 T11 alfa]
etaM=data(:,1); Q11=data(:,2); N11=data(:,3); T11=data(:,4);
 alfa=data(:,5);

%remove data for negative efficiencies (irrelevant)
positive=find(etaM>=0);
etaM=etaM(positive); Q11=Q11(positive); N11=N11(positive);
 T11=T11(positive); alfa=alfa(positive);

%find rated values according to best efficiency point (BEP)
etaM_R=max(etaM);
pos1=find(etaM==etaM_R);
Q11_R=Q11(pos1); N11_R=N11(pos1); T11_R=T11(pos1); alfa_R=alfa(pos1);

%non-dimensionalize by dividing by rated values
eta=etaM./etaM_R; q=Q11./Q11_R; n=N11./N11_R; t=T11./T11_R;
 y=sind(alfa)./sind(alfa_R);

%model inputs for the geometrical similar turbine
model_inputs=table2array(readtable('Hillchart_r3_medBEP.xlsx',...
                         'Sheet','Turbin_design','Range','U57:U65'));
alfa_1R=model_inputs(1,1); sigma=model_inputs(5,1);
 psi=model_inputs(6,1); xi=model_inputs(8,1);

%-------------------------------------------------------------------------
% Efficiency - flow plot at different speeds and const rated head
%-------------------------------------------------------------------------

n_query=linspace(0.8,1.3,11);              %n=1.0 included
dq=0.01;
qhalf1=linspace(min(q),1,ceil((1-min(q))/dq));
qhalf2=linspace(1,max(q),ceil((max(q)-1)/dq));
q_query=[qhalf1 qhalf2(2:end)];            %q=1.0 included

[N,Q]=meshgrid(n_query,q_query);           %each column in N has
 const speed
                                           %each row in Q has const
 flow
F = scatteredInterpolant(n,q,eta,'natural');%create interpolant that
 fits a surface on the form eta=F(n,q)
ETA=F(N,Q);                                %evaluate F at query
 points specified by N,Q matrices

figure(1)
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('$\tilde{\eta}$ [-]','Interpreter','latex','FontSize',16)
title('High head turbine','Interpreter','latex','FontSize',16)
hold on

%plot one curve per speed
for j=2:length(n_query)
    txt={'$\tilde{\omega} = $',num2str(n_query(j))};
    plot(Q(:,j),ETA(:,j),'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10,'Location','south')
end

axis([0 2 0 1.05])
hold off
legend show

%-------------------------------------------------------------------------
% TESTING CURVE FITTINGS
%-------------------------------------------------------------------------

n1=1.0;                                    %change to investigate other speeds
pos2=find(n_query==n1);
q1=Q(:,pos2); eta_meas=ETA(:,pos2);

%calc model efficiency without eta_i(q) at h=1, omega=1, varying q
eta_model=xi*(cos(asin(q1.*sind(alfa_1R))) +
 q1.*tand(alfa_1R)*sind(alfa_1R)) - psi;

eta_i=eta_meas./eta_model;                 %bc eta_model*eta_i(q)~=eta_meas

%klipper vekk for q>1.57 fordi der begynner eta gå oppover igjen
for j=1:length(q1)
   if q1(j)>1.57
      break
   end
end
q1=q1(1:j-1); eta_meas=eta_meas(1:j-1); eta_model=eta_model(1:j-1);
 eta_i=eta_i(1:j-1);

eta_i_before=eta_i;                        %saving original data
eta_i=smooth(eta_i);                       %smooth data before fitting

q2=linspace(0,2,201);                      %for plotting
q1=[q1;2]; eta_i=[eta_i;0];                %adding (2,0) for better curve
 shape in q > q_limit


%-------------------------------------------------------------------------
% 1. POLYNOMIAL FUNCTION
%-------------------------------------------------------------------------

%extra weight on point (q,eta)=(1,1)
w=ones(size(eta_i));
pos3=find(q_query==1);
w(pos3)=10;                                %specify >1
```

```
poly=fit(q1,eta_i,'poly9','Weight',w);

% pp = (poly.p1)*q1.^9 + (poly.p2)*q1.^8 + (poly.p3)*q1.^7 +
 (poly.p4)*q1.^6 + (poly.p5)*q1.^5 + (poly.p6)*q1.^4 + ...
%      (poly.p7)*q1.^3 + (poly.p8)*q1.^2 + (poly.p9)*q1 + (poly.p10);
pp = (poly.p1)*q2.^9 + (poly.p2)*q2.^8 + (poly.p3)*q2.^7 +
 (poly.p4)*q2.^6 + (poly.p5)*q2.^5 + (poly.p6)*q2.^4 + ...
    (poly.p7)*q2.^3 + (poly.p8)*q2.^2 + (poly.p9)*q2 + (poly.p10);

figure(2)
plot(q1,eta_i,'r*',q2,eta_i_parabola(q2),'k-.',q2,pp,'k-','LineWidth',1)
legend('Data for curve fitting','$q (2-q)$','Polynomial of degree 9')
set(legend,'Interpreter','latex','FontSize',10,'Location','south')
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('$\tilde{\eta}$ [-]','Interpreter','latex','FontSize',16)
axis([0 2 0 1.05])

%test resulting function
syms x
p = poly.p1*x^9 + poly.p2*x^8 + poly.p3*x^7 + poly.p4*x^6 +
 poly.p5*x^5 + poly.p6*x^4 + poly.p7*x^3 + poly.p8*x^2 + poly.p9*x +
 poly.p10;
dpdx = diff(p);
poly_derivative=vpa(subs(dpdx,x,1))     %should be 0 or close to 0


%-------------------------------------------------------------------------
% 2. FOURIER SERIES FUNCTION
%-------------------------------------------------------------------------

fou=fit(q1,eta_i,'fourier6','Weight',w);

% ff = fou.a0 + fou.a1*cos(1*fou.w*q1) + fou.b1*sin(1*fou.w*q1) +
 fou.a2*cos(2*fou.w*q1) + fou.b2*sin(2*fou.w*q1)...
%      + fou.a3*cos(3*fou.w*q1) + fou.b3*sin(3*fou.w*q1) +
 fou.a4*cos(4*fou.w*q1) + fou.b4*sin(4*fou.w*q1)...
%      + fou.a5*cos(5*fou.w*q1) + fou.b5*sin(5*fou.w*q1) +
 fou.a6*cos(6*fou.w*q1) + fou.b6*sin(6*fou.w*q1);
ff = fou.a0 + fou.a1*cos(1*fou.w*q2) + fou.b1*sin(1*fou.w*q2) +
 fou.a2*cos(2*fou.w*q2) + fou.b2*sin(2*fou.w*q2)...
    + fou.a3*cos(3*fou.w*q2) + fou.b3*sin(3*fou.w*q2) +
 fou.a4*cos(4*fou.w*q2) + fou.b4*sin(4*fou.w*q2)...
    + fou.a5*cos(5*fou.w*q2) + fou.b5*sin(5*fou.w*q2) +
 fou.a6*cos(6*fou.w*q2) + fou.b6*sin(6*fou.w*q2);

figure(3)
plot(q1,eta_i,'r*',q2,eta_i_parabola(q2),'k-.',q2,ff,'k-','LineWidth',1)
legend('Data for curve fitting','$q (2-q)$','Fourier series of 6
 harmonics')
set(legend,'Interpreter','latex','FontSize',10,'Location','south')
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('$\tilde{\eta}$ [-]','Interpreter','latex','FontSize',16)
axis([0 2 0 1.05])

%test resulting function
syms x
f = fou.a0 + fou.a1*cos(1*fou.w*x) + fou.b1*sin(1*fou.w*x) +
 fou.a2*cos(2*fou.w*x) + fou.b2*sin(2*fou.w*x)...
    + fou.a3*cos(3*fou.w*x) + fou.b3*sin(3*fou.w*x) +
 fou.a4*cos(4*fou.w*x) + fou.b4*sin(4*fou.w*x)...
    + fou.a5*cos(5*fou.w*x) + fou.b5*sin(5*fou.w*x) +
 fou.a6*cos(6*fou.w*x) + fou.b6*sin(6*fou.w*x);
dfdx = diff(f);
fou_derivative=vpa(subs(dfdx,x,1))      %should be 0 or close to 0


%-------------------------------------------------------------------------
% 3. TWO-TERM EXPONENTIAL FUNCTIONS
%    e1(q) for q<=1, e2(q) for q>=1
%-------------------------------------------------------------------------

q1_1=q1(find(q1<=1));                                   %q1<=1
q1_2=q1(find(q1>=1));                                   %q1>=1
eta_i_1=eta_i(find(q1<=1));
eta_i_2=eta_i(find(q1>=1));

%extra weight on point (q,eta)=(1,1)
w1=ones(size(eta_i_1)); w1(end)=1000;
w2=ones(size(eta_i_2)); w2(1)=1000;

%two exponential functions on eta-increasing and eta-decreasing
 intervals
expo1=fit(q1_1,eta_i_1,'exp2','Weight', w1);
expo2=fit(q1_2,eta_i_2,'exp2','Weight', w2);

% e1 = (expo1.a)*exp((expo1.b)*q1_1) + (expo1.c)*exp((expo1.d)*q1_1);
% e2 = (expo2.a)*exp((expo2.b)*q1_2) + (expo2.c)*exp((expo2.d)*q1_2);
q2_1=q2(find(q2<=1));
q2_2=q2(find(q2>=1));
e1 = (expo1.a)*exp((expo1.b)*q2_1) + (expo1.c)*exp((expo1.d)*q2_1);
e2 = (expo2.a)*exp((expo2.b)*q2_2) + (expo2.c)*exp((expo2.d)*q2_2);

figure(4)
plot(q1,eta_i,'r*',q2,eta_i_parabola(q2),'k-.',...
     q2_1,e1,'k-',q2_2,e2,'k-','LineWidth',1)
legend('Data for curve fitting','$q (2-q)$','Exponential for
 $q<=1$','Exponential for $q>=1$')
set(legend,'Interpreter','latex','FontSize',10,'Location','south')
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('$\tilde{\eta}$ [-]','Interpreter','latex','FontSize',16)
axis([0 2 0 1.05])

%test resulting function
syms x
ee1 = expo1.a*exp(expo1.b*x) + expo1.c*exp(expo1.d*x);
dee1dx = diff(ee1);
e1_derivative=vpa(subs(dee1dx,x,1))     %should be 0 or close to 0


%test resulting function
syms x
ee2 = expo2.a*exp(expo2.b*x) + expo2.c*exp(expo2.d*x);
dee2dx = diff(ee2);
e2_derivative=vpa(subs(dee2dx,x,1))     %should be 0 or close to 0


%-------------------------------------------------------------------------
% 4. TWO-TERM POWER FUNCTIONS
%    pow1(q) for q<=1, pow2(q) for q>=1
%-------------------------------------------------------------------------

%two power functions on eta-increasing and eta-decreasing intervals
pow1=fit(q1_1,eta_i_1,'power2','Weight', w1);
pow2=fit(q1_2,eta_i_2,'power2','Weight', w2);

% p1 = (pow1.a)*q1_1.^(pow1.b) + (pow1.c);
% p2 = (pow2.a)*q1_2.^(pow2.b) + (pow2.c);
p1 = (pow1.a)*q2_1.^(pow1.b) + (pow1.c);
p2 = (pow2.a)*q2_2.^(pow2.b) + (pow2.c);

figure(5)
plot(q1,eta_i,'r*',q2,eta_i_parabola(q2),'k-.',...
     q2_1,p1,'k-',q2_2,p2,'k-','LineWidth',1)
legend('Data for curve fitting','$q (2-q)$','Power for $q<=1$','Power
 for $q>=1$')
set(legend,'Interpreter','latex','FontSize',10,'Location','south')
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('$\tilde{\eta}$ [-]','Interpreter','latex','FontSize',16)
axis([0 2 0 1.05])

%test resulting function
syms x
pp1 = pow1.a*x^pow1.b + pow1.c;
dpp1dx = diff(pp1);
pow1_derivative=vpa(subs(dpp1dx,x,1))   %should be 0 or close to 0

%test resulting function
syms x
pp2 = pow2.a*x.^pow2.b + pow2.c;
dpp2dx = diff(pp2);
pow2_derivative=vpa(subs(dpp2dx,x,1))   %should be 0 or close to 0


%-------------------------------------------------------------------------
% Comparing eta_i(q) kurves
%-------------------------------------------------------------------------
figure(6)
plot(q2,pp,q2,ff,q2_1,e1,q2_2,e2,q2_1,p1,q2_2,p2,'LineWidth',1)
legend('Polynomial of degree 9','Fourier series of 6
 harmonics','Exponential for $q<=1$','Exponential for $q>=1$','Power
 for $q<=1$','Power for $q>=1$')
set(legend,'Interpreter','latex','FontSize',10,'Location','south')
title(['High head turbine, $\tilde{\omega}$ =
 ',num2str(n1)],'Interpreter','latex','FontSize',16)
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('$\tilde{\eta}$ [-]','Interpreter','latex','FontSize',16)
axis([0 2 0 1.05])

figure(7)
plot(q2,pp,q2,ff,'LineWidth',1)
legend('Polynomial of degree 9','Fourier series of 6 harmonics')
set(legend,'Interpreter','latex','FontSize',10,'Location','south')
title(['High head turbine, $\tilde{\omega}$ =
 ',num2str(n1)],'Interpreter','latex','FontSize',16)
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('$\tilde{\eta}$ [-]','Interpreter','latex','FontSize',16)
axis([0 2 0 1.05])

figure(8)
plot(q2_1,e1,q2_2,e2,q2_1,p1,q2_2,p2,'LineWidth',1)
legend('Exponential for $q<=1$','Exponential for $q>=1$','Power for
 $q<=1$','Power for $q>=1$')
set(legend,'Interpreter','latex','FontSize',10,'Location','south')
title(['High head turbine, $\tilde{\omega}$ =
 ',num2str(n1)],'Interpreter','latex','FontSize',16)
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('$\tilde{\eta}$ [-]','Interpreter','latex','FontSize',16)
axis([0 2 0 1.05])


%-------------------------------------------------------------------------
% Set curve coefficients global for use in eta_i(q) functions
%-------------------------------------------------------------------------
setGlobals_polynomial(poly);
setGlobals_fourier(fou);
setGlobals_exponential(expo1,expo2);
setGlobals_power(pow1,pow2);

% 1. POLYNOMIAL FUNCTION - set global variables
function setGlobals_polynomial(poly)
    global poly_p1 poly_p2 poly_p3 poly_p4 poly_p5 poly_p6 poly_p7
 poly_p8 poly_p9 poly_p10
    poly_p1 = poly.p1;
    poly_p2 = poly.p2;
    poly_p3 = poly.p3;
    poly_p4 = poly.p4;
    poly_p5 = poly.p5;
    poly_p6 = poly.p6;
    poly_p7 = poly.p7;
    poly_p8 = poly.p8;
    poly_p9 = poly.p9;
    poly_p10 = poly.p10;
end

% 2. FOURIER SERIES FUNCTION - set global variables
function setGlobals_fourier(fou)
```

```
    global fou_a0 fou_a1 fou_b1 fou_a2 fou_b2 fou_a3 fou_b3 fou_a4
 fou_b4 fou_a5 fou_b5 fou_a6 fou_b6 fou_w
    fou_a0 = fou.a0;
    fou_a1 = fou.a1;
    fou_b1 = fou.b1;
    fou_a2 = fou.a2;
    fou_b2 = fou.b2;
    fou_a3 = fou.a3;
    fou_b3 = fou.b3;
    fou_a4 = fou.a4;
    fou_b4 = fou.b4;
    fou_a5 = fou.a5;
    fou_b5 = fou.b5;
    fou_a6 = fou.a6;
    fou_b6 = fou.b6;
    fou_w = fou.w;
end

% 3. TWO-TERM EXPONENTIAL FUNCTIONS - set global variables
function setGlobals_exponential(expo1,expo2)
    global expo1_a expo1_b expo1_c expo1_d expo2_a expo2_b expo2_c
 expo2_d
    expo1_a = expo1.a;
    expo1_b = expo1.b;
    expo1_c = expo1.c;
    expo1_d = expo1.d;
    expo2_a = expo2.a;
    expo2_b = expo2.b;
    expo2_c = expo2.c;
    expo2_d = expo2.d;
end

% 4. TWO-TERM POWER FUNCTIONS - set global variables
function setGlobals_power(pow1,pow2)
    global pow1_a pow1_b pow1_c pow2_a pow2_b pow2_c
    pow1_a = pow1.a;
    pow1_b = pow1.b;
    pow1_c = pow1.c;
    pow2_a = pow2.a;
    pow2_b = pow2.b;
    pow2_c = pow2.c;
end

%parabola proposed by Nielsen
function y=eta_i_parabola(x)
    y=x.*(2-x);
end
```

curve_relationships.m:

```
% Script for investigating relationships between speed numbers and
% curve coefficients in the four different proposed eta_i(q) curves

clear all
close all
clc

%------------------------------------------------------------------------
% High head turbine
%------------------------------------------------------------------------
data=table2array(readtable('Hillchart_r3_medBEP.xlsx','Sheet','Høy',...
                           'Range','B2:F257'));
etaM=data(:,1); Q11=data(:,2); N11=data(:,3); T11=data(:,4);
 alfa=data(:,5);
positive=find(etaM>=0);
etaM=etaM(positive); Q11=Q11(positive); N11=N11(positive);
 T11=T11(positive); alfa=alfa(positive);
etaM_R=max(etaM);
pos1=find(etaM==etaM_R);
Q11_R=Q11(pos1); N11_R=N11(pos1); T11_R=T11(pos1); alfa_R=alfa(pos1);
eta=etaM./etaM_R; q=Q11./Q11_R; n=N11./N11_R; t=T11./T11_R;
 y=sind(alfa)./sind(alfa_R);
model_inputs=table2array(readtable('Hillchart_r3_medBEP.xlsx','Sheet',...

  'Turbin_design','Range','U31:U65'));
alfa_1R=model_inputs(27,1); sigma=model_inputs(31,1);
 psi=model_inputs(32,1); xi=model_inputs(34,1);
speednumber=model_inputs(1,1);

n_query=linspace(0.8,1.3,11);
dq=0.01;
qhalf1=linspace(min(q),1,ceil((1-min(q))/dq));
qhalf2=linspace(1,max(q),ceil((max(q)-1)/dq));
q_query=[qhalf1 qhalf2(2:end)];
[N,Q]=meshgrid(n_query,q_query);
F = scatteredInterpolant(n,q,eta,'natural');
ETA=F(N,Q);

n1=1.0;
pos2=find(n_query==n1);
q1=Q(:,pos2); eta_meas=ETA(:,pos2);
eta_model=xi*(cos(asin(q1.*sind(alfa_1R))) +
 q1.*tand(alfa_1R)*sind(alfa_1R)) - psi;
eta_i=eta_meas./eta_model;

for j=1:length(q1)
   if q1(j)>1.57
      break
   end
end
q1=q1(1:j-1); eta_meas=eta_meas(1:j-1); eta_model=eta_model(1:j-1);
 eta_i=eta_i(1:j-1);
eta_i_before=eta_i;
eta_i=smooth(eta_i);

w=ones(size(eta_i));
pos3=find(q_query==1);
w(pos3)=10;

poly=fit([q1;2],[eta_i;0],'poly9','Weight',[w;1]);

fou=fit([q1;2],[eta_i;0],'fourier6','Weight',[w;1]);

pos4=find(q1<=1); pos5=find(q1>=1);
q1_1=q1(pos4); q1_2=q1(pos5);
eta_i_1=eta_i(pos4); eta_i_2=eta_i(pos5);

w1=ones(size(eta_i_1)); w1(end)=1000;
w2=ones(size(eta_i_2)); w2(1)=1000;

expo1=fit(q1_1,eta_i_1,'exp2','Weight', w1);
expo2=fit(q1_2,eta_i_2,'exp2','Weight', w2);

pow1=fit(q1_1,eta_i_1,'power2','Weight', w1);
pow2=fit(q1_2,eta_i_2,'power2','Weight', w2);

speednumber_highH=speednumber;

p1_high = poly.p1;
p2_high = poly.p2;
p3_high = poly.p3;
p4_high = poly.p4;
p5_high = poly.p5;
p6_high = poly.p6;
p7_high = poly.p7;
p8_high = poly.p8;
p9_high = poly.p9;
p10_high = poly.p10;

a0_high = fou.a0;
a1_high = fou.a1;
b1_high = fou.b1;
a2_high = fou.a2;
b2_high = fou.b2;
a3_high = fou.a3;
b3_high = fou.b3;
a4_high = fou.a4;
b4_high = fou.b4;
a5_high = fou.a5;
b5_high = fou.b5;
a6_high = fou.a6;
b6_high = fou.b6;
w_high = fou.w;

exp1_a_high=expo1.a;
exp1_b_high=expo1.b;
```

```
exp1_c_high=expo1.c;
exp1_d_high=expo1.d;
exp2_a_high=expo2.a;
exp2_b_high=expo2.b;
exp2_c_high=expo2.c;
exp2_d_high=expo2.d;

pow1_a_high=pow1.a;
pow1_b_high=pow1.b;
pow1_c_high=pow1.c;
pow2_a_high=pow2.a;
pow2_b_high=pow2.b;
pow2_c_high=pow2.c;


figure(1)
plot(q1,eta_i,'r-','Linewidth',1)
hold on


%-------------------------------------------------------------------------
% Medium head turbine
%-------------------------------------------------------------------------
data=table2array(readtable('Hillchart_r3_medBEP.xlsx','Sheet','Mellom2',...
                           'Range','L2:P305'));
etaM=data(:,1); Q11=data(:,2); N11=data(:,3); T11=data(:,4);
 alfa=data(:,5);
positive=find(etaM>=0);
etaM=etaM(positive); Q11=Q11(positive); N11=N11(positive);
 T11=T11(positive); alfa=alfa(positive);
etaM_R=max(etaM);
pos1=find(etaM==etaM_R);
Q11_R=Q11(pos1); N11_R=N11(pos1); T11_R=T11(pos1); alfa_R=alfa(pos1);
eta=etaM./etaM_R; q=Q11./Q11_R; n=N11./N11_R; t=T11./T11_R;
 y=sind(alfa)./sind(alfa_R);
model_inputs=table2array(readtable('Hillchart_r3_medBEP.xlsx','Sheet',...

  'Turbin_design','Range','T31:T65'));
alfa_1R=model_inputs(27,1); sigma=model_inputs(31,1);
 psi=model_inputs(32,1); xi=model_inputs(34,1);
speednumber=model_inputs(1,1);

n_query=linspace(0.9,1.7,17);
dq=0.01;
qhalf1=linspace(min(q),1,ceil((1-min(q))/dq));
qhalf2=linspace(1,max(q),ceil((max(q)-1)/dq));
q_query=[qhalf1 qhalf2(2:end)];
[N,Q]=meshgrid(n_query,q_query);
F = scatteredInterpolant(n,q,eta,'natural');
ETA=F(N,Q);

n1=1.0;
pos2=find(n_query==n1);
q1=Q(:,pos2); eta_meas=ETA(:,pos2);
```

```
eta_model=xi*(cos(asin(q1.*sind(alfa_1R))) +
 q1.*tand(alfa_1R)*sind(alfa_1R)) - psi;
eta_i=eta_meas./eta_model;

for j=1:length(q1)
   if q1(j)>1.48
      break
   end
end
q1=q1(1:j-1); eta_meas=eta_meas(1:j-1); eta_model=eta_model(1:j-1);
 eta_i=eta_i(1:j-1);
eta_i_before=eta_i;
eta_i=smooth(eta_i,12);

w=ones(size(eta_i));
pos3=find(q_query==1);
w(pos3)=10;

poly=fit([q1;2],[eta_i;0],'poly9','Weight',[w;1]);

fou=fit(q1,eta_i,'fourier5','Weight',w);

pos4=find(q1<=1); pos5=find(q1>=1);
q1_1=q1(pos4); q1_2=q1(pos5);
eta_i_1=eta_i(pos4); eta_i_2=eta_i(pos5);

w1=ones(size(eta_i_1)); w1(end)=1000;
w2=ones(size(eta_i_2)); w2(1)=1000;

expo1=fit(q1_1,eta_i_1,'exp2','Weight', w1);
expo2=fit(q1_2,eta_i_2,'exp2','Weight', w2);

pow1=fit(q1_1,eta_i_1,'power2','Weight', w1);
pow2=fit(q1_2,eta_i_2,'power2','Weight', w2);

speednumber_medH_v2=speednumber;

p1_m2 = poly.p1;
p2_m2 = poly.p2;
p3_m2 = poly.p3;
p4_m2 = poly.p4;
p5_m2 = poly.p5;
p6_m2 = poly.p6;
p7_m2 = poly.p7;
p8_m2 = poly.p8;
p9_m2 = poly.p9;
p10_m2 = poly.p10;

a0_m2 = fou.a0;
a1_m2 = fou.a1;
b1_m2 = fou.b1;
a2_m2 = fou.a2;
b2_m2 = fou.b2;
a3_m2 = fou.a3;
```

```
b3_m2 = fou.b3;
a4_m2 = fou.a4;
b4_m2 = fou.b4;
a5_m2 = fou.a5;
b5_m2 = fou.b5;
w_m2 = fou.w;

exp1_a_m2=expo1.a;
exp1_b_m2=expo1.b;
exp1_c_m2=expo1.c;
exp1_d_m2=expo1.d;
exp2_a_m2=expo2.a;
exp2_b_m2=expo2.b;
exp2_c_m2=expo2.c;
exp2_d_m2=expo2.d;

pow1_a_m2=pow1.a;
pow1_b_m2=pow1.b;
pow1_c_m2=pow1.c;
pow2_a_m2=pow2.a;
pow2_b_m2=pow2.b;
pow2_c_m2=pow2.c;


figure(1)
plot(q1,eta_i,'b-','Linewidth',1)
hold on


%-------------------------------------------------------------------------
% Low head turbine
%-------------------------------------------------------------------------
data=table2array(readtable('Hillchart_r3_medBEP.xlsx','Sheet','Lavtrykk',...
                           'Range','B2:F119'));
etaM=data(:,1); Q11=data(:,2); N11=data(:,3); T11=data(:,4);
 alfa=data(:,5);
positive=find(etaM>=0);
etaM=etaM(positive); Q11=Q11(positive); N11=N11(positive);
 T11=T11(positive); alfa=alfa(positive);
etaM_R=max(etaM);
pos1=find(etaM==etaM_R);
Q11_R=Q11(pos1); N11_R=N11(pos1); T11_R=T11(pos1); alfa_R=alfa(pos1);
eta=etaM./etaM_R; q=Q11./Q11_R; n=N11./N11_R; t=T11./T11_R;
 y=sind(alfa)./sind(alfa_R);
model_inputs=table2array(readtable('Hillchart_r3_medBEP.xlsx','Sheet',...

  'Turbin_design','Range','R31:R65'));
alfa_1R=model_inputs(27,1); sigma=model_inputs(31,1);
 psi=model_inputs(32,1); xi=model_inputs(34,1);
speednumber=model_inputs(1,1);

n_query=linspace(0.95,1.45,11);
dq=0.01;
qhalf1=linspace(min(q),1,ceil((1-min(q))/dq));
```

```
qhalf2=linspace(1,max(q),ceil((max(q)-1)/dq));
q_query=[qhalf1 qhalf2(2:end)];
[N,Q]=meshgrid(n_query,q_query);
F = scatteredInterpolant(n,q,eta,'natural');
ETA=F(N,Q);

n1=1.0;
pos2=find(n_query==n1);
q1=Q(:,pos2); eta_meas=ETA(:,pos2);
eta_model=xi*(cos(asin(q1.*sind(alfa_1R))) +
 q1.*tand(alfa_1R)*sind(alfa_1R)) - psi;
eta_i=eta_meas./eta_model;

for j=1:length(q1)
   if q1(j)>1.20
      break
   end
end
q1=q1(1:j-1); eta_meas=eta_meas(1:j-1); eta_model=eta_model(1:j-1);
 eta_i=eta_i(1:j-1);
eta_i_before=eta_i;
eta_i=smooth(eta_i,8);

w=ones(size(eta_i));
pos3=find(q_query==1);
w(pos3)=10;

poly=fit(q1,eta_i,'poly4','Weight',w);

fou=fit(q1,eta_i,'fourier5','Weight',w);

pos4=find(q1<=1); pos5=find(q1>=1);
q1_1=q1(pos4); q1_2=q1(pos5);
eta_i_1=eta_i(pos4); eta_i_2=eta_i(pos5);

w1=ones(size(eta_i_1)); w1(end)=1000;
w2=ones(size(eta_i_2)); w2(1)=1000;

expo1=fit(q1_1,eta_i_1,'exp2','Weight', w1);
expo2=fit(q1_2,eta_i_2,'exp2','Weight', w2);

pow1=fit(q1_1,eta_i_1,'power2','Weight', w1);
pow2=fit(q1_2,eta_i_2,'power2','Weight', w2);

speednumber_lowH=speednumber;

p1_low = poly.p1;
p2_low = poly.p2;
p3_low = poly.p3;
p4_low = poly.p4;
p5_low = poly.p5;

a0_low = fou.a0;
a1_low = fou.a1;
```

```
b1_low = fou.b1;
a2_low = fou.a2;
b2_low = fou.b2;
a3_low = fou.a3;
b3_low = fou.b3;
a4_low = fou.a4;
b4_low = fou.b4;
a5_low = fou.a5;
b5_low = fou.b5;
w_low = fou.w;

exp1_a_low=expo1.a;
exp1_b_low=expo1.b;
exp1_c_low=expo1.c;
exp1_d_low=expo1.d;
exp2_a_low=expo2.a;
exp2_b_low=expo2.b;
exp2_c_low=expo2.c;
exp2_d_low=expo2.d;

pow1_a_low=pow1.a;
pow1_b_low=pow1.b;
pow1_c_low=pow1.c;
pow2_a_low=pow2.a;
pow2_b_low=pow2.b;
pow2_c_low=pow2.c;
```

```
%----------------------------------------------------------------------
% Plot curve fitting basises for shape comparison
%----------------------------------------------------------------------

%legends
txt_highH=['High head, $\Omega$ =
 ',num2str(round(speednumber_highH,2))];
txt_medH_v2=['Medium head, $\Omega$ =
 ',num2str(round(speednumber_medH_v2,2))];
txt_lowH=['Low head, $\Omega$ = ',num2str(round(speednumber_lowH,2))];

figure(1)
plot(q1,eta_i,'g-','Linewidth',1)
legend(txt_highH,txt_medH_v2,txt_lowH)
set(legend,'Interpreter','latex','FontSize',10,'Location','south')
title('Experimental data divided by model
 efficiency','Interpreter','latex','FontSize',16)
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('$\tilde{\eta}$ [-]','Interpreter','latex','FontSize',16)
axis([0 2 0 1.1])
hold off


%----------------------------------------------------------------------
% RELATION BETWEEN CURVE COEFFICIENTS AND SPEED NUMBERS
%----------------------------------------------------------------------
```

```
% x-axis
s=[speednumber_highH, speednumber_medH_v2, speednumber_lowH];

%----------------------------------------------------------------------
% 1. Polynomial coefficients relationships
%----------------------------------------------------------------------
% y-axis
p1=[p1_high, p1_m2, p1_low];
p2=[p2_high, p2_m2, p2_low];
p3=[p3_high, p3_m2, p3_low];
p4=[p4_high, p4_m2, p4_low];
p5=[p5_high, p5_m2, p5_low];
% p6=[p6_high, p6_m2, p6_low];
% p7=[p7_high, p7_m2, p7_low];
% p8=[p8_high, p8_m2, p8_low];
% p9=[p9_high, p9_m2, p9_low];

% s,p6,'g*-',s,p7,'bo-',s,p8,'ro-',s,p9,'co-',
% ,'p6','p7','p8','p9'

figure(2)
plot(s,p1,'b*-',s,p2,'r*-',s,p3,'c*-',s,p4,'m*-',s,p5,'k*-','Linewidth',1);
legend('p1','p2','p3','p4','p5')
set(legend,'Interpreter','latex','FontSize',10)
title('Polynomial coefficients','Interpreter','latex','FontSize',16)
xlabel('Speed number, $\Omega$','Interpreter','latex','FontSize',16)
ylabel('Coefficients','Interpreter','latex','FontSize',16)


%----------------------------------------------------------------------
% 2. Fourier coefficients relationships
%----------------------------------------------------------------------
% y-axis
a0=[a0_high, a0_m2, a0_low];
a1=[a1_high, a1_m2, a1_low];
b1=[b1_high, b1_m2, b1_low];
a2=[a2_high, a2_m2, a2_low];
b2=[b2_high, b2_m2, b2_low];
a3=[a3_high, a3_m2, a3_low];
b3=[b3_high, b3_m2, b3_low];
a4=[a4_high, a4_m2, a4_low];
b4=[b4_high, b4_m2, b4_low];
a5=[a5_high, a5_m2, a5_low];
b5=[b5_high, b5_m2, b5_low];
freq=[w_high, w_m2, w_low];

figure(3)
plot(s,a0,'b*-',s,a1,'r*-',s,b1,'c*-',s,a2,'m*-',s,b2,'k*-',s,a3,'g*-',s,b3,
legend('a0','a1','b1','a2','b2','a3','b3','a4','b4','a5','b5')
set(legend,'Interpreter','latex','FontSize',10)
title('Fourier series
 coefficients','Interpreter','latex','FontSize',16)
xlabel('Speed number, $\Omega$','Interpreter','latex','FontSize',16)
ylabel('Coefficients','Interpreter','latex','FontSize',16)
```

```
figure(4)
plot(s,freq,'co-','Linewidth',1)
legend('$\omega$')
set(legend,'Interpreter','latex','FontSize',10)
title('Fourier series
 frequencies','Interpreter','latex','FontSize',16)
xlabel('Speed number, $\Omega$','Interpreter','latex','FontSize',16)
ylabel('Frequency, $\omega$','Interpreter','latex','FontSize',16)


%----------------------------------------------------------------------
% 3. Exponential coefficients relationships
%----------------------------------------------------------------------
% y-axis
exp1_a=[exp1_a_high, exp1_a_m2, exp1_a_low];
exp1_b=[exp1_b_high, exp1_b_m2, exp1_b_low];
exp1_c=[exp1_c_high, exp1_c_m2, exp1_c_low];
exp1_d=[exp1_d_high, exp1_d_m2, exp1_d_low];
exp2_a=[exp2_a_high, exp2_a_m2, exp2_a_low];
exp2_b=[exp2_b_high, exp2_b_m2, exp2_b_low];
exp2_c=[exp2_c_high, exp2_c_m2, exp2_c_low];
exp2_d=[exp2_d_high, exp2_d_m2, exp2_d_low];

figure(5)
plot(s,exp1_a,'b*-',s,exp1_b,'r*-',s,exp1_c,'c*-',s,exp1_d,'m*-',s,exp2_a,
legend('a, $q<=1$','b, $q<=1$','c, $q<=1$','d, $q<=1$','a, $q>=1$','b,
 $q>=1$','c, $q>=1$','d, $q>=1$')
set(legend,'Interpreter','latex','FontSize',10)
title('Exponential function
 coefficients','Interpreter','latex','FontSize',16)
xlabel('Speed number, $\Omega$','Interpreter','latex','FontSize',16)
ylabel('Coefficients','Interpreter','latex','FontSize',16)


%----------------------------------------------------------------------
% 4. Power coefficients relationships
%----------------------------------------------------------------------
% y-axis
pow1_a=[pow1_a_high, pow1_a_m2, pow1_a_low];
pow1_b=[pow1_b_high, pow1_b_m2, pow1_b_low];
pow1_c=[pow1_c_high, pow1_c_m2, pow1_c_low];
pow2_a=[pow2_a_high, pow2_a_m2, pow2_a_low];
pow2_b=[pow2_b_high, pow2_b_m2, pow2_b_low];
pow2_c=[pow2_c_high, pow2_c_m2, pow2_c_low];

figure(6)
plot(s,pow1_a,'b*-',s,pow1_b,'r*-',s,pow1_c,'c*-',s,pow2_a,'m*-',s,pow2_b,
legend('a, $q<=1$','b, $q<=1$','c, $q<=1$','a, $q>=1$','b, $q>=1$','c,
 $q>=1$')
set(legend,'Interpreter','latex','FontSize',10)
title('Power function
 coefficients','Interpreter','latex','FontSize',16)
xlabel('Speed number, $\Omega$','Interpreter','latex','FontSize',16)
```

```
ylabel('Coefficients','Interpreter','latex','FontSize',16)


%----------------------------------------------------------------------
% WEIGHTED SHAPE FUNCTION APPROACH
%----------------------------------------------------------------------

ds=0.1;
s_query=linspace(speednumber_highH,speednumber_lowH,7);
x=(s_query-speednumber_highH)./(speednumber_lowH-speednumber_highH);

%----------------------------------------------------------------------
% 1. POLYNOMIAL FUNCTION
%----------------------------------------------------------------------

q=linspace(0,2,201);

%High head:
f_high = p1_high*q.^9 + p2_high*q.^8 + p3_high*q.^7 + p4_high*q.^6 +
 p5_high*q.^5 + p6_high*q.^4 + ...
        p7_high*q.^3 + p8_high*q.^2 + p9_high*q + p10_high;
      %poly9
%Low head:
f_low = p1_low*q.^4 + p2_low*q.^3 + p3_low*q.^2 + p4_low*q + p5_low;
      %poly4

figure(7)
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('$\eta_i$ [-]','Interpreter','latex','FontSize',16)
title('Polynomial $\eta_i ( q )$ curves
 (n=9)','Interpreter','latex','FontSize',16)
hold on

for i=1:length(x)
    y = (1-x(i))*f_high + x(i)*f_low;
    txt=['$\Omega = $',num2str(round(s_query(i),2))];
    plot(q,y,'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10,'Location','south')
end

axis([0 2 0 1.05])
hold off
legend show


%----------------------------------------------------------------------
% 2. FOURIER SERIES FUNCTION
%----------------------------------------------------------------------

q=linspace(0,1.5,151);

%High head:
```

```
f_high = a0_high + a1_high*cos(1*w_high.*q) + b1_high*sin(1*w_high.*q)
 + a2_high*cos(2*w_high.*q) + b2_high*sin(2*w_high.*q) + ...
    a3_high*cos(3*w_high.*q) + b3_high*sin(3*w_high.*q) +
 a4_high*cos(4*w_high.*q) + b4_high*sin(4*w_high.*q) + ...
    a5_high*cos(5*w_high.*q) + b5_high*sin(5*w_high.*q) +
 a6_high*cos(6*w_high.*q) + b6_high*sin(6*w_high.*q);      %fourier6
%Low head:
f_low = a0_low + a1_low*cos(1*w_low.*q) + b1_low*sin(1*w_low.*q) +
 a2_low*cos(2*w_low.*q) + b2_low*sin(2*w_low.*q) + ...
    a3_low*cos(3*w_low.*q) + b3_low*sin(3*w_low.*q) +
 a4_low*cos(4*w_low.*q) + b4_low*sin(4*w_low.*q) + ...
    a5_low*cos(5*w_low.*q) + b5_low*sin(5*w_low.*q);      %fourier5

figure(8)
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('$\eta_i$ [-]','Interpreter','latex','FontSize',16)
title('Fourier series $\eta_i ( q )$ curves
 (n=6)','Interpreter','latex','FontSize',16)
hold on

for i=1:length(x)
    y = (1-x(i))*f_high + x(i)*f_low;
    txt=['$\Omega = $',num2str(round(s_query(i),2))];
    plot(q,y,'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10,'Location','south')
end

axis([0 2 0 1.05])
hold off
legend show


%-------------------------------------------------------------------------
% 3. TWO-TERM EXPONENTIAL FUNCTIONS
%    e1(q) for q<=1, e2(q) for q>=1
%-------------------------------------------------------------------------

q=linspace(0,1.6,161);

q1=q(find(q<=1));   q2=q(find(q>1));

%High head:
f_high_1 = exp1_a_high*exp(exp1_b_high*q1) +
 exp1_c_high*exp(exp1_d_high*q1);   %q<=1
f_high_2 = exp2_a_high*exp(exp2_b_high*q2) +
 exp2_c_high*exp(exp2_d_high*q2);   %q>1
f_high = [f_high_1,f_high_2];
%Low head:
f_low_1 = exp1_a_low*exp(exp1_b_low*q1) +
 exp1_c_low*exp(exp1_d_low*q1);        %q<=1
f_low_2 = exp2_a_low*exp(exp2_b_low*q2) +
 exp2_c_low*exp(exp2_d_low*q2);        %q>1
f_low = [f_low_1,f_low_2];
```

```
figure(9)
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('$\eta_i$ [-]','Interpreter','latex','FontSize',16)
title('$\eta_i ( q )$ Exponential
 curves','Interpreter','latex','FontSize',16)
hold on

for i=1:length(x)
    y = (1-x(i))*f_high + x(i)*f_low;
    txt=['$\Omega = $',num2str(round(s_query(i),2))];
    plot(q,y,'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10,'Location','south')
end

axis([0 2 0 1.05])
hold off
legend show


%-------------------------------------------------------------------------
% 4. TWO-TERM POWER FUNCTIONS
%    pow1(q) for q<=1, pow2(q) for q>=1
%-------------------------------------------------------------------------

q1=q(find(q<=1));   q2=q(find(q>1));

%High head:
f_high_1 = pow1_a_high*q1.^pow1_b_high + pow1_c_high;        %q<=1
f_high_2 = pow2_a_high*q2.^pow2_b_high + pow2_c_high;        %q>1
f_high = [f_high_1,f_high_2];
%Low head:
f_low_1 = pow1_a_low*q1.^pow1_b_low + pow1_c_low;            %q<=1
f_low_2 = pow2_a_low*q2.^pow2_b_low + pow2_c_low;            %q>1
f_low = [f_low_1,f_low_2];

figure(10)
xlabel('q [-]','Interpreter','latex','FontSize',16)
ylabel('$\eta_i$ [-]','Interpreter','latex','FontSize',16)
title('$\eta_i ( q )$ Power
 curves','Interpreter','latex','FontSize',16)
hold on

for i=1:length(x)
    y = (1-x(i))*f_high + x(i)*f_low;
    txt=['$\Omega = $',num2str(round(s_query(i),2))];
    plot(q,y,'-','LineWidth',1,'DisplayName',txt);
    set(legend,'Interpreter','latex','FontSize',10,'Location','south')
end

axis([0 2 0 1.05])
hold off
legend show
```

Hill diagram construction

- Script name: $hillchart\_meas\_all\_turbines.m$
- Purpose: Generates Hill charts from the Rainpower measurements or any set of experimental data, which are loaded into MATLAB from Excel. Elevation levels for the contour lines must be specified. This script constructs diagrams for all three turbines: Figure 1, 2 and 3 corresponds to the high, medium and low head runner, respectively.
- Script name: $hillchart\_model\_no\_eta\_i\_all\_turbines.m$
- Purpose: Reads model input columns from the turbine design Excel sheet, and generates Hill charts from the model equations **without** η_i. Elevation levels for the contour lines must be specified. This script constructs diagrams for all three turbines: Figure 1, 2 and 3 corresponds to the high, medium and low head runner, respectively.
- Script name: $hillchart\_model\_with\_eta\_i\_highH.m$, $hillchart\_model\_with\_eta\_i\_medH.m$, $hillchart\_model\_with\_eta\_i\_lowH.m$
- Purpose: Reads model input column from the turbine design Excel sheet, and generates a Hill chart from the model equations **with a general** η_i. The $\eta_i(q)$ function must be specified by commenting out all the other functions one do not want to apply, at the bottom of the script. Before running, the correct $eta\_i\_xxx.m$ script must be run first to set all necessary curve coefficients as global variables. Elevation levels for the contour lines must be specified. "Validation areas" (unit speed and flow ranges) corresponding to the Rainpower data are clearly marked as a dashed rectangle.

hillchart_meas_all_turbines.m:

```
% Hill charts from Rainpower measurements - all 3 turbine types

clear all
close all
clc

de=0.025;  %for equdistant elevation curves


%--------------------------------------------------------------------------
% High head turbine
%--------------------------------------------------------------------------

%load data from Rainpower file and sort it
data=table2array(readtable('Hillchart_r3_medBEP.xlsx','Sheet','Høy',...
                'Range','B2:F257'));    %[etaM Q11 N11 T11 alfa]
etaM=data(:,1); Q11=data(:,2); N11=data(:,3); T11=data(:,4);
 alfa=data(:,5);

%remove data for negative efficiencies (irrelevant)
positive=find(etaM>=0);
etaM=etaM(positive); Q11=Q11(positive); N11=N11(positive);
 T11=T11(positive); alfa=alfa(positive);

%find rated values according to best efficiency point (BEP)
etaM_R=max(etaM);
pos=find(etaM==etaM_R);
Q11_R=Q11(pos); N11_R=N11(pos); T11_R=T11(pos); alfa_R=alfa(pos);

%non-dimensionalize by dividing by rated values
eta=etaM./etaM_R; q=Q11./Q11_R; n=N11./N11_R; t=T11./T11_R;
 y=sind(alfa)./sind(alfa_R);

n_min=min(n)        % speed range lower limit
n_max=max(n)        % speed range upper limit
q_min=min(q)        % flow range lower limit
q_max=max(q)        % flow range upper limit

%--------------------------------------------------------------------------
y_query=linspace(0.2,1.4,13);                   %y=1.0 included

dn=0.01;
nhalf1=linspace(min(n),1,ceil((1-min(n))/dn));
nhalf2=linspace(1,max(n),ceil((max(n)-1)/dn));
n_query=[nhalf1 nhalf2(2:end)];                 %n=1.0 included

dq=0.01;
qhalf1=linspace(min(q),1,ceil((1-min(q))/dq));
qhalf2=linspace(1,max(q),ceil((max(q)-1)/dq));
q_query=[qhalf1 qhalf2(2:end)];                 %q=1.0 included
%--------------------------------------------------------------------------

[N,Q]=meshgrid(n_query,q_query);                %each column in N has
 const speed
                                                %each row in Q has
 const flow
F_ETA=scatteredInterpolant(n,q,eta,'natural');  %create interpolant
 that fits a surface on the form eta=F_ETA(n,q)
ETA=F_ETA(N,Q);                                 %evaluate F_ETA at
 query points specified by N,Q matrices

for i=3:length(n_query-2)
    ETA(:,i)=smooth(ETA(:,i));                  %comment out for raw
 data
end

elev=[linspace(0,0.8,17) linspace(0.825,1.00,8)];
%elev=0:de:1;

figure(1)
contour(N,Q,ETA,elev,'-','LineWidth',1)
legend('$\tilde{\eta}$')
colorbar
caxis([0 1])
xlabel('$\tilde{\omega} / \sqrt{h}$
 [-]','Interpreter','latex','FontSize',16)
ylabel('$q / \sqrt{h}$ [-]','Interpreter','latex','FontSize',16)
title('Measurements: Hill
 diagram','Interpreter','latex','FontSize',16)
hold on

[Y,N]=meshgrid(y_query,n_query);                %each column in Y has
 const opening degree
                                                %each row in N has
 const speed
F_Q=scatteredInterpolant(y,n,q,'natural');      %create interpolant
 that fits a surface on the form q=F_Q(y,n)
Q=F_Q(Y,N);                                     %evaluate F_Q at query
 points specified by Y,N matrices

for i=1:length(y_query)                         %plot one curve per
 opening degree
    Q(3:end-2,i)=smooth(Q(3:end-2,i));          %comment out for raw
 data
    txt=['y = ',num2str(y_query(i))];
    plot(N(2:end,i),Q(2:end,i),'-','LineWidth',1,'DisplayName',txt);
  %exclude lowest speed and flow from plot
end

set(legend,'Interpreter','latex','FontSize',10,'Location','west')
axis([0 1.8 0 1.8])
hold off
legend show
```

```
% Medium head turbine
%--------------------------------------------------------------------------

%load data from Rainpower file and sort it
data=table2array(readtable('Hillchart_r3_medBEP.xlsx','Sheet','Mellom2',...
                'Range','L2:P305'));    %[etaM Q11 N11 T11 alfa]
etaM=data(:,1); Q11=data(:,2); N11=data(:,3); T11=data(:,4);
 alfa=data(:,5);

%remove data for N11=0 bc it gives h,q,t->Inf later
nonzero=find(N11~=0);
etaM=etaM(nonzero); Q11=Q11(nonzero); N11=N11(nonzero);
 T11=T11(nonzero); alfa=alfa(nonzero);

%remove data for negative efficiencies (irrelevant)
positive=find(etaM>=0);
etaM=etaM(positive); Q11=Q11(positive); N11=N11(positive);
 T11=T11(positive); alfa=alfa(positive);

%find rated values according to best efficiency point (BEP)
etaM_R=max(etaM);
pos=find(etaM==etaM_R);
Q11_R=Q11(pos); N11_R=N11(pos); T11_R=T11(pos); alfa_R=alfa(pos);

%non-dimensionalize by dividing by rated values
eta=etaM./etaM_R; q=Q11./Q11_R; n=N11./N11_R; t=T11./T11_R;
 y=sind(alfa)./sind(alfa_R);

n_min=min(n)        % speed range lower limit
n_max=max(n)        % speed range upper limit
q_min=min(q)        % flow range lower limit
q_max=max(q)        % flow range upper limit

%--------------------------------------------------------------------------
y_query=linspace(0.3,1.4,12);                   %y=1.0 included

dn=0.01;
nhalf1=linspace(min(n),1,ceil((1-min(n))/dn));
nhalf2=linspace(1,max(n),ceil((max(n)-1)/dn));
n_query=[nhalf1 nhalf2(2:end)];                 %n=1.0 included

dq=0.01;
qhalf1=linspace(min(q),1,ceil((1-min(q))/dq));
qhalf2=linspace(1,max(q),ceil((max(q)-1)/dq));
q_query=[qhalf1 qhalf2(2:end)];                 %q=1.0 included
%--------------------------------------------------------------------------

[N,Q]=meshgrid(n_query,q_query);                %each column in N has
 const speed
                                                %each row in Q has
 const flow
F_ETA=scatteredInterpolant(n,q,eta,'natural');  %create interpolant
 that fits a surface on the form eta=F_ETA(n,q)
ETA=F_ETA(N,Q);                                 %evaluate F_ETA at
 query points specified by N,Q matrices

for i=3:length(n_query-2)
    ETA(:,i)=smooth(ETA(:,i));                  %comment out for raw
 data
end

elev=[linspace(0,0.8,17) linspace(0.825,1.00,8)];
%elev=0:de:1;

figure(2)
contour(N,Q,ETA,elev,'-','LineWidth',1)
legend('$\tilde{\eta}$')
colorbar
caxis([0 1])
xlabel('$\tilde{\omega} / \sqrt{h}$
 [-]','Interpreter','latex','FontSize',16)
ylabel('$q / \sqrt{h}$ [-]','Interpreter','latex','FontSize',16)
title('Measurements: Hill
 diagram','Interpreter','latex','FontSize',16)
hold on

[Y,N]=meshgrid(y_query,n_query);                %each column in Y has
 const opening degree
                                                %each row in N has
 const speed
F_Q=scatteredInterpolant(y,n,q,'natural');      %create interpolant
 that fits a surface on the form q=F_Q(y,n)
Q=F_Q(Y,N);                                     %evaluate F_Q at query
 points specified by Y,N matrices

for i=1:length(y_query)                         %plot one curve per
 opening degree
    Q(3:end-2,i)=smooth(Q(3:end-2,i));          %comment out for raw
 data
    txt=['y = ',num2str(y_query(i))];
    plot(N(3:end,i),Q(3:end,i),'-','LineWidth',1,'DisplayName',txt);
  %exclude lowest speed and flow from plot
end

set(legend,'Interpreter','latex','FontSize',10,'Location','west')
axis([0 1.8 0 1.8])
hold off
legend show
```

```
%--------------------------------------------------------------------------
% Low head turbine
%--------------------------------------------------------------------------

%load data from Rainpower file and sort it
data=table2array(readtable('Hillchart_r3_medBEP.xlsx','Sheet','Lavtrykk',..
```

```
                        'Range','B2:F119'));    %[etaM Q11 N11 T11
 alfa]
etaM=data(:,1); Q11=data(:,2); N11=data(:,3); T11=data(:,4);
 alfa=data(:,5);

%remove data for negative efficiencies (irrelevant)
positive=find(etaM>=0);
etaM=etaM(positive); Q11=Q11(positive); N11=N11(positive);
 T11=T11(positive); alfa=alfa(positive);

%find rated values according to best efficiency point (BEP)
etaM_R=max(etaM);
pos=find(etaM==etaM_R);
Q11_R=Q11(pos); N11_R=N11(pos); T11_R=T11(pos); alfa_R=alfa(pos);

%non-dimensionalize by dividing by rated values
eta=etaM./etaM_R; q=Q11./Q11_R; n=N11./N11_R; t=T11./T11_R;
 y=sind(alfa)./sind(alfa_R);

n_min=min(n)        % speed range lower limit
n_max=max(n)        % speed range upper limit
q_min=min(q)        % flow range lower limit
q_max=max(q)        % flow range upper limit

%-------------------------------------------------------------------------
y_query=linspace(0.3,1.3,11);                   %y=1.0 included

dn=0.01;
nhalf1=linspace(min(n),1,ceil((1-min(n))/dn));
nhalf2=linspace(1,max(n),ceil((max(n)-1)/dn));
n_query=[nhalf1 nhalf2(2:end)];                 %n=1.0 included

dq=0.01;
qhalf1=linspace(min(q),1,ceil((1-min(q))/dq));
qhalf2=linspace(1,max(q),ceil((max(q)-1)/dq));
q_query=[qhalf1 qhalf2(2:end)];                 %q=1.0 included
%-------------------------------------------------------------------------

[N,Q]=meshgrid(n_query,q_query);                %each column in N has
 const speed
                                                %each row in Q has
 const flow
F_ETA=scatteredInterpolant(n,q,eta,'natural');  %create interpolant
 that fits a surface on the form eta=F_ETA(n,q)
ETA=F_ETA(N,Q);                                 %evaluate F_ETA at
 query points specified by N,Q matrices

for i=3:length(n_query-2)
    ETA(:,i)=smooth(ETA(:,i));                  %comment out for raw
 data
end

elev=[linspace(0,0.8,17) linspace(0.825,1.00,8)];
%elev=0:de:1;
```

```
figure(3)
contour(N,Q,ETA,elev,'-','LineWidth',1)
legend('$\tilde{\eta}$')
colorbar
caxis([0 1])
xlabel('$\tilde{\omega} / \sqrt{h}$
 [-]','Interpreter','latex','FontSize',16)
ylabel('$q / \sqrt{h}$ [-]','Interpreter','latex','FontSize',16)
title('Measurements: Hill
 diagram','Interpreter','latex','FontSize',16)
hold on

[Y,N]=meshgrid(y_query,n_query);                %each column in Y has
 const opening degree
                                                %each row in N has
 const speed
F_Q=scatteredInterpolant(y,n,q,'natural');      %create interpolant
 that fits a surface on the form q=F_Q(y,n)
Q=F_Q(Y,N);                                     %evaluate F_Q at query
 points specified by Y,N matrices

for i=1:length(y_query)                         %plot one curve per
 opening degree
    Q(3:end-2,i)=smooth(Q(3:end-2,i));          %comment out for raw
 data
    txt=['y = ',num2str(y_query(i))];

 plot(N(2:end-1,i),Q(2:end-1,i),'-','LineWidth',1,'DisplayName',txt);
  %exclude lowest and highest speed and flow from plot
end

set(legend,'Interpreter','latex','FontSize',10,'Location','west')
axis([0 1.8 0 1.8])
hold off
legend show
```

hillchart_model_no_eta_i_all_turbines.m:

```
% Hill charts from non-linear turbine model without eta_i
 implementation - all 3 turbine types

clear all
close all
clc

de=0.025;  %for equdistant elevation curves


%-------------------------------------------------------------------------
% High head turbine
%-------------------------------------------------------------------------

% Design parameters from excel sheet - choose turbine by specifying
 column
 design=table2array(readtable('Hillchart_r3_medBEP.xlsx','Sheet',...
                    'Turbin_design','Range','U9:U80'));
% necessary model input:
 alfa_1R=design(49,1); sigma=design(53,1); psi=design(54,1);
 xi=design(56,1);

%-------------------------------------------------------------------------
y_query=linspace(0.2,1.4,13);                   %y=1.0 included
n_query=linspace(0,1.56,157);                   %n=1.0 included
q_query=linspace(0,2,201);                      %q=1.0 included
h=1;

n_query2=linspace(0.81,1.30,100);
q_query2=linspace(0.06,1.70,100);
%-------------------------------------------------------------------------

figure(1)
axis([0 1.8 0 1.8])
xlabel('$\tilde{\omega}$ [-]','Interpreter','latex','FontSize',16)
ylabel('$q$ [-]','Interpreter','latex','FontSize',16)
title('Model: Hill diagram at
 h=1','Interpreter','latex','FontSize',16)
set(legend,'Interpreter','latex','FontSize',10,'Location','west')
hold on

%torque
t=zeros(length(q_query),length(n_query));       %each row const flow,
 each column const speed
for j=1:length(n_query)
    qdivy=sqrt(h-sigma*(n_query(j)^2-1));
    y=q_query./qdivy;
    alfa_1=asin(y.*sin(alfa_1R*pi/180));        %radians
    m_S=xi*qdivy*(cos(alfa_1)+tan(alfa_1R*pi/180)*sin(alfa_1));
    t(:,j)=q_query.*(m_S - psi.*n_query(j));
end
```

```
%efficiency
eta=zeros(length(q_query),length(n_query));     %each row const flow,
 each column const speed
for i=1:length(q_query)
    eta(i,:)=(t(i,:).*n_query)./(q_query(i)*h);
end

[N,Q]=meshgrid(n_query,q_query);                %each column in N has
 const speed
                                                %each row in Q has
 const flow
ETA=real(eta);

elev=[linspace(0,0.8,17) linspace(0.825,1.00,8)];
%elev=0:de:1;

contour(N,Q,ETA,elev,'-','LineWidth',1)
colorbar
caxis([0 1])

%flow
q=zeros(length(y_query),length(n_query));       %each row const
 opening, each column const speed
for i=1:length(y_query)
    q(i,:)=y_query(i)*sqrt(h-sigma*(n_query.^2-1));

    %plot one curve per opening degree
    plot(n_query,q(i,:),'-','LineWidth',1)
end

plot(n_query2,ones(length(n_query2))*min(q_query2),'k--',n_query2,ones(leng

 ones(length(q_query2))*min(n_query2),q_query2,'k--',ones(length(q_query2))
    'LineWidth',1)

legend('$\tilde{\eta}$','y = 0.2','y = 0.3','y = 0.4','y = 0.5','y =
 0.6','y = 0.7','y = 0.8','y = 0.9',...
       'y = 1.0','y = 1.1','y = 1.2','y = 1.3','y = 1.4')


%-------------------------------------------------------------------------
% Medium head turbine
%-------------------------------------------------------------------------

% Design parameters from excel sheet - choose turbine by specifying
 column
 design=table2array(readtable('Hillchart_r3_medBEP.xlsx','Sheet',...
                    'Turbin_design','Range','T9:T80'));

% necessary model input:
 alfa_1R=design(49,1); sigma=design(53,1); psi=design(54,1);
 xi=design(56,1);

%-------------------------------------------------------------------------
```

```
y_query=linspace(0.3,1.4,12);                   %y=1.0 included
n_query=linspace(0,1.78,179);                   %n=1.0 included
q_query=linspace(0,2,201);                      %q=1.0 included
h=1;

n_query2=linspace(0.86,1.71,100);
q_query2=linspace(0.08,1.48,100);
%-------------------------------------------------------------------------

figure(2)
axis([0 1.8 0 1.8])
xlabel('$\tilde{\omega}$ [-]','Interpreter','latex','FontSize',16)
ylabel('$q$ [-]','Interpreter','latex','FontSize',16)
title('Model: Hill diagram at
 h=1','Interpreter','latex','FontSize',16)
set(legend,'Interpreter','latex','FontSize',10,'Location','west')
hold on

%torque
t=zeros(length(q_query),length(n_query));       %each row const flow,
 each column const speed
for j=1:length(n_query)
    qdivy=sqrt(h-sigma*(n_query(j)^2-1));
    y=q_query./qdivy;
    alfa_1=asin(y.*sin(alfa_1R*pi/180));        %radians
    m_S=xi*qdivy*(cos(alfa_1)+tan(alfa_1R*pi/180)*sin(alfa_1));
    t(:,j)=q_query.*(m_S - psi.*n_query(j));
end

%efficiency
eta=zeros(length(q_query),length(n_query));     %each row const flow,
 each column const speed
for i=1:length(q_query)
    eta(i,:)=(t(i,:).*n_query)./(q_query(i)*h);
end

[N,Q]=meshgrid(n_query,q_query);                %each column in N has
 const speed
                                                %each row in Q has
 const flow
ETA=real(eta);

elev=[linspace(0,0.8,17) linspace(0.825,1.00,8)];
%elev=0:de:1;

contour(N,Q,ETA,elev,'-','LineWidth',1)
colorbar
caxis([0 1])

%flow
q=zeros(length(y_query),length(n_query));       %each row const
 opening, each column const speed
for i=1:length(y_query)
    q(i,:)=y_query(i)*sqrt(h-sigma*(n_query.^2-1));

    %plot one curve per opening degree
    plot(n_query,q(i,:),'-','LineWidth',1)
end

plot(n_query2,ones(length(n_query2))*min(q_query2),'k--',n_query2,ones(leng

 ones(length(q_query2))*min(n_query2),q_query2,'k--',ones(length(q_query2))
    'LineWidth',1)

legend('$\tilde{\eta}$','y = 0.3','y = 0.4','y = 0.5','y = 0.6','y =
 0.7','y = 0.8','y = 0.9',...
       'y = 1.0','y = 1.1','y = 1.2','y = 1.3','y = 1.4')


%-------------------------------------------------------------------------
% Low head turbine
%-------------------------------------------------------------------------

% Design parameters from excel sheet - choose turbine by specifying
 column
 design=table2array(readtable('Hillchart_r3_medBEP.xlsx','Sheet','Turbin_de

% necessary model input:
 alfa_1R=design(49,1); sigma=design(53,1); psi=design(54,1);
 xi=design(56,1);

%-------------------------------------------------------------------------
y_query=linspace(0.3,1.3,11);                   %y=1.0 included
n_query=linspace(0,1.9,191);                    %n=1.0 included
q_query=linspace(0,2,201);                      %q=1.0 included
h=1;

n_query2=linspace(0.94,1.47,100);
q_query2=linspace(0.32,1.34,100);
%-------------------------------------------------------------------------

figure(3)
axis([0 1.8 0 1.8])
xlabel('$\tilde{\omega}$ [-]','Interpreter','latex','FontSize',16)
ylabel('$q$ [-]','Interpreter','latex','FontSize',16)
title('Model: Hill diagram at
 h=1','Interpreter','latex','FontSize',16)
set(legend,'Interpreter','latex','FontSize',10,'Location','west')
hold on

%torque
t=zeros(length(q_query),length(n_query));       %each row const flow,
 each column const speed
for j=1:length(n_query)
    qdivy=sqrt(h-sigma*(n_query(j)^2-1));
    y=q_query./qdivy;
    alfa_1=asin(y.*sin(alfa_1R*pi/180));        %radians
    m_S=xi*qdivy*(cos(alfa_1)+tan(alfa_1R*pi/180)*sin(alfa_1));
```


```
    t(:,j)=q_query.*(m_S - psi.*n_query(j));
end

%efficiency
eta=zeros(length(q_query),length(n_query));     %each row const flow,
 each column const speed
for i=1:length(q_query)
    eta(i,:)=(t(i,:).*n_query)./(q_query(i)*h);
end

[N,Q]=meshgrid(n_query,q_query);                %each column in N has
 const speed
                                                %each row in Q has
 const flow
ETA=real(eta);

elev=[linspace(0,0.8,17) linspace(0.825,1.00,8)];
%elev=0:de:1;

contour(N,Q,ETA,elev,'-','LineWidth',1)
colorbar
caxis([0 1])

%flow
q=zeros(length(y_query),length(n_query));       %each row const
 opening, each column const speed
for i=1:length(y_query)
    q(i,:)=y_query(i)*sqrt(h-sigma*(n_query.^2-1));

    %plot one curve per opening degree
    plot(n_query,q(i,:),'-','LineWidth',1)
end

plot(n_query2,ones(length(n_query2))*min(q_query2),'k--',n_query2,ones(leng

 ones(length(q_query2))*min(n_query2),q_query2,'k--',ones(length(q_query2))
    'LineWidth',1)

legend('$\tilde{\eta}$','y = 0.3','y = 0.4','y = 0.5','y = 0.6','y =
 0.7','y = 0.8','y = 0.9',...
       'y = 1.0','y = 1.1','y = 1.2','y = 1.3')
```

hillchart_model_with_eta_i_highH.m:

```
% Hill charts from non-linear turbine model with eta_i implementation
 specified in eta_i functions at the bottom of script

% clear all
close all
clc


%--------------------------------------------------------------------------
% High head turbine
%--------------------------------------------------------------------------

% Design parameters from excel sheet - choose turbine by specifying
 column, run its 'eta_i_xxx' script before this script
 design=table2array(readtable('Hillchart_r3_medBEP.xlsx','Sheet',...
                  'Turbin_design','Range','U9:U80'));

% necessary model input:
 alfa_1R=design(49,1); sigma=design(53,1); psi=design(54,1);
 xi=design(56,1);

%--------------------------------------------------------------------------
y_query=linspace(0.2,1.4,13);                   %y=1.0 included
n_query=linspace(0,1.55,156);                   %n=1.0 included
q_query=linspace(0.04,2,197);                   %q=1.0 included
h=1;

n_query2=linspace(0.81,1.30,100);
q_query2=linspace(0.06,1.70,100);
%--------------------------------------------------------------------------

figure(1)
axis([0 1.8 0 1.8])
xlabel('$\tilde{\omega}$ [-]','Interpreter','latex','FontSize',16)
ylabel('$q$ [-]','Interpreter','latex','FontSize',16)
title('Model with $\eta_i(q)$: Hill diagram at
 h=1','Interpreter','latex','FontSize',16)
set(legend,'Interpreter','latex','FontSize',10,'Location','west')
hold on

%torque
t=zeros(length(q_query),length(n_query));       %each row const flow,
 each column const speed
for j=1:length(n_query)
    qdivy=sqrt(h-sigma*(n_query(j)^2-1));
    y=q_query./qdivy;
    alfa_1=asin(y.*sin(alfa_1R*pi/180));        %radians
    m_S=xi*qdivy*(cos(alfa_1)+tan(alfa_1R*pi/180)*sin(alfa_1));

    t(:,j)=eta_i(q_query).*q_query.*(m_S - psi.*n_query(j));
end
```

```
%efficiency
eta=zeros(length(q_query),length(n_query));     %each row const flow,
 each column const speed
for i=1:length(q_query)
    eta(i,:)=(t(i,:).*n_query)./(q_query(i)*h);
end

[N,Q]=meshgrid(n_query,q_query);                %each column in N has
 const speed
                                                %each row in Q has
 const flow
ETA=real(eta);

%elev=[linspace(0.05,0.8,16) linspace(0.825,1.00,8)];
elev=[0.01 linspace(0.05,0.8,16) linspace(0.825,1.00,8)];

contour(N,Q,ETA,elev,'-','LineWidth',1)
colorbar
caxis([0 1])

%flow
q=zeros(length(y_query),length(n_query));       %each row const
 opening, each column const speed
for i=1:length(y_query)
    q(i,:)=y_query(i)*sqrt(h-sigma*(n_query.^2-1));

    %plot one curve per opening degree
    plot(n_query,q(i,:),'-','LineWidth',1)
end

plot(n_query2,ones(length(n_query2))*min(q_query2),'k--',...
    n_query2,ones(length(n_query2))*max(q_query2),'k--',...
    ones(length(q_query2))*min(n_query2),q_query2,'k--',...
    ones(length(q_query2))*max(n_query2),q_query2,'k--',...
    'LineWidth',1)

legend('$\tilde{\eta}$','y = 0.2','y = 0.3','y = 0.4','y = 0.5',...
       'y = 0.6','y = 0.7','y = 0.8','y = 0.9','y = 1.0',...
       'y = 1.1','y = 1.2','y = 1.3','y = 1.4')

%--------------------------------------------------------------------------

% uncomment the relevant eta_i(q) function
% eta_i multiplied into torque equation, should affect t-, p- and eta-
plots
% q = single flow value/ vector at constant speed/ opening degree

%--------------------------------------------------------------------------

% 1. POLYNOMIAL FUNCTION
function y=eta_i(q)
    global poly_p1 poly_p2 poly_p3 poly_p4 poly_p5 poly_p6 poly_p7
 poly_p8 poly_p9 poly_p10
```

```
    y = poly_p1*q.^9 + poly_p2*q.^8 + poly_p3*q.^7 + poly_p4*q.^6 +
 poly_p5*q.^5 + poly_p6*q.^4 + ...
        poly_p7*q.^3 + poly_p8*q.^2 + poly_p9*q + poly_p10;
end

% 2. FOURIER SERIES FUNCTION
function y=eta_i(q)
    global fou_a0 fou_a1 fou_b1 fou_a2 fou_b2 fou_a3 fou_b3 fou_a4
 fou_b4 fou_a5 fou_b5 fou_a6 fou_b6 fou_w
    y = fou_a0 + fou_a1*cos(1*fou_w.*q) + fou_b1*sin(1*fou_w.*q) +
 fou_a2*cos(2*fou_w.*q) + fou_b2*sin(2*fou_w.*q)...
    + fou_a3*cos(3*fou_w.*q) + fou_b3*sin(3*fou_w.*q) +
 fou_a4*cos(4*fou_w.*q) + fou_b4*sin(4*fou_w.*q)...
    + fou_a5*cos(5*fou_w.*q) + fou_b5*sin(5*fou_w.*q) +
 fou_a6*cos(6*fou_w.*q) + fou_b6*sin(6*fou_w.*q);
end

% 3. TWO-TERM EXPONENTIAL FUNCTIONS
function y=eta_i(q)
    global expo1_a expo1_b expo1_c expo1_d expo2_a expo2_b expo2_c
 expo2_d
    y=zeros(size(q));

    for k=1:length(q)
        if q(k)<=1
            y(k) = expo1_a*exp(expo1_b*q(k)) +
 expo1_c*exp(expo1_d*q(k));
        elseif q(k)>1
            y(k) = expo2_a*exp(expo2_b*q(k)) +
 expo2_c*exp(expo2_d*q(k));
        end
    end
end

% 4. TWO-TERM POWER FUNCTIONS
function y=eta_i(q)
    global pow1_a pow1_b pow1_c pow2_a pow2_b pow2_c
    y=zeros(size(q));

    for k=1:length(q)
        if q(k)<1
            y(k) = pow1_a*q(k)^pow1_b + pow1_c;
        elseif q(k)>=1
            y(k) = pow2_a*q(k)^pow2_b + pow2_c;
        end
    end
end

% 5. PARABOLA PROPOSED BY NIELSEN
function y=eta_i(q)
    y=q.*(2-q);
end
```