Master of Science Thesis in Electrical Engineering
Department of Electrical Engineering, Linköping University, 2023

Parameter Estimation in a Permanent Magnet Synchronous Motor

Martin Axelsson and Jesper Barreng

LINKÖPING UNIVERSITY

Master of Science Thesis in Electrical Engineering

Parameter Estimation in a Permanent Magnet Synchronous Motor

Martin Axelsson and Jesper Barreng

LiTH-ISY-EX--23/5595--SE

Supervisor: **Filipe Barbosa**
ISY, Linköpings universitet
Fredrik Zachrisson
Mechatronics Design Team, Atlas Copco

Examiner: **Svante Gunnarsson**
ISY, Linköpings universitet

Division of Automatic Control
Department of Electrical Engineering
Linköping University
SE-581 83 Linköping, Sweden

Sammanfattning

Denna masteruppsats undersökte möjligheten att estimera temperaturberoende parametrar i en synkronmotor med permanentmagneter med hjälp av Kalmanfilter. Fokus låg på att estimera motorns resistans och magnetiska flöde för att övervaka temperaturvariationer under motordrift. Användningen av Kalmanfilter motiverades av deras förmåga att hantera icke-linjäriteter och osäkerheter. Detta gjorde dem lämpliga för att hantera variationerna som uppstår under en typisk åtdragning av ett skruvförband, vid användning av ett verktyg från Atlas Copco.

De första iterationerna av Kalmanfiltrets tillståndsmodell fokuserade främst på att uppskatta resistansen. Därefter utvecklades en liknande version för uppskattning av det magnetiska flödet. Slutligen kombinerades dessa två versioner för att skapa den slutgiltiga modellen, vilket möjliggjorde samtidig estimering av både resistansen och det magnetiska flödet. masteruppsatsen utforskade tre olika varianter av Kalman-filter: Extended Kalman Filter, Adaptive Extended Kalman Filter och Unscented Kalman Filter. Syftet med masteruppsatsen var att utvärdera prestandan hos dessa filter och identifiera eventuella skillnader som uppstår på grund av deras olikheter.

Övergången från simulerad data till verkliga data tydliggjorde begränsningarna med att enbart förlita sig på simulerade modeller. Tester på verklig data avslöjade komplexiteter och osäkerheter som förbises i de förenklade simuleringsmodellerna. Särskilt närvaron av brus i signalerna blev tydligt. Även om Kalmanfilterna kunde uppskatta resistansen och det magnetiska flödet samtidigt utan brus, medförde närvaron av brus utmaningar för samtliga varianter av Kalmanfilter. Estimeringsprestandan var bristfällig och indikerade att nuvarande nivå av noggrannhet inte var tillräcklig för att tillförlitligt övervaka temperaturförändringar i permanentmagnet synkronmotor.

Förutom brus avslöjade datan från ett verktyg oväntade fenomen så som avvikelser mellan simuleringsmodellen och det verktyg som användes i masteruppsatsen, samt vikten av korrekt data och motorparametrar för att Kalmanfiltrena ska ge adekvata resultat.

Sammanfattningsvis betonade denna masteruppsats utmaningarna med samtidig estimering av resistans och magnetiskt flöde, samt begränsningarna i att enbart förlita sig på simuleringsdata. Masteruppsatsen visade potentialen hos Kalmanfilter för temperaturberoende parameterestimering i en permanentmagnet synkronmotor. Detta skulle i framtiden kunna möjliggöra en lösning utan sensorer. Ytterligare undersökningar och förbättringar krävs dock för att lösa de identifierade svårigheterna och uppnå tillförlitlig parameterestimering.

Abstract

This thesis investigated the feasibility of estimating temperature-dependent parameters in a permanent magnet synchronous motor using Kalman filters. The primary focus was on estimating the motor's stator winding resistance and rotor flux linkage to monitor temperature variations during motor operation. The utilisation of Kalman filters was motivated by their capability to handle nonlinearities and uncertainties, which made them well suited for addressing the variations encountered during the tightening operation at Atlas Copco.

The initial iterations of the Kalman filter's state transition vector primarily focused on estimating the resistance. Subsequently, a similar version was developed for flux estimation. Eventually, these two versions were combined to create the final model, enabling simultaneous estimation of both the resistance and flux. The thesis explored three different variants of the Kalman filter: the Extended Kalman Filter, the Adaptive Extended Kalman Filter, and the Unscented Kalman Filter. The thesis aimed to evaluate the performance of these filters and identify any differences.

The transition from simulated data to real world data illustrated the limitations of relying solely on simulation models. Testing on real world data uncovered complexities and uncertainties that were overlooked in the simplified simulation models. Notably, the presence of noise in the signals. While the Kalman filters were capable of estimating the resistance and flux simultaneously without noise, the introduction of noise posed challenges for all the Kalman filter variants. As a result, the estimation performance was poor, indicating that the current accuracy levels were insufficient to reliably monitor temperature changes in the PMSM.

In addition to noise, the real world data revealed unexpected phenomena such as discrepancies between the simulation model and the tool used in the thesis, as well as the importance of accurate data and motor parameters for the Kalman filters to yield adequate results.

Overall, this thesis emphasised the challenges associated with simultaneous estimation of resistance and flux, and the limitations of relying solely on simulations. This thesis demonstrated the potential of using Kalman filters for temperature dependent parameter estimation in a permanent magnet synchronous motor. A variant which in the future could enable a sensorless solution. However, further research and improvements are necessary to overcome the identified challenges and achieve reliable parameter estimation.

Acknowledgments

First and foremost, we would like to acknowledge the support and cooperation of Atlas Copco, particularly the professionals at Atlas Copco, who graciously allowed us access to their facilities and provided valuable resources for the practical aspects of this thesis. Their collaboration and willingness to share their expertise were pivotal in bridging the gap between theoretical concepts and real world applications.

A special thanks goes to our supervisor, Fredrik Zachrisson, for his unwavering support and invaluable guidance throughout this journey. His expertise and constructive feedback were indispensable in shaping the direction of this thesis and improving its quality.

We would also like to extend a heartfelt thanks to our examiner Svante Gunnarsson and supervisor Filipe Barbosa at Linköping University for their help throughout this thesis. Their feedback helped us with both shaping the thesis and clarify the academic aspect of writing a Master thesis.

To all those who played a part, big or small, in this endeavor, we offer our heartfelt thanks. Your contributions made a significant impact, and we are truly honored and privileged to have had the opportunity to work with and learn from each and every one of you.

Stockholm, June 2023
Martin Axelsson and Jesper Barreng

Contents

Notation

Nomenclature

Notation	Explanation
v_{sa}	Stator voltage in phase a
v_{sb}	Stator voltage in phase b
v_{sc}	Stator voltage in phase c
R_s	Resistance in stator winding
i_{sa}	Stator current in phase a
i_{sb}	Stator current in phase b
i_{sc}	Stator current in phase c
Ψ_{sa}	Stator flux in phase a
Ψ_{sb}	Stator flux in phase b
Ψ_{sc}	Stator flux in phase c
Ψ_{ra}	Rotor flux in phase a
Ψ_{rb}	Rotor flux in phase b
Ψ_{rc}	Rotor flux in phase c
p	Number of pole pairs
θ_r	Rotor angle
θ_{el}	Electric rotor angle
L_s	Stator inductance
ω_r	Angular velocity of rotor
ω_{el}	Electric angular velocity of rotor
i_α	Current in α-coordinates
i_β	Current in β-coordinates
i_d	Current in d-coordinates
i_q	Current in q-coordinates
T_{em}	Motor torque in the dq-frame
T_L	Load torque
f	Viscous damping constant
J	Rotor inertia

Acronyms

Abbreviation	Explanation
PMSM	Permanent Magnet Synchronous Motor
SPMSM	Surface-mounted Permanent Magnet Synchronous Motor
FOC	Field Oriented Control
EKF	Extended Kalman Filter
AEKF	Adaptive Extended Kalman Filter
UKF	Unscented Kalman Filter
UT	Unscented Transform
PI	Proportional Integral (regulator)

1

Introduction

Chapter 1 serves as a starting point, as it addresses the problem by delving into its appurtenant background. Furthermore, this chapter establishes the scope of the study, clarifying the boundaries within which the thesis is presented. The methodology is outlining the approach and techniques employed to gather and analyse data.

1.1 Background

Electrical motors are important components of modern society, driving everything from home appliances to industrial machinery. These motors consume approximately 45% of the world's electricity [22], making their efficiency and reliability crucial to conserving energy and reducing our carbon footprint. As the world continues to shift towards sustainable energy sources, the electrification of various sectors is becoming increasingly important. In this context, the development of robust and efficient electrical machinery is crucial.

Atlas Copco is a company that designs, develops, and manufactures electrical assembly systems. These systems are sold and distributed worldwide and are utilized in a variety of industries, including automotive, aerospace, and manufacturing.

One critical factor that impacts the lifespan of electrical motors is the operating temperature. Motors that operate at excessively high temperatures can suffer from irreversible demagnetization of the rotor magnets. Furthermore, the lifespan of motor winding insulation is markedly reduced when temperatures exceed the specified temperature rating. To ensure the proper functioning of electrical motors, accurate monitoring of winding temperature is crucial.

1.2 Problem Definition

At Atlas Copco, mechanical temperature sensors have been used to monitor stator winding temperature. However, these sensors add size, complexity, and cost to the machinery. Additionally, the current temperature sensor solution is often inaccurate due to the electrical insulation requirements. This inaccuracy can lead to premature motor failure.

To address these challenges, Atlas Copco is interested in developing temperature estimation algorithms for high-speed electrical machines used in industrial power tools. The goal of this thesis is to investigate the possibility of monitoring the temperature dependent parameters, winding resistance and rotor flux. This will be done by utilising a variety of Kalman filters.

1.3 Scope

The scope of this Master thesis is to investigate the application of Kalman filters for parameter estimation in Permanent Magnet Synchronous Motor (PMSM) controlled by a Field Oriented Control (FOC). The primary objective is to analyse the performance of three different Kalman filter variants: Extended Kalman Filter (EKF), Adaptive Extended Kalman Filter (AEKF), and Unscented Kalman Filter (UKF).

The thesis will involve utilising both simulated and real-world data to evaluate the applicability of the Kalman filters in estimating the flux and resistance of the PMSM. Simulated data will be employed to assess the performance of the filters under different conditions, while real-world data, involving an actual tightening operation will provide insights into their behavior in practical scenarios.

However, certain boundaries and limitations exist within this thesis. The FOC and PMSM models used will be simplified, lacking internal dynamics that are present in real motors. The parameter estimation will focus solely on estimating the flux and resistance. Additionally, the investigation will be simulation-based, without implementation into an actual tool. The number of test cases will be restricted, primarily focusing on real tightening operations. Lastly, the scope will be specifically centered around Surface-Mounted Permanent Magnet Synchronous Motor (SPMSMs).

1.4 Methodology

To address the problem formulation and gain a comprehensive understanding of the underlying components, an extensive research effort will be undertaken. This will involve delving into literature sources that cover topics such as FOC, PMSM,

and Kalman filters, more specific EKF, AEKF and UKF.

The research methodology employed in this thesis primarily revolves around conducting experiments using simulation models. These models offer the advantage of flexibility and rapid prototyping, enabling swift modifications that would be challenging to implement on a real tool. The Kalman filters, will be implemented as Matlab scripts due to the ease of performing mathematical calculations in Matlab.

Data collection for this thesis entails a combination of simulated and real world data. The simulated data will be obtained from model simulations in Simulink. Real world data will be acquired from an actual tool used at Atlas Copco. By employing both simulated and real-world measurements, it will be possible to conduct controlled experiments and compare the performance between simulated and real world data. Simulated data will be directly exported from Simulink to Matlab, while the real world data will be extracted using a dedicated software.

As the objective of the temperature monitoring system is to detect changes in temperature over time, the focus will be on observing how the underlying parameters evolve over time and comparing them to nominal values. By doing so, the implemented method can be evaluated.

One of the primary limitations could be the amount of relevant literature on the topic, which may present challenges in developing the simulation models and comprehending the chosen approach. Additionally, the data extracted from the real tool could be a limitation. In order for a Kalman filter to be applicable, it depends on how the data behaves. Furthermore, the unknown behavior of the tool compared to the literature could be a limitation. Meaning we do not know if the tool behaves in a similar way as it does in the literature. This is an assumption we have to make in order to proceed with this thesis.

1.5 Outline

This thesis is divided into eight chapters. Chapter 1, introduces the problem with appurtenant background. It also defines the scope and methodology together with an outline. Chapter 2 presents the theory that this thesis will build upon. In Chapter 3, the procedure of creating the FOC, PMSM and thermal model in simulation as well as the matrices used in the different Kalman filters are introduced. Chapter 4 and 5 introduces the setup for both simulation and real-world testing. It also includes the respective test cases, data and results. In Chapter 6 challenges that were encountered are addressed. Chapter 7 discusses the results and other miscellaneous subjects related to the thesis that the authors find important to mention. Lastly, Chapter 8 concludes the thesis as well as suggests how to proceed going forward.

2

Theory

In this chapter, an overview of the permanent magnet synchronous motor is provided. The mathematical model of the motor is presented, along with an explanation of the Field Oriented Controller's operational principle and its schematic diagram. Furthermore, various Kalman filters utilised to estimate the temperature dependent parameters are introduced. Lastly, theory that explains the simplified thermal development in a permanent magnet synchronous motor is presented.

2.1 Permanent Magnet Synchronous Motors

This section presents the basic construction of a PMSM, provides a description of the three-phase system, and explains the mathematical model of the PMSM. The mathematical model includes different transforms used to simplify the calculations.

2.1.1 Working Principle of a PMSM

Like any rotating electric motor, the PMSM consists of a rotating part, the rotor and a stationary part, the stator. A principal cross section for different types of magnetic configurations in a PMSM is visualised in Figure 2.1.

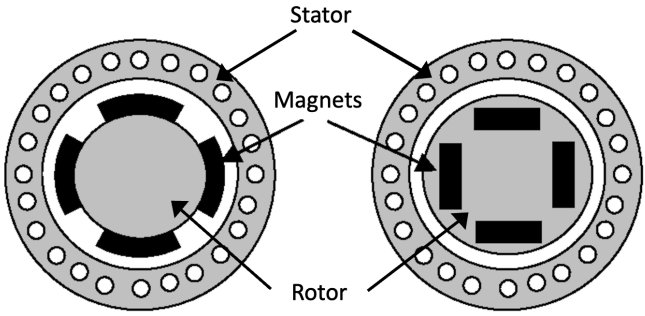


Figure 2.1: *Principal cross section of a Surface-Mounted Permanent Magnet Synchronous Motor (left) and an Interior-Mounted Permanent Magnet Synchronous Motor (right) [16].*

The air gap between the stator and rotor is essential as it enables the rotor to rotate freely. It serves as a necessary space that allows for unimpeded movement and prevents physical contact between the two components. In Atlas Copcos PMSM construction, both the rotor and stator are equipped with iron cores. The stator's iron core is then wound with copper wire, while the rotor's iron core is encircled by neodymium magnets. This design choice facilitates the smooth flow of magnetic flux generated by the permanent magnets.

The fundamental operational principle of a PMSM relies on the dynamic interplay between the magnetic fields of the stator and rotor. The rotor's magnetic field is established by the permanent magnets, which create a continuous magnetic flux. This magnetic field exhibits synchronous rotation with the field generated by the stator.

The synchronised rotation of the two magnetic fields creates an interaction that produces a torque on the rotor. The torque generated through this interaction is driving the rotational movement of the motor [12]. In the scope of this thesis, the permanent magnets are attached on the surface of the rotor and based on this, the motor is classified as a *surface-mounted permanent magnet synchronous motor (SPMSM)*.

2.1.2 The Y-connected Three Phase System

The magnetic field created by the stator is supplied via a Y-connected three phase system. In a three-phase system, the coils are energised with an alternating current, meaning that the direction of the current changes periodically. These coils are positioned 120 degrees apart from each other in a spatial distribution. This angular displacement ensures a balanced distribution of electrical power. Due to the spatial arrangement of the coils, the magnetic field rotates as the current flowing through the coils alternates.

The arrangement of the three phases within the system is commonly referred to as the *abc*-frame. In this configuration, phase a is associated with one of the coils, phase b with another, and phase c with the remaining coil. These three phases collectively form the foundation of the Y-connected three-phase system. Figure 2.2 illustrates the arrangement of the three phases in the *abc*-frame [3].

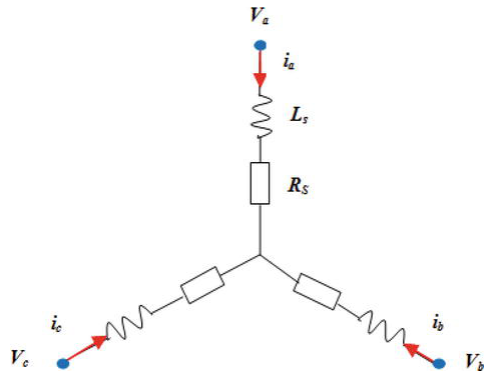


Figure 2.2: *Three phase Y-connection [20].*

2.1.3 Mathematical Model

The mathematical model consists of three separate parts describing the motor. The first is the electrical equations, the second is the electromagnetic equation and the third is the mechanical equation.

The electrical equations of the three-phases that constitute the stator can be written as (2.1) - (2.3).

$$v_{sa} = R_s i_{sa} + \frac{d\Psi_{sa}}{dt} \tag{2.1}$$

$$v_{sb} = R_s i_{sb} + \frac{d\Psi_{sb}}{dt} \tag{2.2}$$

$$v_{sc} = R_s i_{sc} + \frac{d\Psi_{sc}}{dt} \tag{2.3}$$

where, $v_{s,abc}$ denotes the voltage in each stator phase, R_s represents the stator resistance, $i_{s,abc}$ corresponds to the current flowing through each stator phase, and $\Psi_{s,abc}$ is the stator flux present in each stator phase.

A constant flux in the rotor is created due to the permanent magnets, and the electromagnetic force is assumed to be sinusoidal. Hence the stator flux can be expressed as in (2.4) - (2.6).

$$\Psi_{sa} = L_{ss} i_{sa} + \Psi_{ra} cos(p\theta_r) \tag{2.4}$$

$$\Psi_{sb} = L_{ss} i_{sb} + \Psi_{rb} cos(p\theta_r - \frac{2\pi}{3}) \tag{2.5}$$

$$\Psi_{sc} = L_{ss} i_{sc} + \Psi_{rc} cos(p\theta_r + \frac{2\pi}{3}) \tag{2.6}$$

where the term L_{ss} refers to the inductance matrix specific to the stator. $\Psi_{r,abc}$ is the amplitude of the flux induced in each stator phase by the permanent magnets. The constant p denotes the number of pole pairs in the motor, and θ_r represents the rotor angle [4]. Due to the machine used in this thesis, which is a SPMSM the inductance matrix L_{ss} can be modeled as a constant L_s [5].

By combining (2.1) - (2.3) with (2.4) - (2.6) via (2.7), it yields (2.8) - (2.10).

$$\frac{d}{dt}(\bullet) = \frac{d\theta_r}{dt}\frac{d}{d\theta_r}(\bullet) = \omega_{el}\frac{d}{d\theta_r}(\bullet) \tag{2.7}$$

$$v_{sa} = R_s i_{sa} + \frac{d}{dt}(L_s i_{sa}) + \omega_{el}\frac{d}{d\theta_r}(\Psi_{ra}) \tag{2.8}$$

$$v_{sb} = R_s i_{sb} + \frac{d}{dt}(L_s i_{sb}) + \omega_{el}\frac{d}{d\theta_r}(\Psi_{rb}) \tag{2.9}$$

$$v_{sc} = R_s i_{sc} + \frac{d}{dt}(L_s i_{sc}) + \omega_{el}\frac{d}{d\theta_r}(\Psi_{rc}) \tag{2.10}$$

where ω_{el} represents the electrical angular velocity of the rotor.

According to Faradays law of induction, the last component in (2.8) - (2.10) is the back-emf induced by the rotor and can be expressed as (2.11) - (2.13).

$$e_a = \omega_{el}\frac{d}{d\theta_r}(\Psi_{ra}) \tag{2.11}$$

$$e_b = \omega_{el}\frac{d}{d\theta_r}(\Psi_{rb}) \tag{2.12}$$

$$e_c = \omega_{el}\frac{d}{d\theta_r}(\Psi_{rc}) \tag{2.13}$$

In order to get the electrical angular velocity, ω_{el} and electrical angle, θ_{el} one can multiply the rotor angular velocity, ω_r and rotor angle, θ_r with the number of pole pairs, p which yields (2.14) - (2.15).

$$\omega_{el} = p\omega_r \tag{2.14}$$

$$\theta_{el} = p\theta_r \tag{2.15}$$

Representing the three-phase alternating current as a two-phase direct current is preferred due to its ability to simplify calculations and implementation. This approach offers two main advantages. Firstly, by converting the three-phase AC system into a two-phase DC system, the complexity of mathematical calculations is reduced. Two-phase systems involve fewer variables and equations compared to their three-phase counterparts. Secondly, this simplification makes it easier to implement. The general transformation principal from three-phase AC to two-phase DC, can be seen in Figure 2.3.

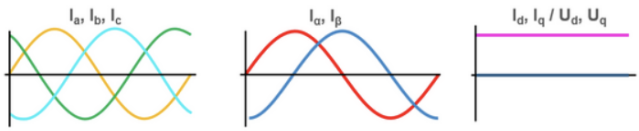


Figure 2.3: *Three-phase AC to two-phase DC [3].*

As depicted in Figure 2.3 the transformation from the *abc*-frame to *dq*-frame is done via the $\alpha\beta$-frame by utilising the Clarke transform. The Clarke transform is employed to convert the three stator phases, typically denoted as *abc*, into a two-coordinate time-variant stator phase represented by α and β. These new components exhibit sinusoidal variations with the same amplitude but with a phase shift compared to the original three-phase representation.

The α-component represents the average value of the three-phases, while the β-component represents the sum of the second and third phase divided by $\sqrt{3}$. Together, these two components provide a representation of the original three-phase system in a two-coordinate system. Figure 2.4 depicts the relationship between the two reference frames, the original three-phase currents i_{abc} and the transformed two-coordinate vector i_α, i_β [1].

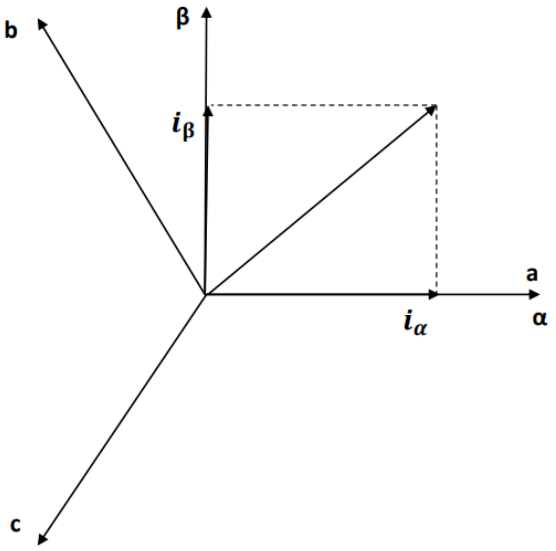


Figure 2.4: *The relationship between the $\alpha\beta$- frame and abc-frame.*

The Clarke transform is visualised in (2.16).

$$\begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} = \frac{2}{3} \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} i_{sa} \\ i_{sb} \\ i_{sc} \end{bmatrix} \tag{2.16}$$

After the Clarke transform, the two-axis orthogonal stationary $\alpha\beta$-frame is converted into a two-coordinate time-invariant rotor frame, meaning it rotates with the rotor. This is achieved by using the Park transform. The Park transform results in a vector that contains two components, namely the direct (d) and quadrature (q) component. The d-axis is responsible for generating flux through the field winding. On the other hand, the q-axis is where torque is produced [12]. These components are fixed to the rotating rotor of the PMSM.

The d-axis is aligned with the rotor and represents the component of the transformed vector. It corresponds to the magnetic flux aligned with the magnetic field of the rotor. The q-axis is positioned at a 90-degree angle from the d-axis. It represents the component of the transformed vector, that corresponds to the magnetic flux which is orthogonal to the rotor magnetic field. Figure 2.5 illustrates the relationship between the two-coordinate $\alpha\beta$-frame and the transformed dq-frame [2].

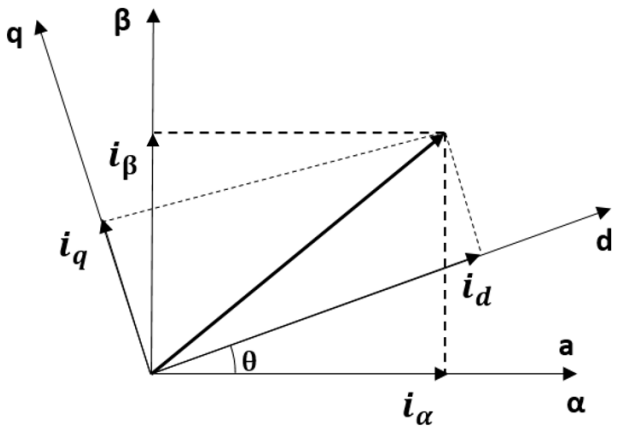


Figure 2.5: *The relationship between the stator fixed $\alpha\beta$- and the rotor fixed dq-reference frame.*

The Park transform is presented in (2.17).

$$\begin{bmatrix} i_d \\ i_q \end{bmatrix} = \begin{bmatrix} cos(\theta_r) & sin(\theta_r) \\ -sin(\theta_r) & cos(\theta_r) \end{bmatrix} \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} \tag{2.17}$$

when (2.8) - (2.10) are transformed using Clarke and Park respectively it yields (2.18) - (2.19).

$$v_d = \frac{d}{dt} i_d L_s + R_s i_d - i_q \omega_{el} L_s \tag{2.18}$$

$$v_q = \frac{d}{dt} i_q L_s + R_s i_q + i_d \omega_{el} L_s + \Psi_r \omega_{el} \tag{2.19}$$

which can be rewritten as (2.20) - (2.21).

$$\frac{d}{dt} i_d = -\frac{R_s}{L_s} i_d + i_q \omega_{el} + \frac{v_d}{L_s} \tag{2.20}$$

$$\frac{d}{dt} i_q = -i_d \omega_{el} - \frac{R_s}{L_s} i_q + \frac{v_q}{L_s} - \frac{\Psi_r \omega_{el}}{L_s} \tag{2.21}$$

Equation (2.20) and (2.21) can be written in state-space form with the state vector $x = [i_d \ i_q]^T$ and the input vector $u = [v_d \ v_q \ \omega_{el}]^T$ as seen in (2.22) - (2.23).

$$\dot{x} = f(x, u) \tag{2.22}$$

$$y = h(x) \tag{2.23}$$

where

$$f(x,u) = \begin{pmatrix} -\frac{R_s}{L_s}x_1 + x_2 u_3 + \frac{u_1}{L_s} \\ -x_1 u_3 + -\frac{R_s}{L_s}x_2 + \frac{u_2}{L_s} - \frac{\Psi_r u_3}{L_s} \end{pmatrix} \tag{2.24}$$

$$h(x) = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \tag{2.25}$$

The electromagnetic equation describing the motor torque, T_{em} in the dq-frame is shown in (2.26).

$$T_{em} = \frac{3}{2} p \Psi_r i_q \tag{2.26}$$

The mechanical equation describing the mechanical rotor angular acceleration is visualised in (2.27).

$$\frac{d}{dt}\omega_r = \frac{T_{em} - f\omega_r - T_L}{J} \tag{2.27}$$

where J is the rotor inertia, f is the viscous damping constant and T_L is the applied load torque [4].

2.2 Field Oriented Control

Atlas Copco utilises a motor with a high angular velocity and low torque. This motor configuration is well-suited for FOC since it aims to maximise the generated torque at the expense of a slight reduction in angular velocity. Figure 2.5 illustrates the relationship, revealing that in order to optimize the current for torque production, it is desirable for the d-current to be minimized, ideally approaching zero. The FOC's working principle is illustrated in Figure 2.6. The principle of this control strategy is to decouple the motor flux and torque to control them separately.

In order to implement the FOC, three requirements need to be fulfilled:

1. The angle between the d- and q-axis needs to be 90 degrees.
2. Independent control of field flux and torque producing current.
3. Instantaneous control of torque producing current.

Requirement one is fulfilled when the Clarke- and Park transform has been applied to the three phase currents, meaning we go from the abc-frame to the dq-frame via the $\alpha\beta$-frame. Requirement two is inherently met since the field flux is generated by the permanent magnets in the rotor and the torque producing current is generated by the stator windings. Instantaneous control of the torque producing current can be achieved via the synchronous reference frame current regulator and the inverter [12].

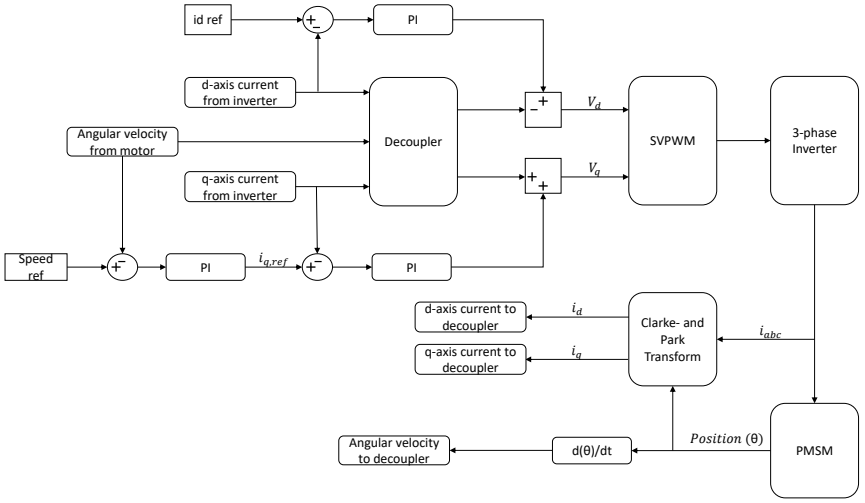


Figure 2.6: *Field oriented control scheme.*

The FOC constitutes of an inner and outer control loop. The inner loop begins with the position of the rotor, received from the encoder built in to the motor. The position together with the currents from the inverter are transformed via the Clarke- and Park transform respectively. This yields the representative currents in the dq-frame. The q-current is compared to the reference q-current and the error is fed to a PI-controller which outputs the q-voltage. The space vector pulse width modulation produces PWM-signals which is sent to the inverter. The inverter sends three phase signals to the PMSM. The outer loop derives the rotor position in order to get the rotor speed. This is compared to the reference speed and the error is fed to a PI-controller which outputs the q-current reference [15].

2.3 Signal Filtering

To handle potential noisy signals resulting from the data extraction process, employing a signal filtering technique is beneficial. Filters serve the purpose of suppressing interfering signals, reducing background noise, and selectively removing specific frequencies while allowing others to pass through. The filter aims at modifying the amplitude and/or phase characteristics of a signal in relation to its frequency, without introducing new frequencies or altering existing components. Instead, they adjust the relative amplitudes and phase relationships of different frequency components.

One such filter is the Butterworth filter. It is one of the types of filtering techniques used at Atlas Copco. The Butterworth filter is a type of signal processing filter designed to achieve a flat frequency response within the passband, zero roll-off response in the stopband and avoid ripples, see Figure 2.7. However, the

Butterworth filter has some drawbacks, including a wide transition band when transitioning from the passband to the stopband [13].

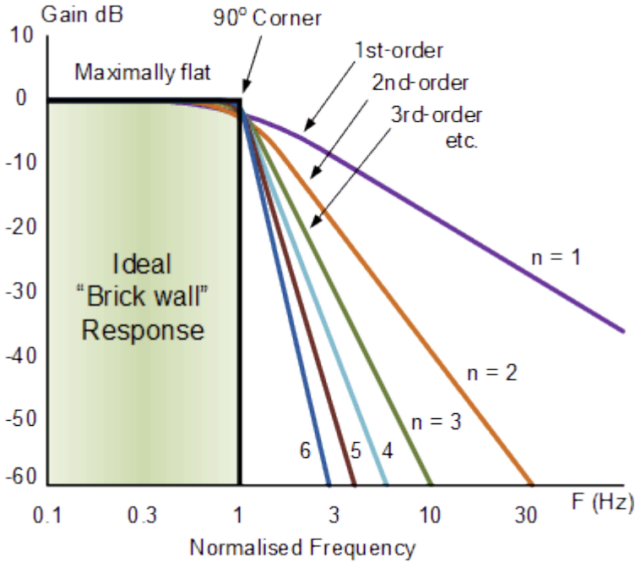


Figure 2.7: *Butterworth filter response [13].*

2.4 Extended Kalman Filter

The EKF is an extension of the Kalman filter developed in the early 1960's. Extension here refers to the filters ability to handle non-linearities [6].

2.4.1 Mathematical Model

The EKF utilises a nonlinear model according to (2.28) - (2.29).

$$x_k = f_k(x_{k-1}, u_k) + w_k \tag{2.28}$$

$$y_k = h_k(x_k) + v_k \tag{2.29}$$

$$E[w_k, w_k^T] = Q_k \tag{2.30}$$

$$E[v_k, v_k^T] = R_k \tag{2.31}$$

where x_k is the state vector, u_k is the input vector and f_k is the state transition vector. y_k is the measurement vector and h_k is the observation vector. w_k and v_k are white Gaussian noise with zero mean, where Q_k is the covariance matrices for the process noise and R_k is the covariance matrices for the measurement noise.

The working principle of the EKF algorithm can be summarised in two steps:

1. **Prediction Step**
   During the prediction phase, the current estimate is generated by utilising the state estimate from the previous timestep and the input from the current step. This estimation, known as the *priori* state estimate, predicts the current state. The current covariance matrix is calculated by using the linearised state transition vector and the covariance from the previous time step, as well as adding the process noise.

2. **Correction Step**
   In the correction phase, the state estimate is corrected by adding together the current predicted estimate and the weighted measurement adjustment. The adjustment is calculated by comparing the predicted measurements with the actual measurements. In addition to updating the state estimate, the corrected covariance matrix is also calculated. This involves subtracting the weighted version of the current predicted covariance from the non-weighted current predicted covariance.

Algorithm 1 showcases a full cycle of the EKF.

Algorithm 1 EKF

1: Initialization:
2: $\quad \hat{\boldsymbol{x}}_{\boldsymbol{0}} = \mathbb{E}[\boldsymbol{x}_{\boldsymbol{0}}]$ and $\boldsymbol{P}_{\boldsymbol{0}} = \mathbb{E}[(\boldsymbol{x}_{\boldsymbol{0}} - \hat{\boldsymbol{x}}_{\boldsymbol{0}})(\boldsymbol{x}_{\boldsymbol{0}} - \hat{\boldsymbol{x}}_{\boldsymbol{0}})^T]$
3: Prediction Step:
4: $\quad \hat{\boldsymbol{x}}_{\boldsymbol{k|k-1}} = \boldsymbol{f_k}(\hat{\boldsymbol{x}}_{\boldsymbol{k-1|k-1}}, \boldsymbol{u_{k|k}})$
5: $\quad \boldsymbol{P_{k|k-1}} = \boldsymbol{F_k}(\boldsymbol{P_{k-1|k-1}})\boldsymbol{F_k^T} + \boldsymbol{Q_k}$
6: Kalman Gain:
7: $\quad \boldsymbol{K_k} = \boldsymbol{P_{k|k-1}}\boldsymbol{H_k^T}(\boldsymbol{H_k}\boldsymbol{P_{k|k-1}}\boldsymbol{H_k^T} + \boldsymbol{R_k})^{-1}$
8: Correction Step:
9: $\quad \hat{\boldsymbol{x}}_{\boldsymbol{k|k}} = \hat{\boldsymbol{x}}_{\boldsymbol{k|k-1}} + \boldsymbol{K_k}(\boldsymbol{y_k} - \boldsymbol{h_k}(\hat{\boldsymbol{x}}_{\boldsymbol{k|k-1}}))$
10: $\quad \boldsymbol{P_{k|k}} = (\boldsymbol{I} - \boldsymbol{K_k}\boldsymbol{H_k})\boldsymbol{P_{k|k-1}}$

The first step in Algorithm 1 is to initialize the filter. This is done by an initial guess regarding the start value of each state. In this step the uncertainty of this initial guess is also included. The vector $\hat{x}_0$ consists of mean values and the covariance P_0 is derived by subtracting the mean from the initial state and then multiplying it with its transposed version. The EKF is an iterative estimator meaning that it employs state estimation based on previous estimations and measurements. Since there are no previous estimates or measurements in the first iteration of the algorithm this initialisation step is necessary.

Once the initialization step is completed, the prediction step follows. As indicated in the fourth row of Algorithm 1, the state prediction step takes the previous state estimate $\hat{x}_{k-1|k-1}$ along with the input vector. These values are then passed through the nonlinear state transition vector, which is discretised using

the forward Euler method. The discretisation process can be observed in equation (2.32).

$$\hat{x}_{k|k-1} = \hat{x}_{k-1|k-1} + T_s f_k \tag{2.32}$$

where T_s is the discrete time step between t_k and t_{k-1}.

In order to retrieve the predicted covariance $P_{k|k-1}$, seen in row five of Algorithm 1, the state transition vector is linearised using (2.33). The linearised state transition vector which yields the state transition matrix F_k, is multiplied with the previous covariance $P_{k-1|k-1}$ and the process noise covariance is added.

$$F_k = \left.\frac{\partial \hat{x}}{\partial x}\right|_{\hat{x}_{k-1|k-1}, u_k} \tag{2.33}$$

When the prediction step is done, the Kalman gain K_k is calculated. It utilises the linearised observation matrix H_k, seen in (2.34) and the predicted covariance. A measurement noise covariance is also added. The Kalman gain works as a weight between previous and current step.

$$H_k = \left.\frac{\partial h}{\partial x}\right|_{\hat{x}_{k|k-1}} \tag{2.34}$$

During the correction step, the state estimation $\hat{x}_{k|k}$ is revised by incorporating the predicted estimate along with the Kalman gain and the adjusted measurements. Simultaneously, the covariance matrix $P_{k|k}$ is updated using the predicted covariance, the Kalman gain, and the linearised observation matrix [18].

2.4.2 Adaptive Extended Kalman Filter

The AEKF is a variation of the EKF that offers the capability to automatically adapt the values of the process noise covariance and measurement noise covariance. This variant follows the overall algorithm of the EKF, with an additional step for calculating the noise covariance matrices.

The calculation of Q_k can be observed in (2.35) and R_k in (2.36).

$$\begin{aligned}
\alpha_1 &= \frac{N_Q - 1}{N_Q} \\
\hat{\omega}_k &= \hat{x}_k - \hat{x}_{k|k-1} \\
\bar{\omega}_k &= \alpha_1 \bar{\omega}_{k-1} + \frac{1}{N_Q}\hat{\omega}_k \\
\Delta Q_k &= \frac{1}{N_Q - 1}(\hat{\omega}_k - \bar{\omega}_k)(\hat{\omega}_k - \bar{\omega}_k)^T + \frac{1}{N_Q}\left((P_{k|k-1}) - (F_k P_{k-1|k-1} {F_k}^T)\right) \\
Q_k &= |\mathrm{diag}(\alpha_1 Q_{k-1} + \Delta Q_k)|
\end{aligned} \tag{2.35}$$

$$
\begin{aligned}
\alpha_2 &= \frac{N_R - 1}{N_R} \\
\bar{e}_k &= \alpha_2 \bar{e}_{k-1} + \frac{1}{N_R} e_k \\
\Delta R_k &= \frac{1}{N_R - 1}(e_k - \bar{e}_k)(e_k - \bar{e}_k)^T - \frac{1}{N_R} H_k P_{k|k-1} H_k^T \\
R_k &= |\text{diag}(\alpha_2 R_{k-1} + \Delta R_k)|
\end{aligned}
\tag{2.36}
$$

where N_R and N_Q are tuning variables and $e_k = y_k - H\hat{x}_{k|k-1}$.

The AEKF utilises (2.35) - (2.36) to iteratively update its estimates and adaptively account for uncertainties in the process and measurement models. These equations allow the filter to continuously incorporate the differences between predicted and measured values, as well as weighted averages, enabling it to dynamically adjust the covariance matrices Q and R. This adaptive adjustment helps the filter to reflect the actual uncertainties, hopefully leading to improved accuracy in the estimation process [8].

2.5 Unscented Kalman Filter

The UKF, presented in [11], alternates the algorithm compared to the EKF by utilising the unscented transform (UT). The UT allows for use of the state transition vector without linearisation by propagating the sigma points through the state transition vector [21], as shown in Figure 2.8.

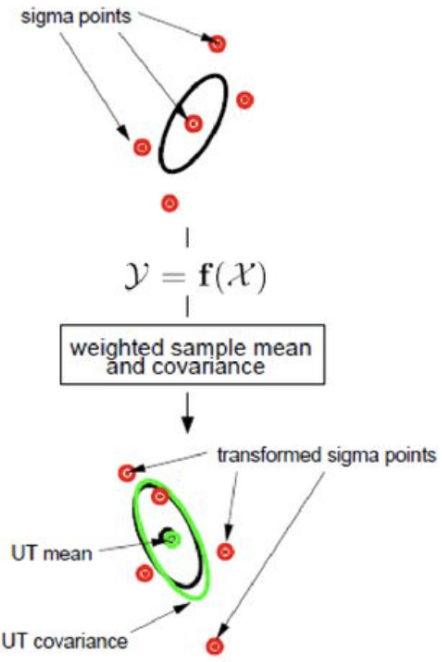


Figure 2.8: *Example of the UT and how it affect the sigma points, mean and covariance [17].*

A complete cycle of the UKF can be seen in Algorithm 2.

Algorithm 2 UKF

Initialization:

$\hat{\boldsymbol{x}}_{\boldsymbol{0}} = \mathbb{E}[\hat{\boldsymbol{x}}_{\boldsymbol{0}}]$ and $\boldsymbol{P_0} = \mathbb{E}[(\boldsymbol{x_0} - \hat{\boldsymbol{x}}_{\boldsymbol{0}})(\boldsymbol{x_0} - \hat{\boldsymbol{x}}_{\boldsymbol{0}})^T]$

Sigma Points:

$\boldsymbol{\mathcal{X}_{k-1}} = [\hat{\boldsymbol{x}}_{\boldsymbol{k-1}} \quad \hat{\boldsymbol{x}}_{\boldsymbol{k-1}} + \eta\sqrt{\boldsymbol{P_{k-1}}} \quad \hat{\boldsymbol{x}}_{\boldsymbol{k-1}} - \eta\sqrt{\boldsymbol{P_{k-1}}}\,]$

Prediction Step:

$\boldsymbol{\mathcal{X}_{k|k-1}} = \boldsymbol{f_k}[\boldsymbol{\mathcal{X}_{k-1}}, \boldsymbol{u_{k-1}}]$

$\hat{\boldsymbol{x}}_{\boldsymbol{k}}^{\boldsymbol{-}} = \sum_{i=0}^{2L} \boldsymbol{W_i^{(m)}} \boldsymbol{\mathcal{X}_{i,k|k-1}}$

$\hat{\boldsymbol{P}}_{\boldsymbol{k}}^{\boldsymbol{-}} = \sum_{i=0}^{2L} \boldsymbol{W_i^{(c)}} [\boldsymbol{\mathcal{X}_{i,k|k-1}} - \hat{\boldsymbol{x}}_{\boldsymbol{k}}^{\boldsymbol{-}}][\boldsymbol{\mathcal{X}_{i,k|k-1}} - \hat{\boldsymbol{x}}_{\boldsymbol{k}}^{\boldsymbol{-}}]^T + \boldsymbol{Q_k}$

$\boldsymbol{\mathcal{Y}_{k|k-1}} = \boldsymbol{h_k}[\boldsymbol{\mathcal{X}_{k|k-1}}]$

$\hat{\boldsymbol{y}}_{\boldsymbol{k}}^{\boldsymbol{-}} = \sum_{i=0}^{2L} \boldsymbol{W_i^{(m)}} \boldsymbol{\mathcal{Y}_{i,k|k-1}}$

Correction Step:

$\boldsymbol{P_{\tilde{y}_k \tilde{y}_k}} = \sum_{i=0}^{2L} \boldsymbol{W_i^{(c)}} [\boldsymbol{\mathcal{Y}_{i,k|k-1}} - \hat{\boldsymbol{y}}_{\boldsymbol{k}}^{\boldsymbol{-}}][\boldsymbol{\mathcal{Y}_{i,k|k-1}} - \hat{\boldsymbol{y}}_{\boldsymbol{k}}^{\boldsymbol{-}}]^T + \boldsymbol{R_k}$

$\boldsymbol{P_{x_k y_k}} = \sum_{i=0}^{2L} \boldsymbol{W_i^{(c)}} [\boldsymbol{\mathcal{X}_{i,k|k-1}} - \hat{\boldsymbol{x}}_{\boldsymbol{k}}^{\boldsymbol{-}}][\boldsymbol{\mathcal{Y}_{i,k|k-1}} - \hat{\boldsymbol{y}}_{\boldsymbol{k}}^{\boldsymbol{-}}]^T$

$\boldsymbol{K_k} = \boldsymbol{P_{x_k y_k}} \boldsymbol{P_{\tilde{y}_k \tilde{y}_k}^{-1}}$

$\hat{\boldsymbol{x}}_{\boldsymbol{k}} = \hat{\boldsymbol{x}}_{\boldsymbol{k}}^{\boldsymbol{-}} + \boldsymbol{K_k}(\boldsymbol{y_k} - \hat{\boldsymbol{y}}_{\boldsymbol{k}}^{\boldsymbol{-}})$

$\hat{\boldsymbol{P}}_{\boldsymbol{k}} = \hat{\boldsymbol{P}}_{\boldsymbol{k}}^{\boldsymbol{-}} + \boldsymbol{K_k} \boldsymbol{P_{\tilde{y}_k \tilde{y}_k}} \boldsymbol{K_k^T}$

The first step in Algorithm 2 is to initialise the algorithm with an initial guess regarding the states and the uncertainty of the initial state guess. $\hat{x}_0$ is a vector consisting of mean values, and the covariance P_0 is calculated by subtracting the mean from the initial guess and multiplying it with its transposed version. After that the sigma points, $\mathcal{X}_{k-1}$ are calculated. There are $2L + 1$ (L is the number of states) number of sigma points. The first sigma point is called the zero point and is the mean of the previous estimate. The other $2L$ sigma points are calculated according to (2.38) - (2.39) where the first L sigma points utilizes the zero point and scales them by a positive factor and the Cholesky decomposition of the covariance (2.38). In the same way, the last L sigma points also utilises the zero points and scales them by subtracting a negative factor and the Cholesky decomposition of the covariance (2.39). This gives the sigma points a spread around the zero point.

$$\hat{x}_{k-1}^{(0)} \tag{2.37}$$

$$\hat{x}_{k-1}^{(L)} + \eta\sqrt{P_{k-1}} \tag{2.38}$$

$$\hat{x}_{k-1}^{(2L-L)} - \eta\sqrt{P_{k-1}} \tag{2.39}$$

where η is the scaling parameter [21] and can be seen in (2.49).

Cholesky Decomposition

The Cholesky Decomposition calculates the square root of a matrix. To define the decomposition, we consider a symmetric matrix A of size $K \times K$. A has a Cholesky decomposition if and only if it can be expressed as a lower triangular matrix L of the same size, where the diagonal entries are strictly positive real numbers and it satisfies (2.40).

$$A = LL^T \tag{2.40}$$

For a Cholesky decomposition to exist, the matrix A must also be positive definite. Assuming all the conditions mentioned earlier are satisfied, the Cholesky decomposition can be computed by solving (2.40). Specifically, one can use matrix multiplication to derive (2.41).

$$A_{tv} = \sum_{u=1}^{K} L_{tu} L_{uv}^T = \sum_{u=1}^{K} L_{tu} \overline{L_{vu}} \tag{2.41}$$

In A_{tv}, each entry is located on the t:th row and v:th column, where both t and v range from 1 to K. Since L is lower triangular, when $u > v$, $L_{vu} = 0$, resulting in (2.42).

$$A_{tv} = L_{tv} \overline{L_{vv}} + \sum_{u<v} L_{tu} \overline{L_{vu}} \tag{2.42}$$

By utilising the rules stated below the entries to L can be derived.

- Solve one column at a time, starting with $v = 1$ and then progress from 2 up to K.
- For each column, compute the diagonal entry L_{vv} and the diagonal entries L_{tv} but only for $t > v$ since $L_{tv} = 0$ when $v < t$.

The diagonal entries are given by solving (2.41) yielding (2.43), always choosing the positive root.

$$L_{vv} = \sqrt{A_{vv} - \sum_{u<v} L_{vu} \overline{L_{vu}}} \tag{2.43}$$

The entries that are not in the diagonal are given in (2.44) [19].

$$L_{tv} = \frac{1}{L_{vv}} \left(A_{tv} \sum_{u<v} L_{tu} \overline{L_{vu}} \right) \tag{2.44}$$

After the sigma points are calculated, the prediction step of the UKF is performed. As seen in row six of Algorithm 2, the UT is used to propagate the sigma points and inputs u_{k-1} through the state transition vector f_k. This is the same state transition vector that is used in the EKF.

The weights, W_i used throughout the UKF algorithm can be calculated according to (2.45) - (2.47).

$$W_0^{(m)} = \frac{\lambda}{\lambda + L} \tag{2.45}$$

$$W_0^{(c)} = \frac{\lambda}{\lambda + L} + (1 - \alpha^2 + \beta) \tag{2.46}$$

$$W_i^{(m)} = w_i^{(c)} = \frac{1}{2(\lambda + L)} \tag{2.47}$$

where α and β are scaling factors, and

$$\lambda = L(\alpha^2 - 1) \tag{2.48}$$

$$\eta = \sqrt{\lambda + L} \tag{2.49}$$

The updated sigma points are used together with the weights to obtain the state estimate mean, $\hat{x}_k^-$ in row seven in the Algorithm 2. The covariance, $\hat{P}_k^-$ is also calculated by subtracting the state estimate mean from the propagated sigma points and weighted. The process noise covariance Q_k is also added as seen in row eight in the Algorithm 2. The final part in the prediction step is to propagate the sigma points obtained in row six through the observation vector, h_k and weigh them in a similar way as for the states.

When the prediction step is complete the correction step is performed. The first thing that is performed is the calculation of the covariance, $P_{\tilde{y}_k \tilde{y}_k}$ for the measurement and the cross covariance $P_{x_k y_k}$. They are calculated by subtracting the measurement mean from the propagated sigma points $\mathcal{Y}_{k|k-1}$ seen in row nine. The measurement noise covariance R_k is also added to the covariance for the measurement. Both of these are used to calculate the Kalman gain, K_k. Finally the updated state estimation $\hat{x}_k$ is obtained by using the predicted state mean and adding a weighted measurement adjustment. The adjustment is determined by comparing the predicted measurement mean against the actual measurements. The covariance is obtained by adding the predicted covariance from the prediction step with the covariance from the measurement mean in the correction step, as well as weighing it with the Kalman gain [21].

2.6 Power Losses in a PMSM

Both the winding resistance and the rotor flux linkage was assumed to have a linear relationship between the temperate increase and the variation of the state [10]. In this thesis, only losses related to the winding resistance and the rotor flux linkage are considered. The loss in the winding resistance can be modeled according to (2.50).

$$P_{Rs} = \frac{3}{2} \hat{I}^2 Rs \tag{2.50}$$

where P_{Rs} is the power generated when current is flowing through the stator winding. $\hat{I}$ is the peak-current for the q and d-axis, but since the d-axis current is mostly zero it is not considered. R_s is the stator winding resistance. The loss related to the rotor flux linkage can be model according to (2.51).

$$P_{\Psi_r} = \int_V \sigma E^2 \, dV = \int_V \frac{J^2}{\sigma} \, dV \tag{2.51}$$

where P_{Ψ_r} is the power generated due to the induced Eddy currents. Eddy currents are loops of electrical current induced within conductors by a changing magnetic field [14], σ is the material conductivity, E is the electric field, J is the Eddy current density and V is the volume of the material [9].

2.7 Parameter Temperature Dependency

The temperature relationship was assumed to be linear according to [23]. If we denote both the estimates of resistance and flux as x, the linear relationship can be seen in (2.52).

$$T = T_0 + \frac{x - x_0}{\alpha x_0} \tag{2.52}$$

where x_0 is the nominal value of both resistance and flux. T and T_0 is the temperature and starting temperature respectively, and α is the temperature coefficient for the respective materials, in this case copper and neodymium.

3

Implementation

In this chapter, Simulink was utilised to develop several models, including the PMSM, FOC and thermal model. Two different state space models were created. The first model employed separate state transition vectors, one for each parameter of interest, while the second model incorporated combined state transition vectors. Subsequently, the latter model was chosen for implementation in each Kalman filter employed in the thesis.

3.1 PMSM Model

The PMSM model was implemented in Simulink, and a visual overview of the model is provided in Figure 3.1. The motor operates based on voltage inputs in the *abc*-frame. The motor generates current outputs in the *abc*-frame, which are then transformed to the dq-frame along with the electrical rotor position θ_{el} using the Clarke and Park transforms, as described in (2.16) and (2.17). The final two outputs of the model are the motor torque T_{em} and the rotor angular velocity ω_m.

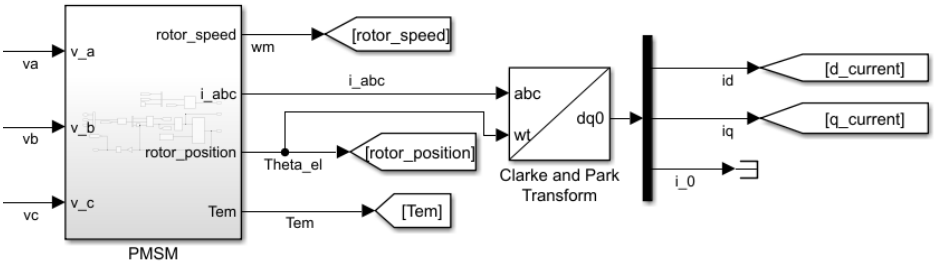


Figure 3.1: *Overview of the PMSM model in Simulink.*

As outlined in Section 2.1.3, the PMSM's internal systems, namely the electrical, electromagnetic, and mechanical systems, are characterised by different sets of equations. Each of these systems has been implemented as separate blocks. An overview of how these systems were integrated within the simulation environment is presented in Figure 3.2. Furthermore, Figure 3.3 - 3.5 provide individual illustrations of the three systems mentioned.

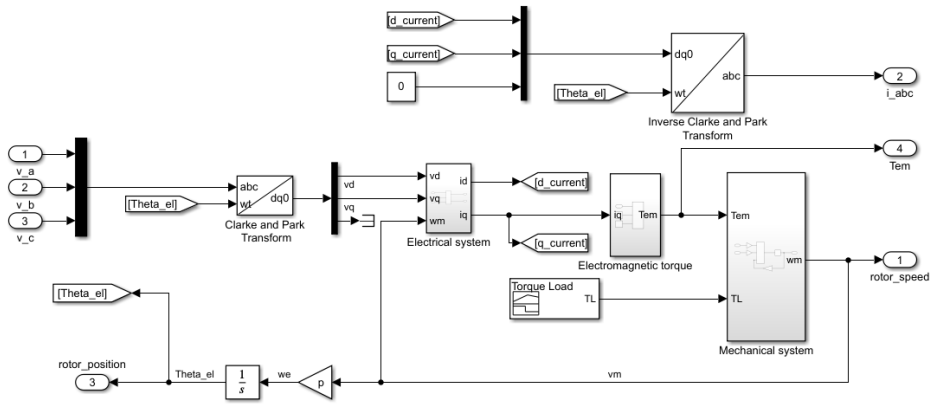


Figure 3.2: *An overview of the components constituting the PMSM in Simulink.*

The block *Electrical System* in Figure 3.2 was derived using (2.20) and (2.21) yielding the d and q-axis currents as shown in Figure 3.3. This block uses the transformed dq-voltages and the rotor angular velocity, ω_m as inputs.

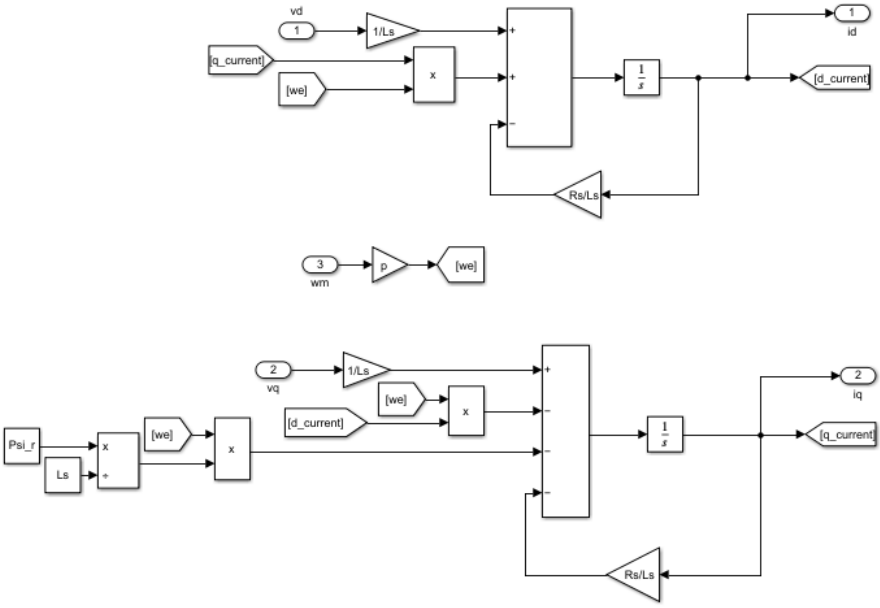


Figure 3.3: *Electrical system implemented in Simulink.*

In Figure 3.4, the block *Electromagnetic System*, which was based on (2.26) is depicted. The input is the q-axis current from the *Electrical System*. The output from the block is the produced torque T_{em}.

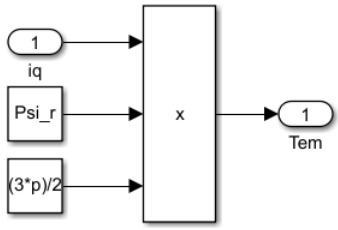


Figure 3.4: *Electromagnetic system implemented in Simulink.*

The produced torque from the *Electromagnetic System* is then used as input to the *Mechanical system* which represented (2.27) in the simulation environment and can be seen in Figure 3.5. The applied load torque T_L is also an input to the system. This system yields the rotor angular velocity ω_m.

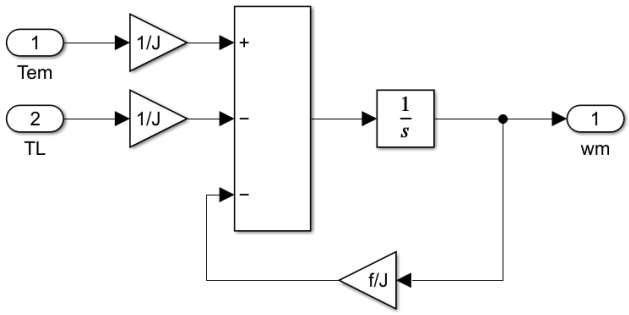


Figure 3.5: *Mechanical system of the PMSM implemented in Simulink.*

The parameters used in the PMSM are viewed in Table 3.1.

Table 3.1: *Parameter values for a specific PMSM, which were provided by Atlas Copco.*

Nominal Parameter	**Value [Unit]**
Winding resistance (R_s)	0.03774[Ω]
Rotor flux linkage (Ψ_r)	0.00831 [Wb]
Inductance (L_s)	0.03264e-3 [H]
Pole pairs (p)	1 [-]
Rotor inertia (J)	3.51e-6 [$kg \cdot m^2$]
Viscous damping (f)	3.45e-6 [$\frac{Ns}{m}$]

3.2 Field Oriented Control

The FOC was created in Simulink. An implementation of the controller is displayed in Figure 3.6. The controller uses the d and q-axis currents from the motor together with the rotor angular velocity ω_m as input. It also utilises a reference angular velocity as well as a reference d-axis current, set by the operator as input. The controller outputs voltages in the dq-frame.

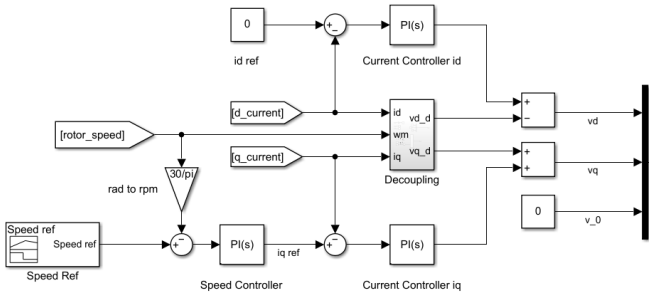


Figure 3.6: *FOC implemented in Simulink.*

The decoupling block, depicted in Figure 3.7, was utilized to mitigate the effects of coupling. The interdependencies can be observed in (2.18) - (2.19), where the q-axis current influences the d-axis voltage and the d-axis current impacts the q-axis voltage. The d and q-axis currents, in conjunction with the rotor angular velocity ω_m, serve as inputs to the decoupler, which yields decoupled d and q-axis voltages.

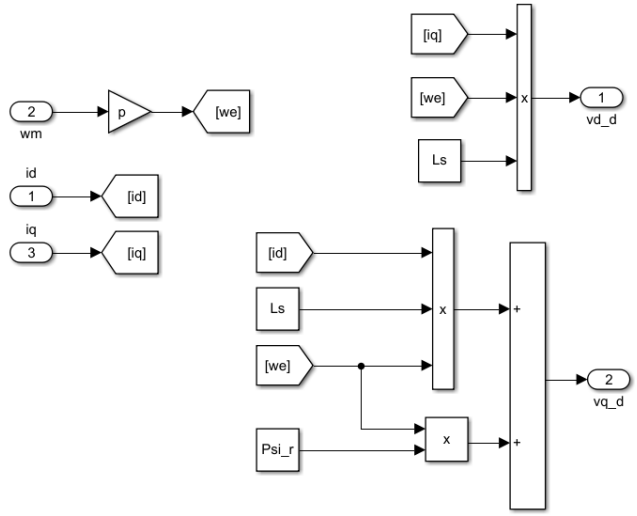


Figure 3.7: *Decoupling block implemented in Simulink.*

The PI-controllers were tuned according to (3.1) - (3.3), the method [7] utilises the motor parameters seen in Table 3.1.

$$w_m = \frac{R_s}{L_s} \tag{3.1}$$

$$w_c = 5 * w_m \tag{3.2}$$

$$w_s = \frac{w_c}{10} \tag{3.3}$$

The parameters used in the FOC can be seen in Table 3.2.

Table 3.2: *Tuning parameters for FOC.*

Parameter	Value
$K_{i_{Speed}}$	$10 * w_s^2 * J$
$K_{p_{Speed}}$	$10 * w_s * J$
$K_{i_{Current}}$	$w_c^2 * L_s$
$K_{p_{Current}}$	$w_c * L_s$

3.3 Thermal Model

In this thesis, the thermal model employed focused on the losses associated with the winding resistance and rotor flux linkage. The thermal model was created since the simulation model lacked internal dynamics that a real PMSM possesses. Its purpose was to simulate the impact of temperature rise on the motor by considering the corresponding variations in winding resistance and rotor flux linkage.

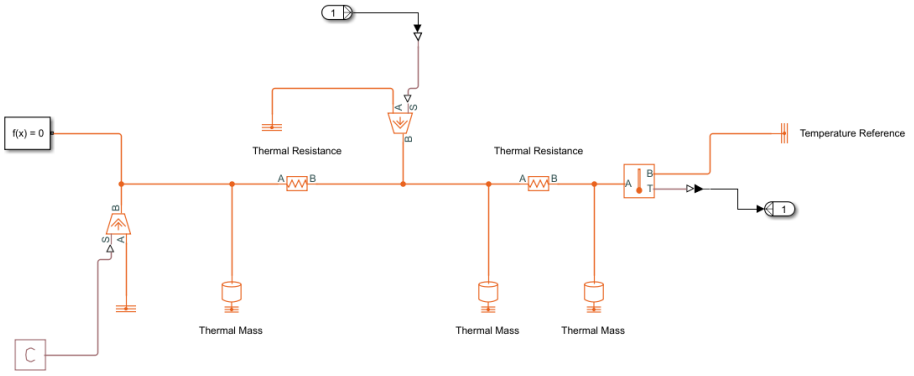


Figure 3.8: *Thermal model implemented in Simulink using the Simscape library.*

By observing Figure 3.8 and starting from the left, the effect loss created by the Eddy currents, seen in (2.51) was fed to the system. The thermal generation

to the surrounding area was generated via the first thermal mass and thermal resistance. After that the effect loss from the winding resistance, seen in (2.50) was provided. Its thermal generation was provided with the next thermal mass and thermal resistance. The final thermal mass symbolised the air surrounding the whole rotor and stator package.

3.4 Implementation of Kalman Filters

In this section, the implementation of the state space vectors for the estimation of the winding resistance R_s and rotor flux linkage Ψ_r are described for the different Kalman filters.

3.4.1 Separate State Transition Vectors

Initially, the concept of separate Kalman filters was implemented, wherein the winding resistance and rotor flux linkage were treated as distinct entities within two different Kalman filters. Subsequently, the estimation of the winding resistance and rotor flux was accomplished by implementing corresponding functions in Matlab. The initial implementation involved the utilization of two parallel Kalman filters, with the first filter dedicated to estimating the winding resistance and the second filter focused on estimating the rotor flux.

EKF

The implementation of the EKF followed Section 2.4.1 and specifically referred to Algorithm 1. In this implementation, the state vector was defined as $x = [i_d \ i_q \ R_s]^T$, and the input vector as $u = [v_d \ v_q \ \omega_{el}]^T$. To discretise the system, the forward Euler method was employed. As a result, the state transition vector utilised for the resistance estimation is shown in (3.4). The formulation of this function was based on (2.20) - (2.21), as well as adding winding resistance as a state.

$$f_{R_s}(x,u) = \begin{pmatrix} (1 - \frac{T_s x_3}{L_s})x_1 + T_s u_3 x_2 + \frac{T_s u_1}{L_s} \\ -T_s u_3 x_1 + (1 - \frac{x_3 T_s}{L_s})x_2 + \frac{T_s u_2}{L_s} - \frac{T_s \Psi_r u_3}{L_s} \\ x_3 \end{pmatrix} \tag{3.4}$$

The linearised form of the vector presented in (3.4) is depicted in (3.5).

$$F_{R_s} = \begin{pmatrix} 1 - \frac{T_s R_s}{L_s} & T_s \omega_{el} & -\frac{T_s i_d}{L_s} \\ -T_s \omega_{el} & 1 - \frac{T_s R_s}{L_s} & -\frac{T_s i_q}{L_s} \\ 0 & 0 & 1 \end{pmatrix} \tag{3.5}$$

The observation vector is illustrated in (3.6).

$$h_{R_s}(x) = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \tag{3.6}$$

The linearisation of the observation vector can be seen in (3.7).

$$H_{R_s} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \tag{3.7}$$

The second Kalman filter was implemented in the same way as the first. However, this filter employed a different state transition vector, specified by (3.8). The state vector was defined as $x = [i_d \ i_q \ \Psi_r]^T$, while the input vector consisted of $u = [v_d \ v_q \ \omega_{el}]^T$.

$$f_{\Psi_r}(x,u) = \begin{pmatrix} (1 - \frac{T_s R_s}{L_s})x_1 + T_s u_3 x_2 + \frac{T_s u_1}{L_s} \\ -T_s u_3 x_1 + (1 - \frac{T_s R_s}{L_s})x_2 + \frac{T_s u_2}{L_s} - \frac{T_s u_3 x_3}{L_s} \\ x_3 \end{pmatrix} \tag{3.8}$$

The linearised matrix of (3.8) is presented in (3.9).

$$F_{\Psi_r} = \begin{pmatrix} 1 - \frac{T_s R_s}{L_s} & T_s \omega_{el} & 0 \\ -T_s \omega_{el} & 1 - \frac{T_s R_s}{L_s} & -\frac{T_s \omega_{el}}{L_s} \\ 0 & 0 & 1 \end{pmatrix} \tag{3.9}$$

The observation vector (3.6) and the linearised observation matrix (3.7) from the first filter were still viable.

UKF

The implementation of the UKF was carried out in accordance with Section 2.5, following Algorithm 2. The UKF utilised identical state transition vectors, as described in (3.4) and (3.8). Additionally, the UKF employed the same observation vector for both filters, as presented in (3.6).

3.4.2 Combined State Transition Vectors

Once the individual Kalman filters were implemented and tested to assess their functionality, a combined version was tested. This combined version entailed fusing together the two the state transition vectors (3.4) and (3.8), into a unified vector that was used for simultaneously estimating both the winding resistance and rotor flux.

EKF

The implementation of the EKF followed Section 2.4.1 and adhered to Algorithm 1. In this implementation, a single filter was employed, which utilised a fused state transition vector. The equations defining the state transition function incorporated (2.20) and (2.21) together with the states R_s and Ψ_r. The state vector was defined as $x = [i_d \ i_q \ \Psi_r \ R_s]^T$, while the input vector consisted of $u = [v_d \ v_q \ \omega_{el}]^T$. Following the discretisation process using the forward Euler method, the resulting state transition vector can be observed in (3.10).

$$f_{R_s,\Psi_r}(x,u) = \begin{pmatrix} (1-\frac{T_s x_4}{L_S})x_1 + T_s u_3 x_2 + \frac{T_s u_1}{L_s} \\ -T_s u_3 x_1 + (1-\frac{T_s x_4}{L_S})x_2 + \frac{T_s u_2}{L_s} - \frac{T_s u_3 x_3}{L_s} \\ x_3 \\ x_4 \end{pmatrix} \tag{3.10}$$

The linearisation of (3.10) is featured in (3.11).

$$F_{R_s,\Psi_r} = \begin{pmatrix} 1-\frac{T_s R_s}{L_s} & T_s\omega_{el} & 0 & -\frac{T_s i_d}{L_s} \\ -T_s\omega_{el} & 1-\frac{T_s R_s}{L_s} & -\frac{T_s\omega_{el}}{L_s} & -\frac{T_s i_q}{L_s} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.11}$$

The observation vector is shown in (3.12).

$$h_{R_s,\Psi_r}(x) = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \tag{3.12}$$

The linearisation of (3.12) is displayed in (3.13).

$$H_{R_s,\Psi_r} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \tag{3.13}$$

AEKF

The AEKF employed an identical state transition vector (3.10) and observation vector (3.12) as the EKF. It followed the implementation outlined in Section 2.4.1, particularly Algorithm 1. However, Section 2.4.2 highlights the disparity in the computation of noise covariance matrices.

UKF

The implementation of the UKF adhered to Section 2.5 and followed Algorithm 2. It employed the identical state transition vector described in (3.10) and utilised the same observation vector as presented in (3.12).

4

Simulation

In this chapter, the process of data extraction from simulation is explained, along with a specific test case centered around a tightening operation. The obtained simulation data and the results of the Kalman filters are showcased to demonstrate the performance of the parameter estimation.

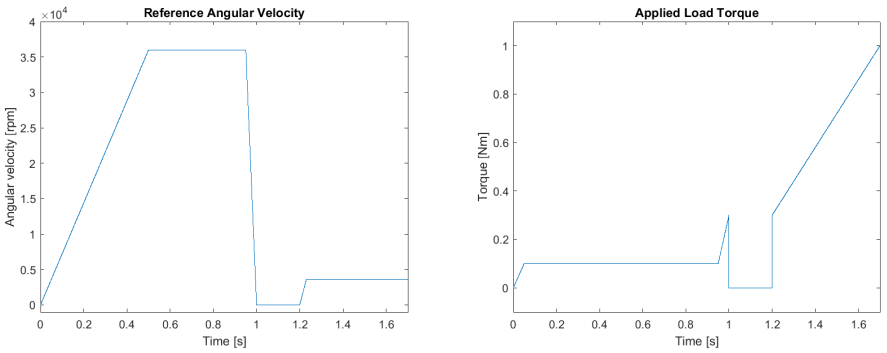
4.1 Simulation Setup

The sampling time, T_s used for simulation in Simulink was set to 1.25e-4 s. A complete simulation and estimation cycle were run according to Appendix A.0.1.

4.2 Test Case

The test case was constructed based on angular velocity and applied load torque. The outer loop in the FOC requires a reference angular velocity in order to control the voltages sent to the motor, as depicted in Figure 3.6. The applied load torque was used as a reference in order for the PMSM to produce the demanded torque. It was implemented according to Figure 3.2.

The standard tightening operation consisted of two distinct phases: the rundown phase and the tightening phase. During the rundown phase, the screw was threaded down until the screw head made contact with the material. Meanwhile the primary focus of the estimation was to estimate the flux. Once the rundown phase was completed, the tightening phase was initiated. In this phase, the screw was torqued according to the specifications provided by the customer. In this phase the emphasis of the estimation was on estimating the resistance.



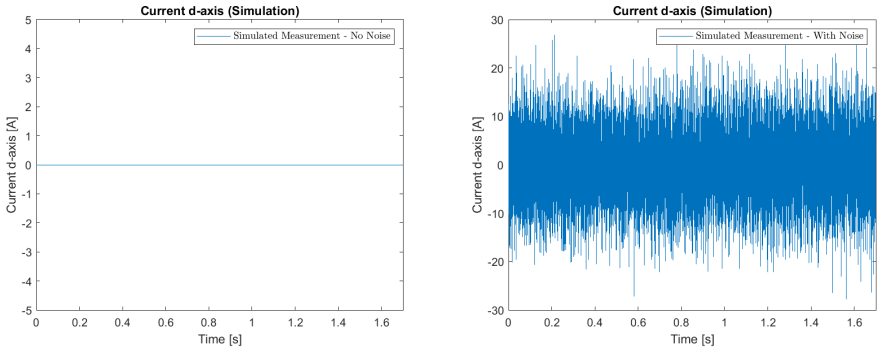
(a) *(b)*

Figure 4.1: *Reference angular velocity and reference load torque for the simulation test.*

As seen in Figure 4.1b, the reference torque ramped up from 0 to 0.1 during the initial 0.05 seconds, indicating the need for a small torque to thread down the screw. During the rundown period, the angular velocity reference was ramped up to its maximum limit and maintained, simulating a rundown procedure as shown in Figure 4.1a. At around 0.95 seconds, the angular velocity dropped significantly, signifying a switch from rundown to tightening. The tightening procedure was initiated by ramping up the torque to its target value. To simulate the rotation of the screw, a low angular velocity was used.

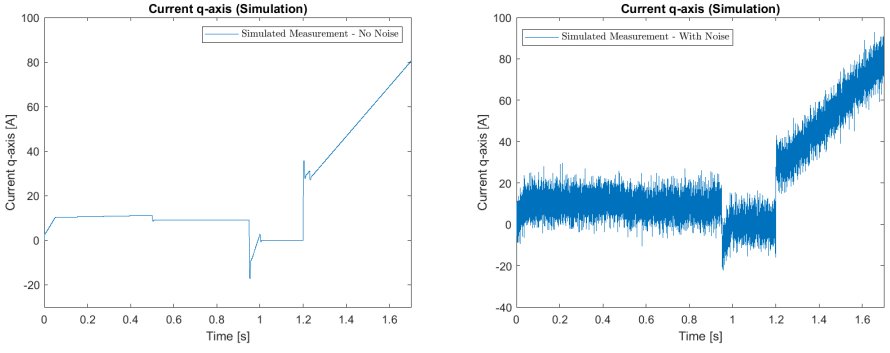
4.3 Simulated Raw Data

The simulated data utilised in this thesis incorporated noise free operations, and operations using noise levels that reflected those present in the actual motor. To determine the noise levels, measurements from a real motor were analysed, allowing for identification of the noise component. Figure 4.1 illustrates the test case, with the rundown phase spanning between $0 - 1$ seconds and the tightening phase spanning between $1.2 - 1.7$ seconds. The time interval between 1 and 1.2 seconds corresponds to the resting phase. The relevant data extracted from the measurable parameters as well as the input parameters is shown in Figures 4.2 - 4.6.



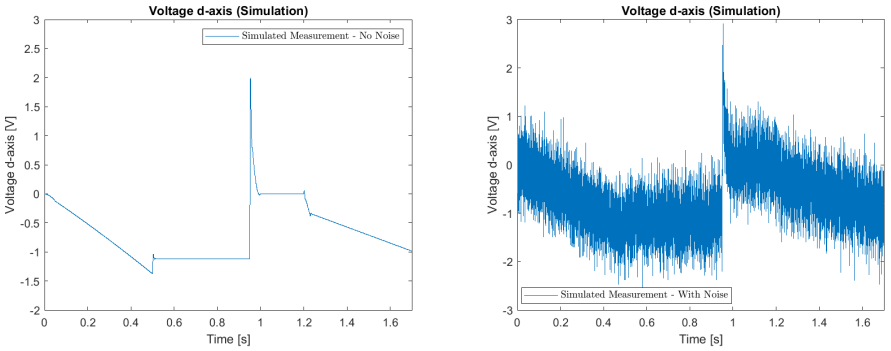
(a) *(b)*

Figure 4.2: *Simulated d-axis current with and without noise.*



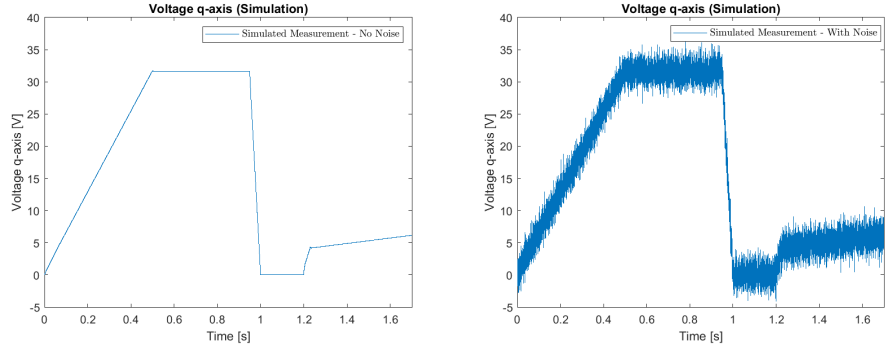
(a) *(b)*

Figure 4.3: *Simulated q-axis current with and without noise.*



(a) *(b)*

Figure 4.4: *Simulated d-axis voltage with and without noise.*



(a) *(b)*

Figure 4.5: *Simulated q-axis voltage with and without noise.*

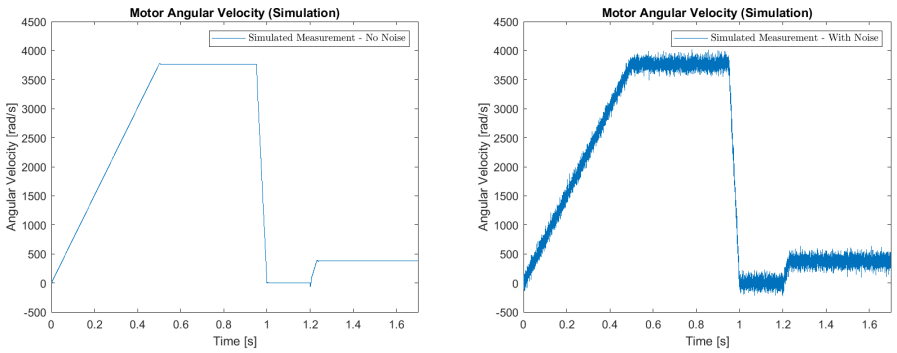


Figure 4.6: *Simulated ω_m with and without noise.*

4.4 Kalman Filter Evaluation

In this section, an evaluation of the Kalman filters was conducted using simulated data. The temperature dependent parameters are presented in plots. Note, since the results appeared very noisy a moving mean utilising a sliding window of 2000 data points was applied for the temperature dependent parameters. The moving mean was used to determine the parameter levels by finding the highest and lowest levels within the respective phase. The estimated temperature dependent parameters are summarised in Table 4.1. The evaluation encompassed two distinct cases, each aimed at examining the filters performance under different conditions. The first case involved applying the Kalman filters directly to unfiltered data without any tuning or adjustments. This approach allowed for an assessment of the filters performance in their default configuration. In the second case, the filters were utilised with filtered data that had undergone a preprocessing step to mitigate noise. Additionally, the Kalman filters themselves were tuned. This approach enabled an evaluation to determine whether there was an improvement in performance.

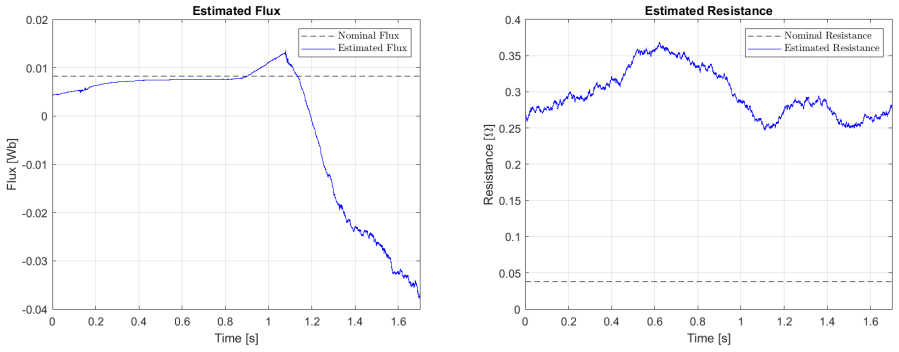
4.4.1 EKF - Default Configuration

In this case, the performance of the EKF was investigated under its default configuration. The default configuration comprised of the initialisation matrices seen in (4.1) and the noise covariance matrices in (4.2).

$$x_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad P_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.1}$$

$$Q = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad R = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{4.2}$$

The results from this case can be seen in Figure 4.7.



(a) *(b)*

Figure 4.7: *The EKF was used to estimate the Flux and Resistance without any data filtering or tuning.*

The flux estimation remained consistent in the rundown part, as shown in Figure 4.7a. It fluctuated between $0.0042 - 0.0111\ Wb$. The resistance remained relatively unchanged throughout the entire operation, as observed in Figure 4.7b. During the tightening phase the resistance fluctuated between $0.250 - 0.294\ \Omega$.

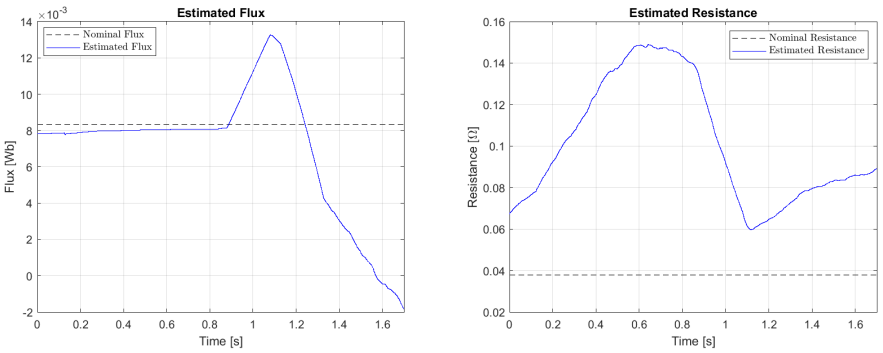
4.4.2 EKF - Modified Configuration

The data in this case was subjected to filtering using a third-order Butterworth filter, with a cutoff frequency of 200 Hz. The Kalman filter was initialised with (4.3), and was tuned using an *ad-hoc* method, which yielded the noise covariance matrices presented in (4.4).

$$x_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad P_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.3}$$

$$Q = \begin{pmatrix} 500 & 0 & 0 & 0 \\ 0 & 500 & 0 & 0 \\ 0 & 0 & 0.005 & 0 \\ 0 & 0 & 0 & 0.005 \end{pmatrix}, \quad R = \begin{pmatrix} 0.1 & 0 \\ 0 & 0.1 \end{pmatrix} \tag{4.4}$$

The results from this case are depicted in Figure 4.8.



(a) *(b)*

Figure 4.8: *Flux and Resistance estimated using the EKF with tuning and filtered data.*

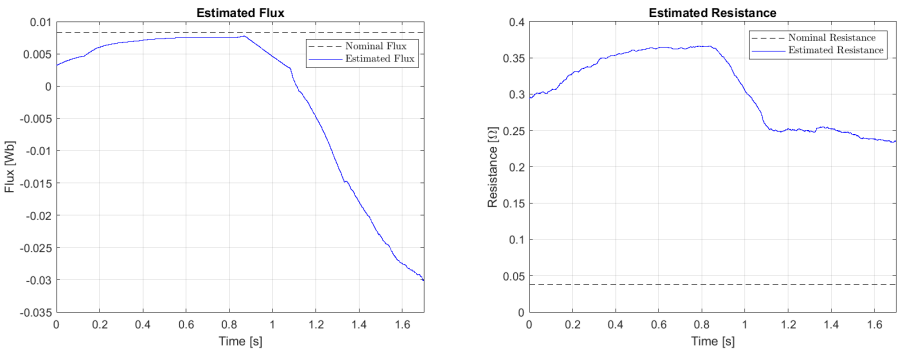
Figure 4.8a indicated that the flux estimation remained consistent during the run-down phase of the operation, with values ranging between $0.0078 - 0.0113\ Wb$. However, for the resistance estimation, the resistance steadily increased throughout the tightening phase, as depicted in Figure 4.8b. It ranged between $0.064 - 0.089\ \Omega$.

4.4.3 AEKF - Default Configuration

The data was left unfiltered in this case. The Kalman filter was initialised using (4.5) and the tuning parameters was set to $N_Q = 2$ and $N_R = 2$. This was due to the fact that if $N_Q = 1$ and $N_R = 1$ had been used, it would have been a standard EKF based on (2.35) - (2.36).

$$x_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad P_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.5}$$

The results from this case are visible in Figure 4.9.



(a) *(b)*

Figure 4.9: *The estimation of $Flux$ and $Resistance$ was carried out using the AEKF without any filtering of the data or tuning.*

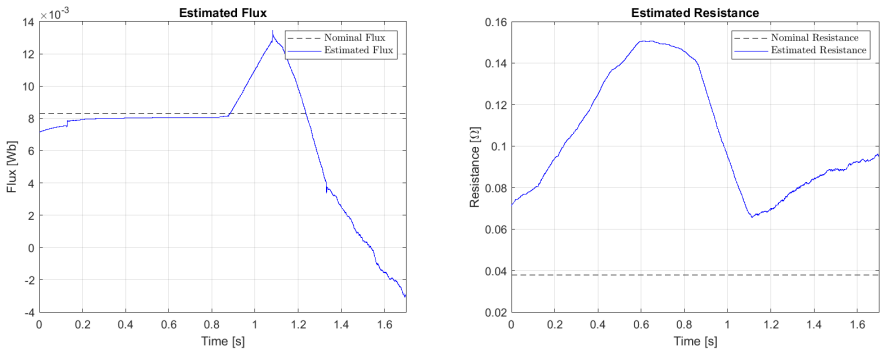
In Figure 4.9a, it could be observed that the flux estimation increased in the beginning but settled down later during the rundown phase. It spanned between $0.0032 - 0.0077\ Wb$. In the tightening phase, the resistance decreased as shown in Figure 4.9b, it spanned between $0.234 - 0.255\ \Omega$.

4.4.4 AEKF - Modified Configuration

A third-order Butterworth filter with a cutoff frequency of 200 Hz was applied to filter the data in this case. The Kalman filter was initialised using (4.6), and the tuning parameters were determined using the *ad-hoc* method. The values of N_Q and N_R were set to 10.

$$x_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad P_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.6}$$

The results from this case can be observed in 4.10.



(a) *(b)*

Figure 4.10: *The estimation of Flux and Resistance was carried out using the AEKF with filtering and tuning.*

The flux remained more or less constant in the rundown phase, as observed in Figure 4.10a. It fluctuated between $0.0072 - 0.0110\ Wb$. As for the resistance estimation it increased in the tightening phase, as depicted in Figure 4.10b. It fluctuated between $0.069 - 0.096\ \Omega$.

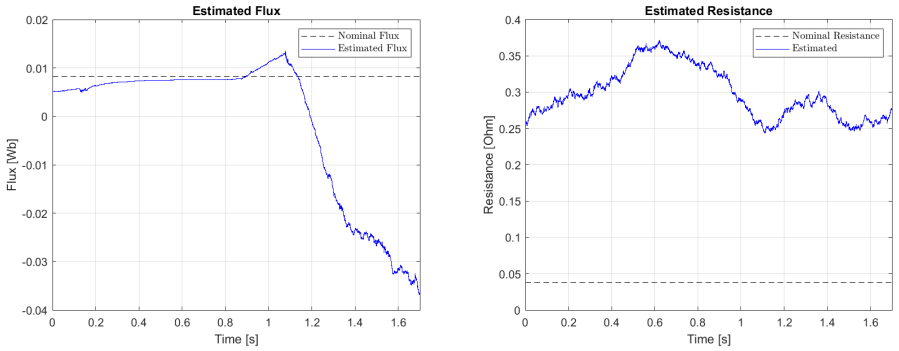
4.4.5 UKF - Default Configuration

In this case, the UKF was investigated with unfiltered data. It was initialised using (4.7) and the noise covariance matrices used can be seen in (4.8).

$$x_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad P_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.7}$$

$$Q = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad R = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{4.8}$$

The results from this case can be seen in 4.11.



(a) *(b)*

Figure 4.11: *The Flux and Resistance were estimated using the UKF without any data filtering or tuning.*

During the rundown phase, the flux was consistent and it ranged between 0.0050 – 0.0112 Wb as illustrated in Figure 4.11a. In the tightening phase, the resistance had a similar behavior as the flux. It ranged between 0.250 – 0.300 Ω as depicted in Figure 4.11b.

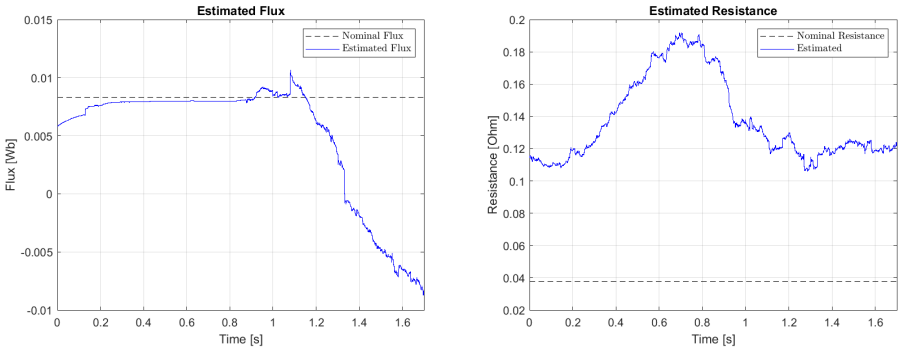
4.4.6 UKF - Modified Configuration

The data underwent filtering using a third-order Butterworth filter with a cutoff frequency of 200 Hz in this case. Additionally, tuning was applied, and the matrices associated with the tuning were determined using the *ad-hoc* method, as presented in (4.10). The Kalman filter was initialised with (4.9).

$$x_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad P_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.9}$$

$$Q = \begin{pmatrix} 100 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 100 \end{pmatrix}, \quad R = \begin{pmatrix} 0.03 & 0 \\ 0 & 0.03 \end{pmatrix} \tag{4.10}$$

The results for this case is evident in Figure 4.12.



(a) *(b)*

Figure 4.12: *The Flux and Resistance were estimated using the UKF with data filtering and tuning.*

Figure 4.12a illustrates the flux which were consistent throughout the rundown phase, and it spanned between 0.0058 – 0.0091 Wb. The resistance was more or less constant through the tightening phase, as depicted in Figure 4.12b, with levels spanning between 0.106 – 0.129 Ω.

Table 4.1: *Estimated temperature dependent parameters.*

Kalman filter	**Configuration**	**Flux** $[Wb]$	**Resistance** $[\Omega]$
EKF	Default	0.0042 – 0.0111	0.250 – 0.294
EKF	Modified	0.0078 – 0.0113	0.064 – 0.089
AEKF	Default	0.0032 – 0.0077	0.234 – 0.255
AEKF	Modified	0.0072 – 0.0110	0.069 – 0.096
UKF	Default	0.0050 – 0.0112	0.250 – 0.300
UKF	Modified	0.0058 – 0.0091	0.106 – 0.129

5
Real World

In this chapter, the process of real world data extraction is explained. Additionally, a specific test case focusing on a tightening operation will be presented. The acquired real-world data, along with the outcomes of the Kalman filters, will be presented to showcase the performance of the parameter estimation techniques.

5.1 Real World Setup

The setup used for real world measurements can be seen in Figure 5.1. This setup consisted of the tool (red arrow) that was being investigated, a computer with the data acquisition program (green arrow). The data from the tool was acquired via a converter box (orange arrow) which converted data in order for the DEWEsoft hardware (yellow arrow) to be able to read the data. Finally the screw joint (blue arrow) was used to simulate a tightening operation in a controlled environment.

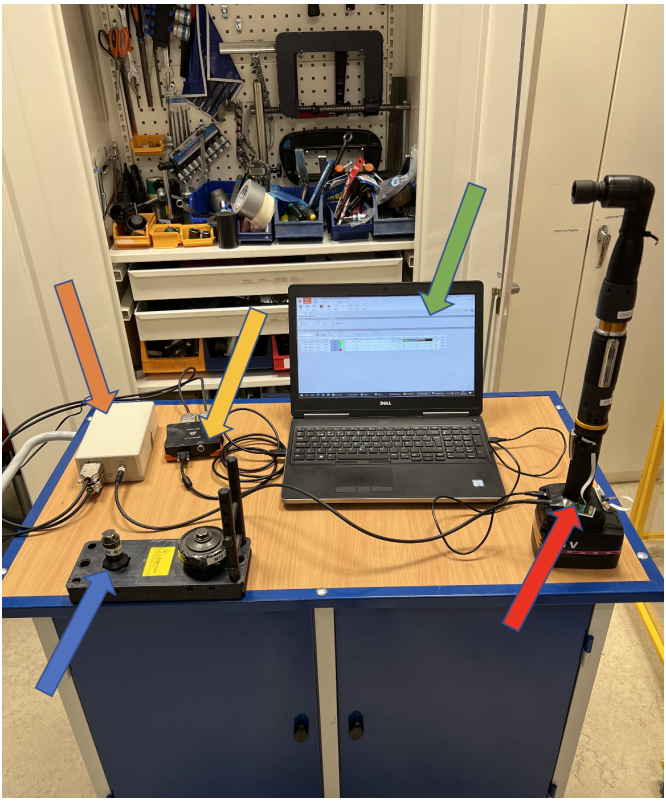
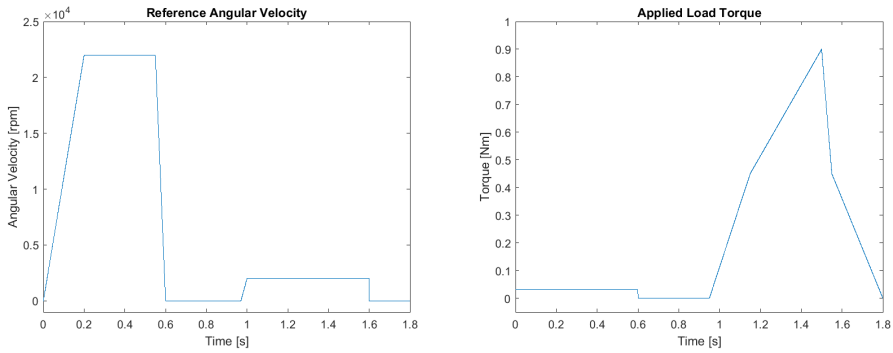


Figure 5.1: *An overview of the setup used for data acquisition.*

The necessary steps that were used in order to conduct measurements of a tightening operation are summarised in Appendix A.0.2.

5.2 Test Case

This test case followed a similar approach as described in Section 4.2, using the same references but with a difference in their implementation, the implementation being an Atlas Copco specific tool configuration. It involved a standard tightening operation, but with different reference levels due to the constraints imposed by the construction of the tool, which limited the achievable maxima. As with Section 4.2, the flux remained the parameter of interest during the run-down phase and the resistance over the tightening phase.



(a) *(b)*

Figure 5.2: *Reference angular velocity and reference torque for the real world test.*

As depicted in Figure 5.2b, the reference torque initially applied a small value to thread down the screw. From 0 to 0.6 seconds, the angular velocity gradually increased to its maximum limit and remained constant, simulating the rundown phase, as evident in Figure 5.2a. Around 0.6 seconds, the rundown phase was finished, followed by a resting phase lasting for the subsequent 0.4 seconds. The tightening phase commenced at approximately 1 second and extended until 1.6 seconds, as observed in Figure 5.2a, where the angular velocity dropped to zero. Concurrently, the load torque gradually decreased, as shown in Figure 5.2b.

5.3 Real World Raw Data

The raw data was obtained in a 16-bit format, where one bit was allocated for representing the sign of the data, indicating whether it was positive or negative. In other words, the data was represented using 15 bits, allowing for a maximum value of 32768. However, the specific maximum value represented varied depending on the type of measured data. For instance, in the case of current measurements, the maximum value corresponded to 187.5 A, while for angular velocity, the maximum value was $4000\pi \ \frac{rad}{s}$. To convert the data into SI units, conversion factors provided in (5.1) - (5.6) were applied.

$$i_d = \frac{i_{d,Measured}}{\left(\frac{32768}{187.5}\right)} \tag{5.1}$$

$$i_q = \frac{i_{q,Measured}}{\left(\frac{32768}{187.5}\right)} \tag{5.2}$$

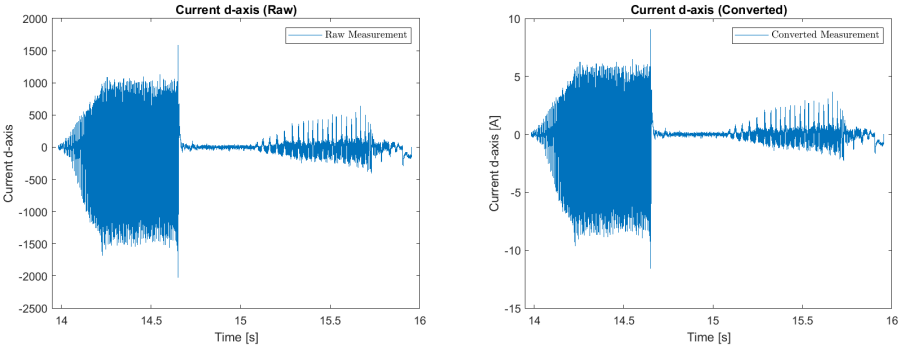
$$v_{Bat} = v_{Bat,Measured} \cdot \left(\frac{84}{32768}\right) \tag{5.3}$$

$$v_d = \frac{v_{d,Measured}}{\left(\frac{32768}{vBat}\right)} \tag{5.4}$$

$$v_q = \frac{v_{q,Measured}}{\left(\frac{32768}{vBat}\right)} \tag{5.5}$$

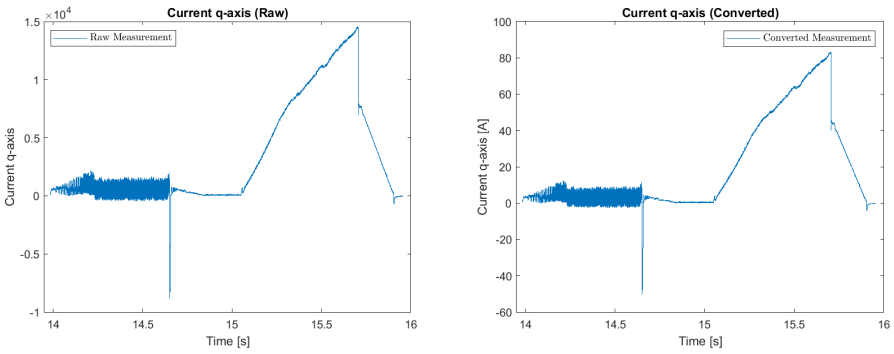
$$w_e = \frac{w_{e,Measured}}{\left(\frac{32768}{4000*\pi}\right)} \tag{5.6}$$

The duration of the rundown phase extended from approximately 14 – 14.7 seconds, while the tightening phase spanned from around 15 – 15.7 seconds. The interval between these two phases constituted the resting phase. Both the raw data extracted from DEWEsoft and the data converted using equations (5.1) - (5.6) resulted in the measurements depicted in Figure 5.3 - 5.7.



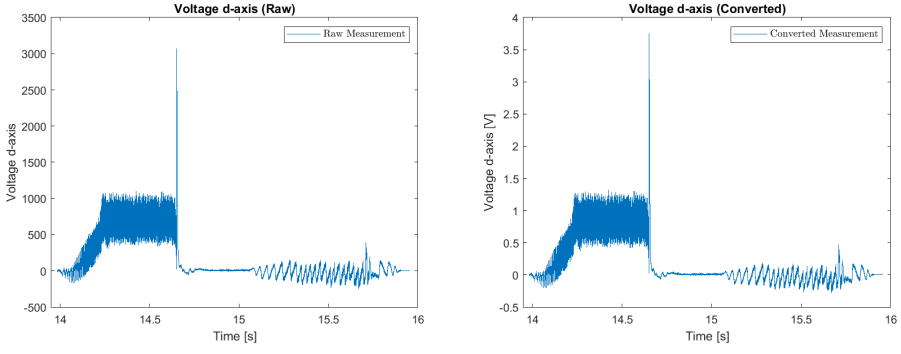
(a) *(b)*

Figure 5.3: *In 5.3a the raw i_d measurement can be observed while in 5.3b the converted i_d is illustrated.*



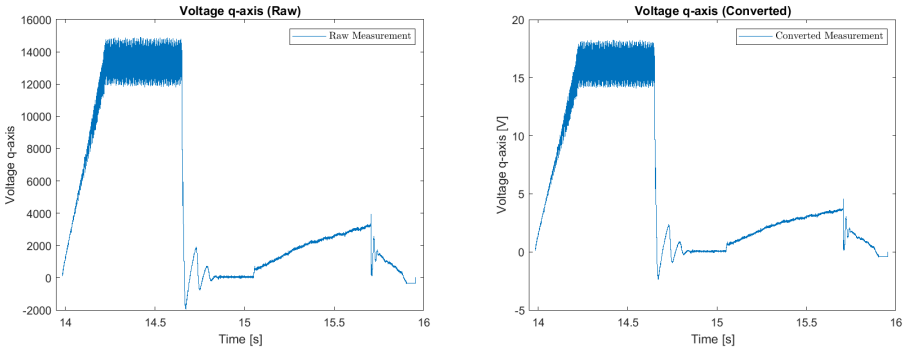
(a) *(b)*

Figure 5.4: *In 5.4a the raw i_q measurement can be observed while in 5.3b the converted i_q is illustrated.*



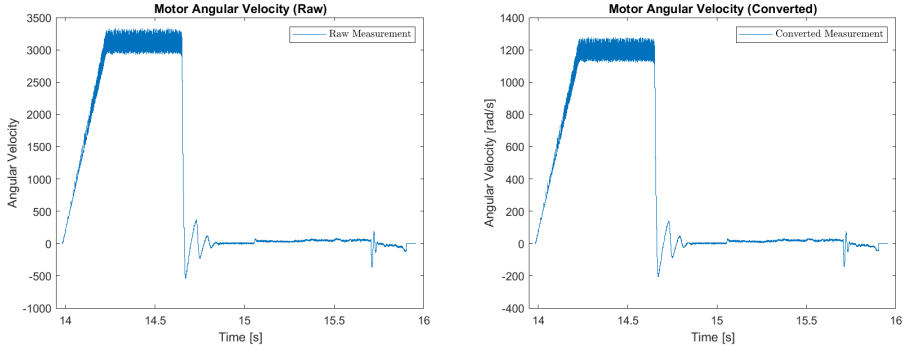
(a) *(b)*

Figure 5.5: *In 5.5a the raw v_d measurement can be observed while in 5.5b the converted v_d is illustrated.*



(a) *(b)*

Figure 5.6: *In 5.6a the raw v_q measurement can be observed while in 5.6b the converted v_q is illustrated.*



(a) *(b)*

Figure 5.7: *In 5.7a the raw ω_m can be observed while in 5.7b the converted ω_m is illustrated.*

5.4 Kalman Filter Evaluation

In this section, an evaluation of the Kalman filters was conducted using real world data. The evaluation process was done in the same way as in Section 4.4 and the estimated temperature dependent parameters are summarised in Table 5.1.

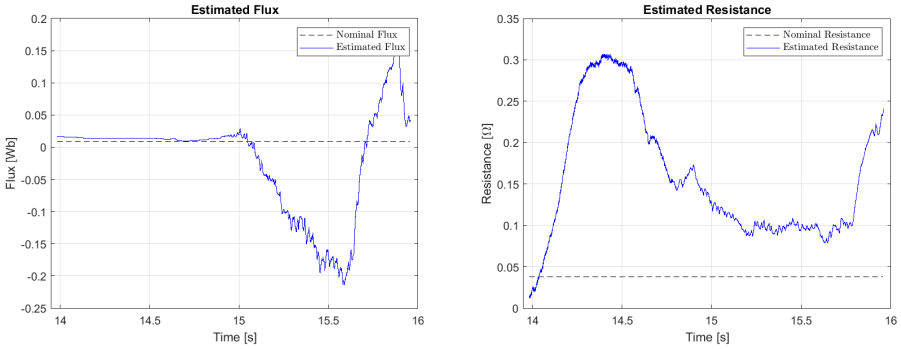
5.4.1 EKF - Default Configuration

In this case, the EKF was evaluated under the default configuration with the exception of a small tune that was applied in order to avoid a singularity issue, which rendered itself in (5.8). The Kalman filter was initialised using (5.7).

$$x_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad P_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5.7}$$

$$Q = \begin{pmatrix} 1.5 & 0 & 0 & 0 \\ 0 & 1.5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad R = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{5.8}$$

The results for this case can be seen in Figure 5.8.



(a) *(b)*

Figure 5.8: *The estimation of $Flux$ and $Resistance$ was carried out using the EKF, without applying filtering to the data and with a small tune.*

In the area of interest, the flux estimation was consistent and fluctuated between $0.0098 - 0.0168$ Wb based on Figure 5.8a. The resistance remained relatively constant throughout the tightening phase, as illustrated in Figure 5.8b. The resistance fluctuated between $0.079 - 0.126$ Ω.

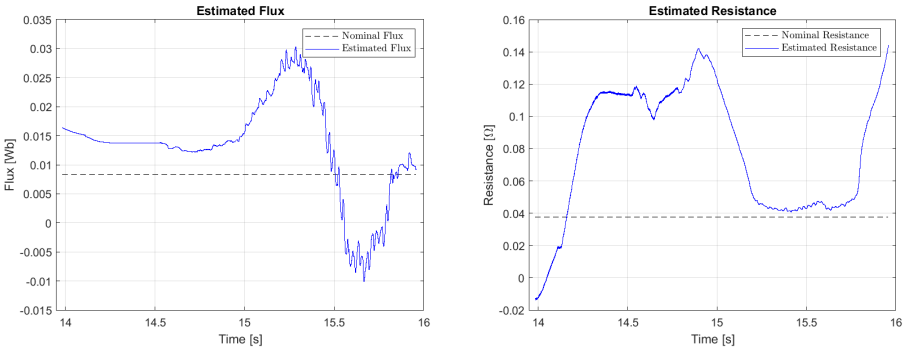
5.4.2 EKF - Modified Configuration

The data underwent a filtering process using a third-order Butterworth filter with a cutoff frequency of 200 Hz in this case. The Kalman filter was initialised with (5.9) and tuned using an *ad-hoc* method, resulting in the matrices presented in (5.10).

$$x_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad P_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5.9}$$

$$Q = \begin{pmatrix} 100 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 \\ 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0.1 \end{pmatrix}, \quad R = \begin{pmatrix} 10 & 0 \\ 0 & 10 \end{pmatrix} \tag{5.10}$$

The results for this case are shown in Figure 5.9.



(a) *(b)*

Figure 5.9: *The estimation of $Flux$ and $Resistance$ was carried out using the EKF, with a tune and filtering of the data.*

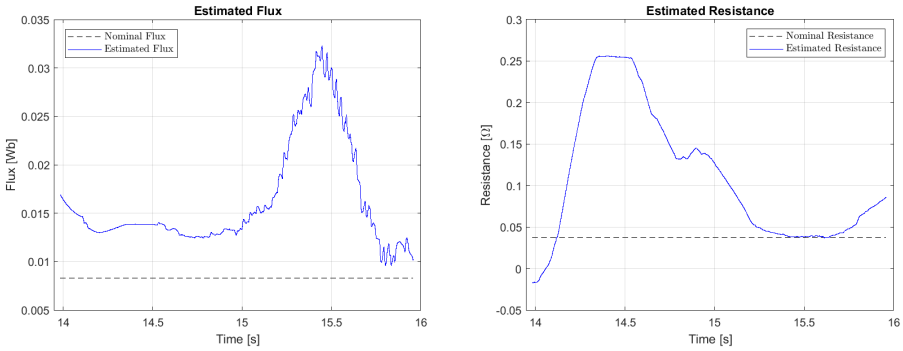
The flux estimation was more or less constant and ranged between 0.0122 – 0.0163 Wb in the rundown phase, indicated in Figure 5.9a. For the tightening part, the resistance decreased in the beginning and increased towards the end. It variated between 0.042 – 0.119 Ω, observed in Figure 5.9b.

5.4.3 AEKF - Default Configuration

In this case, the data was not filtered. The Kalman filter was initialised with (5.11), and the tuning was done in a similar manner as in Section 4.4.3.

$$x_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad P_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5.11}$$

The results from this case are presented in Figure 5.10.



(a) *(b)*

Figure 5.10: *The estimation of Flux and Resistance was performed using the AEKF without any filtering and with a small tune.*

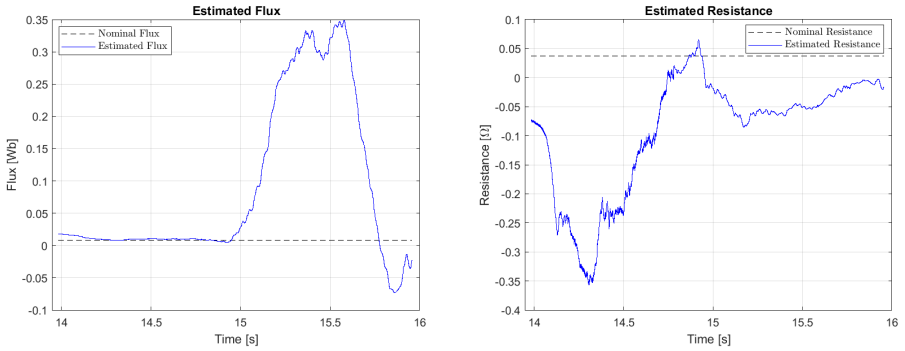
The flux estimation decreased in the beginning but settled quickly in the run-down phase which is observed from Figure 5.11a, it spanned between 0.0126 – 0.0169 Wb. The resistance was decreasing until around the middle of the tightening phase and then increased towards the end, which is depicted in 5.11b. It spanned between 0.038 – 0.124 Ω.

5.4.4 AEKF - Modified Configuration

In this case, the data was filtered using a third-order Butterworth filter with a cutoff frequency of 200 Hz . A tune was found using the *ad-hoc* method and resulted in $N_Q = 3$ and $N_R = 100$. The Kalman filter was initialised using (5.12).

$$x_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad P_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5.12}$$

The results from this case can be seen in Figure 5.11.



(a) *(b)*

Figure 5.11: *The estimation of Flux and Resistance was performed using the AEKF with filtering and tuning applied.*

The flux estimation from the rundown phase was consistent and fluctuated between $0.0083 - 0.0181\ Wb$, which can be seen in Figure 5.11a. The resistance was more or less constant through the tightening phase which could be observed in Figure 5.11b. It fluctuated between $-0.086 - -0.018\ \Omega$.

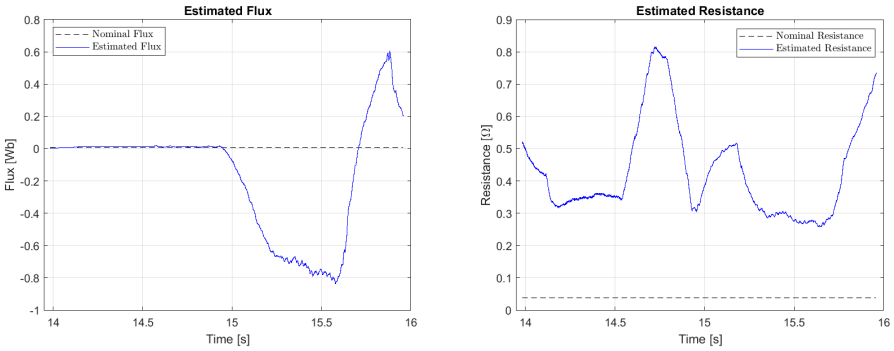
5.4.5 UKF - Default Configuration

The data was not filtered in this case. The UKF was initialised using (5.13). A default tune was applied which can be seen in (5.14).

$$x_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad P_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5.13}$$

$$Q = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad R = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{5.14}$$

The results from this are shown in Figure 5.12.



(a) *(b)*

Figure 5.12: *The estimation of Flux and Resistance was conducted using the UKF without applying any data filtering techniques or tuning.*

In Figure 5.12a, it could be seen that the flux estimation remained more consistent throughout the rundown phase. It ranged between $0.0024 - 0.0193\ Wb$. The resistance estimation increased, to then decrease and lastly increase towards the end of the tightening phase, as depicted in Figure 5.12b. It ranged between $0.258 - 0.516\ \Omega$.

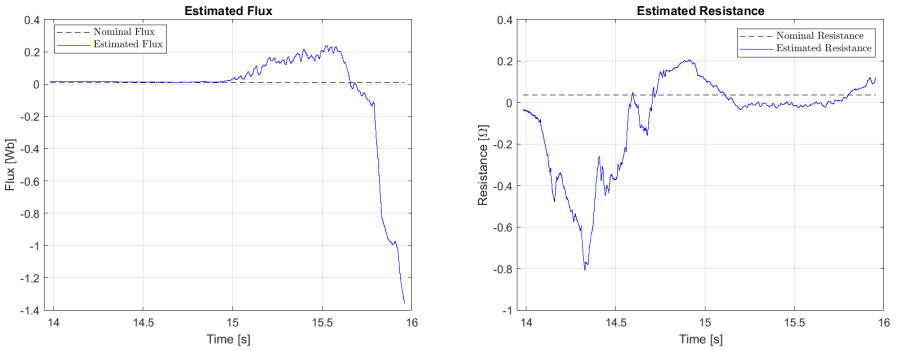
5.4.6 UKF - Modified Configuration

In this case, the data was filtered using a third-order Butterworth filter with a cutoff frequency of 200 Hz. The Kalman filter was tuned using the *ad-hoc* method and the matrices can be seen in (5.16). The Kalman filter was initialised using (5.15).

$$x_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad P_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5.15}$$

$$Q = \begin{pmatrix} 10 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0.1 \end{pmatrix}, \quad R = \begin{pmatrix} 0.03 & 0 \\ 0 & 0.03 \end{pmatrix} \tag{5.16}$$

The results for this case are showcased in Figure 5.13.



(a) *(b)*

Figure 5.13: *The estimation of $Flux$ and $Resistance$ was conducted using the UKF by applying data filtering techniques and tuning.*

Figure 5.13a illustrated the consistency of the flux estimation throughout the rundown phase. The estimation spanned between 0.0096 – 0.0160 Wb. The resistance was more or less constant through the tightening phase as depicted in Figure 5.13b. The estimation spanned from −0.034 – 0.112 Ω.

Table 5.1: *Estimated temperature dependent parameters.*

Kalman filter	**Configuration**	**Flux** $[Wb]$	**Resistance** $[\Omega]$
EKF	Default	0.0098 – 0.0168	0.079 – 0.126
EKF	Modified	0.0122 – 0.0163	0.042 – 0.119
AEKF	Default	0.0126 – 0.0169	0.038 – 0.124
AEKF	Modified	0.0083 – 0.0181	(−0.086) – (−0.018)
UKF	Default	0.0024 – 0.0193	0.258 – 0.516
UKF	Modified	0.0096 – 0.0160	(−0.034) – 0.112

6

Potential Estimation Challenges

This chapter treats the potential information deficiency apparent in Kalman filters for this type of application. It also includes potential problems such as fluctuating i_d, conversion errors and output discrepancy.

6.1 Estimation Difficulties

Due to the estimations not correlating with the nominal values an investigation regarding potential problems was conducted in this section.

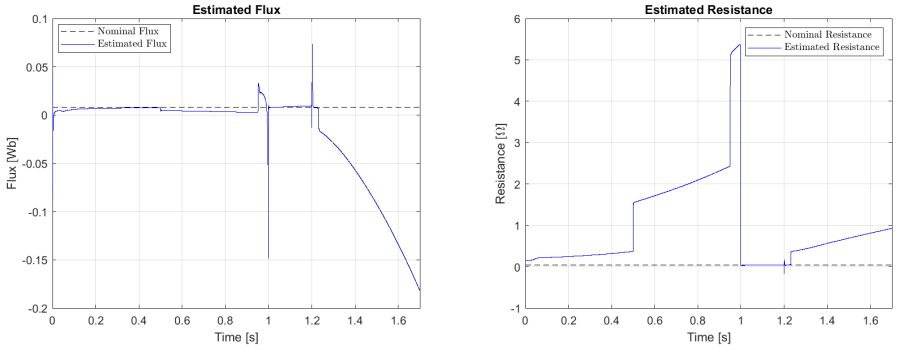
6.1.1 Test 1 - Information Deficiency

This case was created based on the EKF to showcase the problems that possibly occurred when there was an insufficient amount of information present for the Kalman filters to converge. This situation occurred when the value of the d-axis current was equal to zero. The data used for this case was generated without any noise in the measurements. The filter was initialised using (6.1) and the noise covariance matrices used can be seen in (6.2).

$$x_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad P_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{6.1}$$

$$Q = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad R = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{6.2}$$

The results of this case can be seen in Figure 6.1.



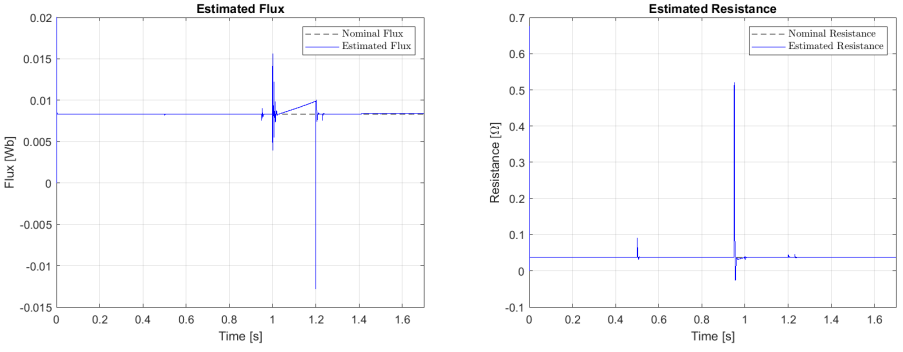
(a) *(b)*

Figure 6.1: *Flux and Resistance estimated under insufficient amount of information.*

It could be seen from Figure 6.1a and 6.1b that this case created difficulties for the EKF to find the nominal values of both the flux and resistance.

Adding Information

The potential information deficiency was handled by injecting a current of $i_d = 0.5\ A$ into the d-axis. This case generated the results observed in Figure 6.2.



(a) *(b)*

Figure 6.2: *Flux and Resistance estimated using added information.*

The estimated flux and resistance found the nominal values throughout the tightening operation, with some exceptions when there were sudden changes in the angular velocity reference.

6.1.2 Test 2 - Fluctuating *d*-axis Current

A large current injection was used to test the hypothesis that a d-axis current fluctuating around zero aggravated the estimation of flux and resistance.

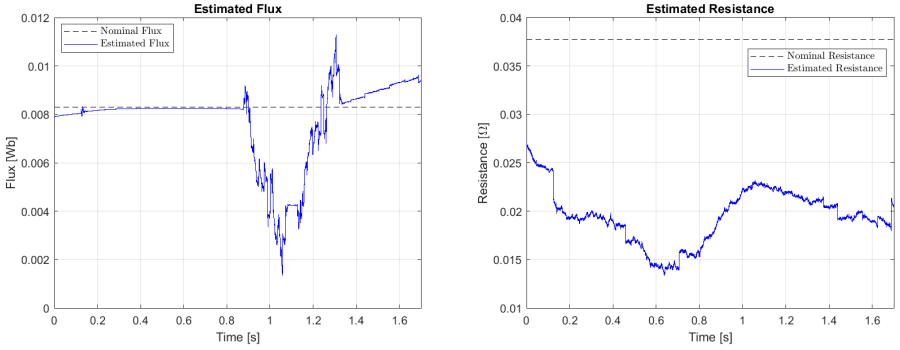
EKF - Default Configuration

The data was not filtered in this case. The Kalman filter was initialised using (6.3) and the noise covariance matrices used can be seen in (6.4).

$$x_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad P_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{6.3}$$

$$Q = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad R = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{6.4}$$

When observing the d-axis current, it was apparent that in order to push the d-axis current to be strictly positive, an injection of 30 A in the d-axis was necessary. This resulted in Figure 6.3.



(a) *(b)*

Figure 6.3: *A positive injection in the d-axis and its effect on Flux and Resistance.*

Both the flux and resistance were consistent in their respective phases which can be seen in Figure 6.3a and 6.3b respectively. The flux fluctuated between $0.0034 - 0.0092$ Wb in the rundown phase, while the resistance ranged between $0.018 - 0.023$ Ω in the tightening phase.

When observing the d-axis current, it was apparent that in order to push the d-axis current to be strictly negative, an injection of -30 A in the d-axis was necessary. This resulted in Figure 6.4.

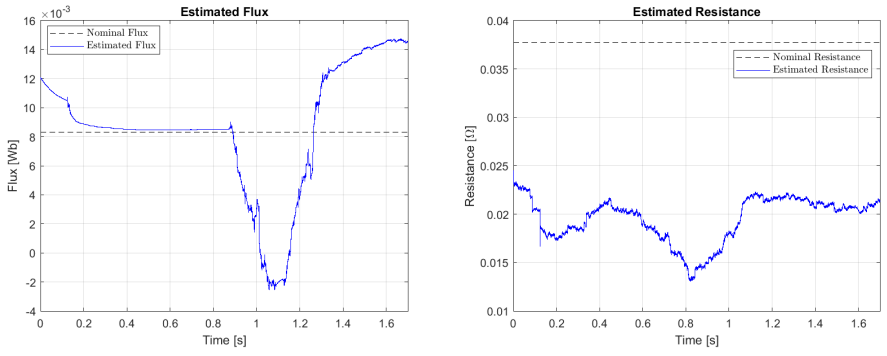
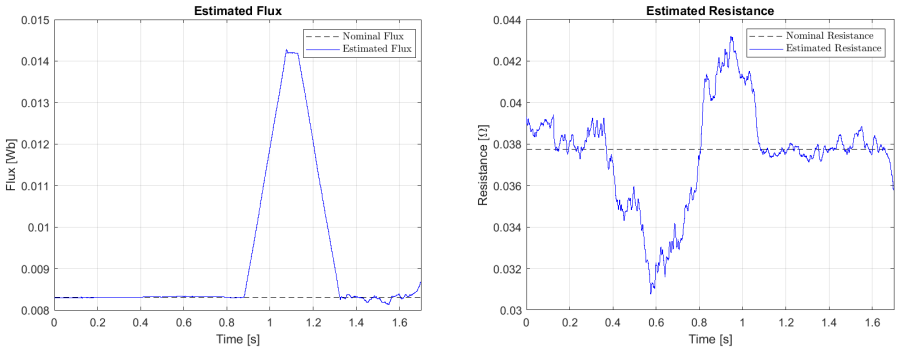


Figure 6.4: *A negative injection in the d-axis and its effect on Flux and Resistance.*

Similar to the positive injection variant, both estimates were more or less constant as depicted in Figure 6.4a and 6.4b. The flux spanned between $0.0015 - 0.012$ Wb in the rundown phase. The resistance fluctuated between $0.020 - 0.022$ Ω in the tightening phase.

EKF - Modified Configuration

In this case, the same filter and tune as in Section 4.4.2 were used. By observing the d-axis current, it was apparent that in order to push the d-axis current to be strictly positive, an injection of 6 A in the d-axis was necessary. This resulted in Figure 6.5.

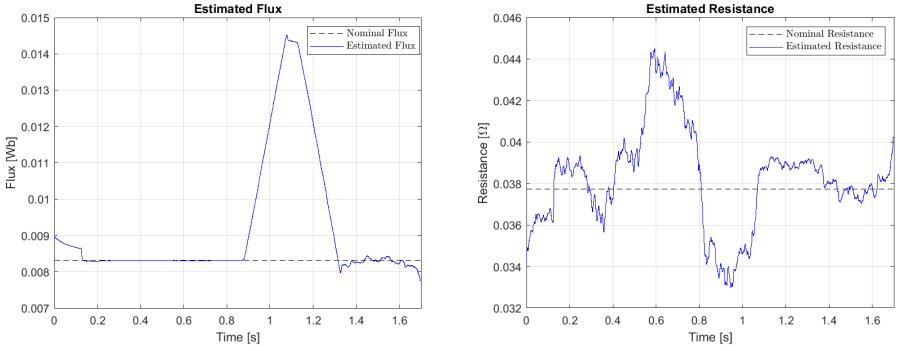


(a) *(b)*

Figure 6.5: *The impact of a positive injection in the d-axis on the estimation of $Flux$ and $Resistance$ when the data was filtered and the EKF was tuned.*

The flux was consistent in the rundown phase as seen in Figure 6.5a and exhibited fluctuations between $0.00829 - 0.012\ Wb$. The resistance however decreased in the tightening phase, evident from Figure 6.5b. It varied between $0.036 - 0.039\ \Omega$.

It could be observed that in order to ensure a strictly negative d-axis current, an injection of $-6\ A$ in the d-axis was required. Consequently, Figure 6.6 were generated.



(a) *(b)*

Figure 6.6: *The impact of a negative injection in the d-axis on the estimation of $Flux$ and $Resistance$ when the data was filtered and the EKF was tuned.*

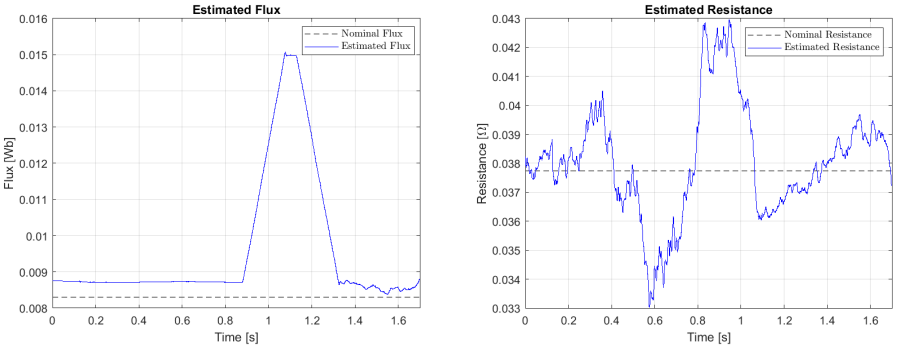
A similar behaviour could be observed for the flux as in the positive injection case, it can be observed in Figure 6.6a. The flux spanned between $0.0083 - 0.012\ Wb$ in the rundown phase. The resistance however was this time more or less constant compared to the positive injection case, as depicted in Figure 6.6b. The resistance

varied between 0.037 – 0.04 Ω in the tightening phase.

6.1.3 Test 3 - Conversion Discrepancies

By comparing the simulated angular velocity, seen in Figure 4.6b, against the real world angular velocity, seen in Figure 5.7b, a discrepancy was identified. This could indicate that the conversion factors found in (5.1) - (5.6) might have had an influence on the estimations. A sensitivity analysis on simulated data was used to test a hypothesis that the conversion factors could influence the estimations.

The first part of this test was to keep the angular velocity unchanged while the rest of the inputs were manipulated in order to simulate a deviation. By comparing the figures from Section 4.3 against 5.3 one could observe that there was a difference in the levels within the data. Hence the gains were arbitrarily chosen to somewhat reflect the difference. They were set to $i_d = 1.07$, $i_q = 1.05$, $v_d = 1.03$ and $v_q = 1.05$. The same settings as in Section 4.4.2 were used.

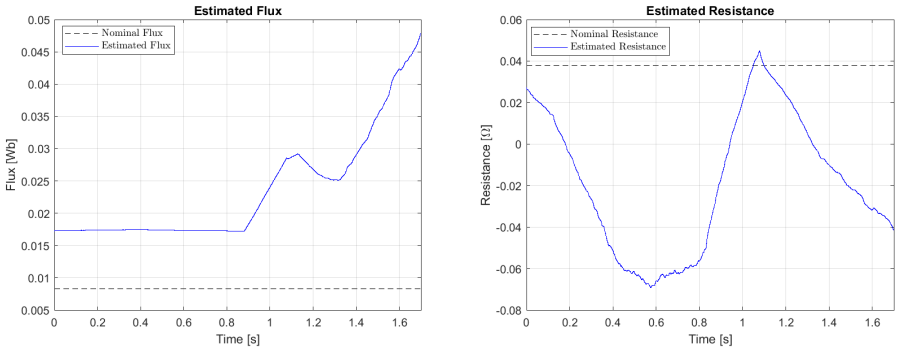


(a) *(b)*

Figure 6.7: *Flux and Resistance under the influence of chosen gains, resulted in small deviations from nominal levels.*

Based on Figure 6.7a the flux was consistent and ranged between $0.0087 - 0.0125\ Wb$ during the rundown phase. During the tightening phase the resistance ranged between 0.036 – 0.04 Ω. It was more or less constant, which can be seen in Figure 6.7b.

The tightening program constructed in Section 5.2 yielded a maximum angular velocity of around 22400 rpm which in $\frac{rad}{s}$ was roughly 2350 $\frac{rad}{s}$. In Figure 5.7b it could be observed that the angular velocity were roughly half of the angular velocity in the constructed program. This lead to the second part of this test where the angular velocity was halved but the gains from the first part were set to one, in order to study its effect on the estimation results.



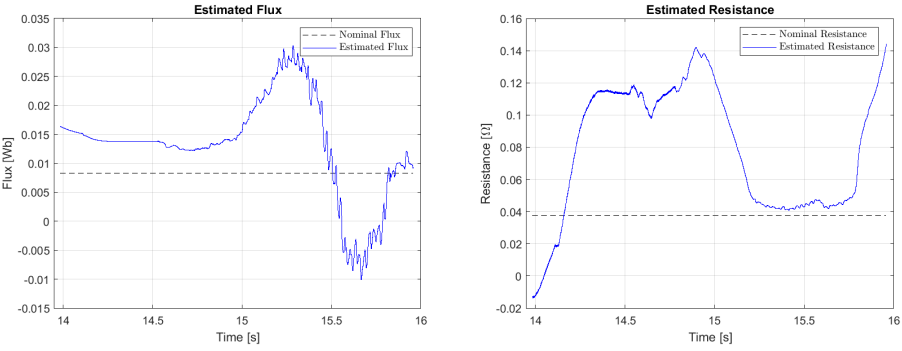
(a) *(b)*

Figure 6.8: *Flux and Resistance estimated under the influence of half the angular velocity.*

As shown in Figure 6.8a, the flux did not change a lot during the rundown phase and ranged between $0.0172 - 0.0241\ Wb$. Observed in Figure 6.8b, the resistance decreased during the tightening phase. It spanned between $-0.042 - 0.023\ \Omega$.

6.1.4 Test 4 - Output Discrepancies

Observations of Figure 4.4b and 5.5b showed a discrepancy between the signs for the d-axis voltage. In simulation, the d-axis voltage was negative during rundown whereas it was positive in the real measurements. The sign in front of u_1 in (2.24) was switched, to test the hypothesis that the wrong sign could affect the accuracy of the estimates. The same settings as in Section 5.4.2 were used. First, the variant with the plus sign in front of u_1 in (2.24) was used.

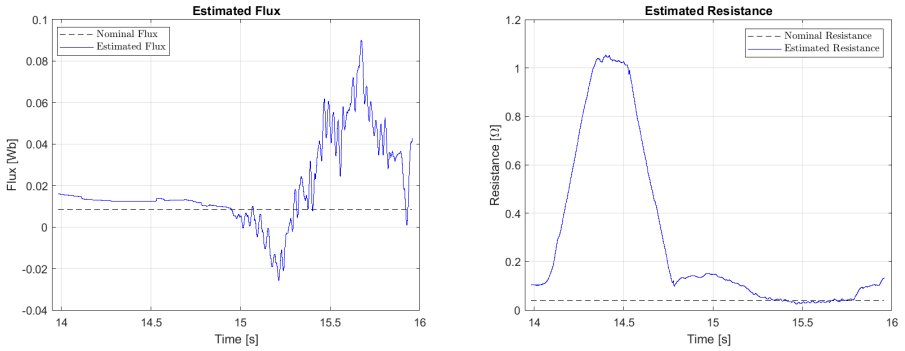


(a) *(b)*

Figure 6.9: *The estimation of Flux and Resistance was carried out using the state transition function with a plus sign in front of u_1.*

The estimation of flux was more consistent during the rundown phase of the tightening operation, as can be seen in Figure 5.9a. The flux fluctuated between $0.0122 - 0.0163\ Wb$. The resistance decreased in the beginning of the tightening phase, to then bottom out in the middle and then increase towards the end as showcased in Figure 5.9b. The resistance ranged between $0.042 - 0.119\ \Omega$.

After the variant with the plus sign in front of u_1, the negative sign was used.



(a) *(b)*

Figure 6.10: *The estimation of $Flux$ and $Resistance$ was carried out using the state transition function with a minus sign in front of u_1.*

The estimation of flux demonstrated consistent behaviour during the rundown phase, as depicted in Figure 6.10a. It exhibited fluctuations within the range of $0.0125 - 0.0157\ Wb$. In Figure 6.10b it could be seen that the resistance decreased during the tightening phase and varied between $0.026 - 0.148\ \Omega$.

7

Discussion

This chapter will discuss the implementation, results as well as the challenges found. Other findings that occurred as the work progressed, affecting the process are also discussed.

7.1 Discussion

The following sections will discuss the implementation and results obtained during this thesis. It will also discuss the estimation challenges that were encountered throughout the process.

7.1.1 Implementation and Results

Kalman filters were selected for parameter estimation in the PMSM application due to their ability to handle non-linearities and uncertainties. Given the dynamic nature of the typical tightening operation at Atlas Copco, it was suitable to employ estimation algorithms capable of adapting to changing conditions. The filter's tunability further supported this decision, since it allowed for adaptability. The investigation of three different Kalman filters, aimed to identify if one Kalman filter was deemed better than the other ones with respect to estimating resistance and flux. Although the EKF, UKF, and AEKF are all Kalman filter variants, Sections 2.4 - 2.5 showcased a difference in the state transition. The EKF linearises the state transition and observation vectors based on the current estimate, while the UKF utilises the UT to propagate the state mean and covariance without requiring linearisation. It was of interest to see if this difference had any effect on the performance of the different Kalman filter variants.

The initial decision was made to develop only a Kalman filter that could estimate the resistance, based on the construction of the motor in consideration. The temperature sensor that this thesis aimed at replacing with estimation algorithms was placed within the copper wires, hence the resistance was of interest. Once the estimation of the resistance was deemed to work, the question arose as to whether it was possible to estimate the flux simultaneously. Further investigation was conducted, and the model was extended to include the flux, as can be seen in (3.10). Even though the inductance L_s is a temperature dependent parameter, it was kept constant since it would otherwise have created an underdetermined problem. This was due to the fact that there would only be two equations available, but three unknown parameters.

By observing Section 4.4 and 5.4 we could see that all Kalman filters had problems with estimating the flux and resistance. It should be mentioned that there was not enough time to test all combinations, with that we mean that we did not try to only filter the data but not tune the data and vice versa. Neither did we put in a lot of time to find the optimal tuning for each and every Kalman filter. This means that perhaps there is better tuning options that has not been tried due to the amount of time it takes to tune a filter using the *ad-hoc* method.

The significance of correct PMSM parameters had a big impact on the behavior of the PMSM system. Specifically, these parameters influenced the system's outputs. Initially, we received a Simulink Data Dictionary containing parameter values, which were assumed to be in SI-units and implemented accordingly. However, it was discovered that this was not the case. For instance, when we implemented a viscous damping value from the dictionary, it affected the current required to overcome resistive forces. Consequently, the motor model demanded significantly higher current in the q-axis to generate the desired torque, compared to using the correct viscous damping value. The nominal flux parameter was another example. An incorrect value not only resulted in inaccurate flux estimation but also impacted the resistance. This confusion led us to question the model and the Kalman filter. It turned out that the nominal flux parameter had an incorrect value, affecting the entire PMSM model. Hence, correct parameters were very important.

Considering the unsatisfactory results obtained from the real world data tests, it was decided to not convert the estimated temperature-dependent parameters, namely resistance and flux, into actual temperature values. The conversion would have involved employing a linear relationship utilising temperature coefficients. Although the temperature coefficients for copper (windings) and neodymium magnets (used in the rotor) are small, accurate estimates are important due to the magnification effect of even small deviations, see (2.52). Inaccurate estimates could lead to significant temperature deviations that exceed the desired level of usefulness.

The thermal model was introduced when the simulation model was considered

finished. Since the simulation model of the PMSM lacked internal dynamics, all parameters were kept static throughout the simulation. In order to test the filters ability to trace changes, a simplified thermal model was constructed. This model allowed simulation of a theoretical temperature increase within the tool based on the losses generated during a run. This temperature increase was then used to create a linear increase in the temperature dependent parameters. The filter was then tested and the tests concluded that it is possible for the Kalman filters to estimate an increase of certain parameters during a run. Making it a viable option for temperature monitoring.

7.1.2 Estimation Challenges

The first result presented in Chapter 6, more specifically Section 6.1.1 was perhaps the most unexpected. If one observe Figure 6.1 it could be seen that the estimations did not converge towards the nominal values but instead the seemed to deviate each time the motor experienced changes in the angular velocity.

Internal discussions took place in order to rule out possible sources of error. After some reasoning and testing, it was found that one possible source of the error could be due to the lack of information available for the estimation. The source behind this was the combination of a FOC together with a PMSM. The FOC had by default a zero current as its reference in the d-axis. By injecting a small current into the d-axis we could see that it significantly improved the estimation of the resistance and flux, which could be observed in Figure 6.2. This implied that the information scarcity was a factor behind the poor estimates, however we are not certain that this is the sole reason for the poor estimates. Further investigation is needed to confirm that this is the case. One way to address this problem could be to perform a rigorous observability analysis of the whole system.

When utilising a FOC, the idea was to keep the d-axis current as close to zero as possible. This was because an increase in the amount of current flowing through the d-axis corresponds to a decrease in the amount of current flowing through the q-axis, see Figure 2.5. This relationship was a consequence of converting the currents from the abc-frame to the dq-frame. Something that was necessary to do in order to simplify working with the equations constituting the FOC, PMSM and Kalman filter. With other words, if a current injection was to be used, it should be as small as possible to not impede the performance of the motor.

Due to the project time being limited, only a constant current injection was considered. A current injection could be done in multiple ways, e.g a square or sinus wave could be used. The best way we believe would be to only inject a current when a sample would be taken. What we mean by this is that we were only interested in estimating the flux during the rundown phase of the tightening operation and the resistance during the tightening phase. We were not even interested in the whole segment but only the end of each phase since the application intended

was temperature monitoring. This could be explained using an example: if we e.g ran 50 consecutive tightening operations, we would not be interested in monitoring the temperature during the complete sequence, since this information would only be used to not destroy the tool. We could then minimise the information by only taking samples in the end of each rundown and tightening phase. This would be sufficient to monitor the temperature increase and shut of the motor in case of overheating. One could argue that, "what about the time in between the samples, could the temperature not increase during that time period?".

This is because the temperature in the magnets increases as the rotor spins, mainly due to Eddy currents and friction that both increases with angular velocity. When the rotor does not spin as fast during the tightening operation, the induced Eddy currents and friction are not as prominent. The same idea applies for the resistance. When a tightening operation is conducted, more current are induced into the copper wires, increasing the temperature in the wires. During the rundown phase, a smaller amount of current is induced meaning that the wires do not get heated up as much. Hence only injecting a current in the end of each phase would be sufficient to get information to monitor the temperature increase in the motor during operation.

Unfortunately a large current injection might have been necessary in this case. One reason behind this is the amount of noise in the signals. We could not imagine that the noise levels would be as great as they turned out to be. In the literature that constituted this thesis, most cases were only done on simulated data or when real data was used the noise levels were quite small. Leaving us to believe that that would also be the case for our application as well.

However, by observing Section 6.1.2 we could see that a sufficiently larger current injection into the d-axis was able to significantly improve the result of the estimation. The best results were found when both a Butterworth filter was utilised and the Kalman filter was tuned. By getting the d-axis current to become strictly positive or negative it seemed to solve the issue with bad performance. Why this is, we do not know unfortunately. We could take a guess but it would not bring any more light onto the problem.

In Section 6.1.3 it was showcased how the conversion factors affected the results. It was found after some testing that the angular velocity seemed to be low compared to the expected angular velocity set in the tightening program. This lead to an uncertainty analysis being conducted in order to determine how much a deviation in those factors could affect the results. When it comes to the first part of *test 2* the small gains did not affect the result that much, it arguably became slightly worse compared to when no gains were used. In the second part of *test 2*, the influence of an incorrect angular velocity on the results was investigated. It was observed that when the angular velocity was reduced to half of its expected value, there was a significant impact on the results, particularly on the estimation of resistance. This suggests that it is important that the conversions are correct

in order to ensure estimations that are reasonably close to the expected nominal values.

Section 6.1.4 was done since there existed a difference in the data extracted from simulation compared to the real world. The simulation model was based on Section 2.1.3. Despite that, the data looked similar for most of the variables except for the voltage in the d-axis, which can be observed by comparing Section 4.3 to 5.3. For the simulated data, the d-axis voltage was negative during the rundown phase while it was positive in the data from the real world. It could be seen that utilising a different sign in (2.24) affected the results. It was tested on real world data, suggesting that the variant with the minus sign was to be used. The results were still not as expected, however using a negative sign seemed to help the filter estimate both flux and resistance closer to the nominal values as well as making the range in which the estimations fluctuated tighter.

8 Conclusion

In this concluding chapter, the insights derived from this thesis are summarised. Potential areas for future investigation are also presented.

8.1 Conclusion

The thesis aimed to explore the feasibility of monitoring temperature dependent parameters in a PMSM using Kalman filters. For ideal simulation conditions, it was possible to monitor these parameters. However, non-ideal conditions posed challenges, limiting the applicability of the three Kalman filters since they performed equally for estimation of the temperature dependent parameters. The Kalman filters could work as intended if the presented challenges were to be resolved in an efficient manner, meaning that the performance of the PMSM is not sacrificed.

8.2 Future Work

If this thesis is continued up on in the future, it would be wise to conclude if the problems identified in this thesis could be resolved or not. Resolving these issues could enable the utilisation of Kalman filters and expansion of the models to include parameters for sensorless solutions, eliminating the need for physical sensors. Furthermore, by combining the motor model with a comprehensive thermal model it could allow for simulation of temperature variations, which are not present today due to the lack of internal dynamics. If the problems could not be resolved, alternative methods such as Model Reference Adaptive System (MRAS), Recursive Least Square (RLS), and Comprehensive Learning Particle Swarm Optimization with Opposition Based Learning (CLPSO-OBL) could be explored for

estimating temperature dependent parameters.

Appendix

A

Matlab Code EKF

Simulation and Motor Parameters

```
%Simulation and Motor Parameters, ITB-A61
Rs = 0.03774; % Nominal Winding resistance
Ls = 0.03264e-3; %Nominal Induction
p = 1; %Pole Pairs
J = 3.51e-6; %Rotor Inertia
f_sim = 3.45e-6; %Viscous Damping Coefficient
Flux0 = 0.00831; %Nominal Rotor Flux Linkage
Ts = 1.25e-4; %Sampling Time for Simulation

%Field Oriented Control Parameters:

%PI-gain
wm = Rs/Ls;
wc = 5*wm;
ws = wc/10;

%Speed Controller Gain
spdKi = 10*(ws^2)*J;
spdKp = 10*ws*J;

%Current Controller
Ki = 1*(wc^2)*Ls;
Kp = 1*wc*Ls;
```

Main EKF-Code

```
1
2  %For tuning of process noise matrix (Q), see
       CovariancePrediction.m
3  %For tuning of measurement noise matrix (R), see
       KalmanGain_res.m and KalmanGain_flux.m
4
5  %Load data from simulation
6  load('parameters.mat')
7
8  %Settings for Butterworth filter applied on id, iq, vd, vq,
       we
9  [b,a] = butter(3,0.05,'low');
10
11 %Initial conditions for states and covariance
12 x0 = [0;0;0;0];
13 P0 = diag([1, 1, 1, 1]);
14
15 %Allocate memory for estimations
16 x_predict = zeros(4,length(ans));
17
18 %Input parameters:
19 % vd = d-axis Voltage
20 % vq = q-axis Voltage
21 % we =  Electrical Angular Velocity
22
23 %State parameters:
24 % id = d-axis Current
25 % iq = q-axis Current
26
27 % x3 = Estimated Flux
28 % x4 = Estimated Resistance
29
30 %Main loop for both estimations
31 %Loops through the simulated data
32
33 for i = 1:length(time)
34
35     %Prediction Step
36
37     %Use initial conditions for first loop, (i = 1)
38     %else, use previous estimate
39
40     if i == 1
41         xk_1_k_1 = x0;
```

```
42      else
43          xk_1_k_1 = xk;
44      end
45
46      Ts(i) = Ts(i);
47      vd(i) = vd(i);
48      vq(i) = vq(i);
49      we(i) = we(i);
50
51      %State Prediction Function, see StatePrediction.m
52      xk_k_1 = StatePrediction(vd(i),vq(i),we(i),Ts(i),
            xk_1_k_1);
53
54      %Covariance prediction
55      %Use initial conditions for first loop, (i = 1)
56      %else, use previous covariance
57
58      if i == 1
59          Pk_1_k_1 = P0;
60      else
61          Pk_1_k_1 = Pk;
62      end
63
64      %Covariance predicition function, see
            CovariancePrediction.m
65      Pk_k_1 = CovariancePrediction(Pk_1_k_1,we(i),Ts(i),
            xk_1_k_1);
66
67      %Kalman Gain function, see KalmanGain.m
68      K  = KalmanGain(Pk_k_1);
69
70      %Correction step
71
72      id(i) = ans(2,i); %d-axis current
73      iq(i) = ans(3,i); %q-axis current
74
75      %State correction function, see StateCorrection.m
76      xk = StateCorrection(K,xk_k_1,id(i),iq(i));
77
78      %Covariance Correction function, see
            CovarianceCorrection.m
79      Pk = CovarianceCorrection(Pk_k_1,K);
80
81      %Save estimations in x_predict
82      x_predict(:,i) = xk;
83  end
```

```
84
85 %Settings for moving mean of Flux and Resistance
86 Flux_mean = movmean(x_predict(3,:),2000);
87 Rs_mean = movmean(x_predict(4,:),2000);
88
89 figure(1)
90 plot(ans(1,:), id);
91 hold on
92 plot(ans(1,:), x_predict(1,:));
93 xlabel('Time [s]');
94 ylabel('Current d-axis [A]');
95 title('Current d-axis');
96 legend('Measured $i_d$', 'Estimated $i_d$', 'Interpreter',
       'latex');
97
98 figure(2)
99 plot(ans(1,:), iq);
100 hold on
101 plot(ans(1,:), x_predict(2,:));
102 xlabel('Time [s]');
103 ylabel('Current q-axis [A]');
104 title('Current q-axis');
105 legend('Measured $i_q$', 'Estimated $i_q$', 'Interpreter',
       'latex');
106
107 figure(3)
108 plot(ans(1,:),x_predict(3,:));
109 hold on
110 plot(ans(1,:),Flux_mean);
111 xlabel('Time [s]');
112 ylabel('Flux [Wb]');
113 title('Estimated Flux');
114 legend('Estimated Flux','Moving Mean', 'Interpreter', '
       latex');
115
116 figure(4)
117 plot(ans(1,:), x_predict(4,:));
118 hold on
119 plot(ans(1,:),Rs_mean);
120 xlabel('Time [s]');
121 ylabel('Resistance [Ohm]');
122 title('Estimated Resistance');
123 legend('Estimated Resistance', 'Moving Mean', 'Interpreter'
       , 'latex');
```

State Prediction Step

```
1
2  function xk_k_1 = StatePrediction(vd,vq,we,Ts,xk_1_k_1)
3
4      %Parameters, ITB-A61
5      Ls = 0.03264e-3; %Nominal Inductance
6
7      %Estimated States
8      x1 = xk_1_k_1(1); % id
9      x2 = xk_1_k_1(2); % iq
10     x3 = xk_1_k_1(3); % Flux
11     x4 = xk_1_k_1(4); % Rs
12
13     %State transition matrix
14     fk = [(-x4/Ls)*x1 + we*x2 + vd/Ls;
15           -we*x1 - (x4/Ls)*x2 + vq/Ls - (x3*we)/Ls;
16           x3;
17           x4];
18
19     xk_k_1 = xk_1_k_1 + Ts*fk;
20 end
```

Covariance Prediction Step

```
1
2  function Pk_k_1 = CovariancePrediction(Pk_1_k_1,we,Ts,
       xk_1_k_1)
3
4      %Parameters, ITB-A61
5      Ls = 0.03264e-3; %Nominal Inductance
6
7      %Estimated states
8      x1 = xk_1_k_1(1); %id
9      x2 = xk_1_k_1(2); %iq
10     x3 = xk_1_k_1(3); %Flux
11     x4 = xk_1_k_1(4); %Rs
12
13     %Linearized State Transition Matrix
14     Fk = [1-(x4*Ts/Ls) we*Ts 0 -Ts*x1/Ls;
15           -we*Ts 1-(x4*Ts/Ls) -we*Ts/Ls -Ts*x2/Ls;
16           0 0 1 0;
17           0 0 0 1];
18
19     %Process Noise Matrix
20     Q = diag([1, 1, 1, 1]);
21
22     Pk_k_1 = Fk*Pk_1_k_1*Fk' + Q;
23 end
```

Kalman Gain

```
1
2  function K  = KalmanGain(Pk_k_1)
3
4      %Linearized Observation Matrix
5      H = [1, 0, 0, 0;
6           0, 1, 0, 0];
7
8      %Measurement Noise Matrix
9      R = diag([1, 1]);
10
11     K = Pk_k_1*H'*inv(H*Pk_k_1*H'+ R);
12 end
```

State Correction Step

```
1
2  function xk = StateCorrection(K,xk_k_1,id,iq)
3
4      %H*x
5      y_prim = [xk_k_1(1), xk_k_1(2)]';
6
7      %Measurements
8      y = [id;iq];
9
10     xk = xk_k_1 + K*(y-y_prim);
11 end
```

Covariance Correction Step

```
function Pk = CovarianceCorrection(Pk_k_1,K)

    %Linearized Observation Matrix
    H = [1, 0, 0, 0;
         0, 1, 0, 0];

    Pk = (eye(4) - K*H)*Pk_k_1;
end
```

Matlab Code AEKF

Main AEKF-Code

```
1
2  % Load Data From Simulation
3
4  load('parameters.mat')
5
6  %Initial conditions
7  x0 = [0;0;0;0];
8  P0 = diag([1, 1, 1, 1]);
9  Q0 = diag([1 1 1 1]);
10 R0 = diag([0 0]);
11 e0 = [0;0];
12
13 % Settings for Butterworth filter applied on id, iq, vd, vq
       , we
14 [b,a] = butter(3,0.05,'low');
15
16 %Allocate memory for estimations
17 x_predict = zeros(4,length(time));
18
19
20 % Input parameters:
21 % vd = d-axis voltage
22 % vq = q-axis voltage
23 % we =  electrical angular velocity
24
25 %State parameters:
26 % id = d-axis current
27 % iq = q-axis current
28
29 % Main loop for estimations
30 % Loops through the simulated data
31
32 for i = 1:length(time)
33
34     %Prediction step
35
36     %Use initial conditions for first loop, (i = 1)
37     %Else, use previous estimate
38
39     if i == 1
40         xk_1_k_1 = x0;
41     else
```

```
42          xk_1_k_1 = xk;
43      end
44
45      %In order to use the adaptive EKF, one needs to run two
            or more
46      %iterations, hence the if-statement below
47
48      if i >= 2
49          omega_k_1_bar = omega_k_bar;
50          Q_1 = Q;
51          R_1 = R;
52          ek_1_bar = ek_bar;
53
54      else
55          omega_k_1_bar = x0;
56          Q = Q0;
57          R = R0;
58          Q_1 = Q0;
59          R_1 = R0;
60          ek_1_bar = e0;
61      end
62
63      Ts(i) = 1.25e-4;
64      vd(i) = ans(4,i);
65      vq(i) = ans(5,i);
66      we(i) = ans(6,i);
67
68      %State Prediction Function, see StatePrediction.m
69      xk_k_1 = StatePrediction(vd(i),vq(i),we(i),Ts(i),
            xk_1_k_1);
70
71      %Covariance prediction
72      %Use initial conditions for first loop, (i = 1)
73      %else, use previous covariance
74
75      if i == 1
76          Pk_1_k_1 = P0;
77      else
78          Pk_1_k_1 = Pk;
79      end
80
81      %Covariance predicition function, see
            CovariancePrediction.m
82      [Pk_k_1,Fk] = CovariancePrediction(Pk_1_k_1,we(i),Ts(i)
            ,xk_1_k_1,Q);
83
```

```
84      %Kalman Gain Function, see KalmanGain.m
85      K   = KalmanGain(Pk_k_1,R);
86
87      %Correction Step
88
89      id(i) = ans(2,i); %d-axis current
90      iq(i) = ans(3,i); %q-axis current
91
92      %State Correction Function, see StateCorrection.m
93      [xk,ek] = StateCorrection(K,xk_k_1,id(i),iq(i));
94
95      %Covariance Correction Function, see
            CovarianceCorrection.m
96      Pk = CovarianceCorrection(Pk_k_1,K);
97
98      %Process noise (Q), Adaptive Tuning
99      Nq = 1; %Tuning variable
100     alpha1 = (Nq-1)/Nq;
101     omega_k = xk - xk_k_1;
102     omega_k_bar = alpha1*omega_k_1_bar + (1/Nq)*omega_k;
103     delta_Q = (1/(Nq-1))*(omega_k-omega_k_bar)*(omega_k-
            omega_k_bar)') + (1/Nq)*(Pk_k_1 - Fk*Pk_1_k_1*Fk');
104     Q_diag = abs(diag([alpha1*Q_1+delta_Q]));
105     Q = diag([Q_diag(1) Q_diag(2) Q_diag(3) Q_diag(4)]);
106
107     %Measurement noise (R), Adaptive Tuning
108     Nr = 1; %Tuning variable
109     H = [1 0 0 0;0 1 0 0];
110     alpha2 = (Nr-1)/Nr;
111     ek_bar = alpha2*ek_1_bar + (1/Nr)*ek;
112     delta_R = (1/(Nr-1))*(ek-ek_bar)*(ek-ek_bar)' - (1/Nr)*
            H*Pk_k_1*H';
113     R_diag = abs(diag([alpha2*R_1+delta_R]));
114     R = diag([R_diag(1) R_diag(2)]);
115
116     %Save Estimations In x_predict
117     x_predict(:,i) = xk;
118 end
119
120 close all
121 Flux_mean = movmean(x_predict(3,:),2000);
122 Rs_mean = movmean(x_predict(4,:),2000);
123
124 figure(1)
125 plot(ans(1,:), id);
126 hold on
```

```
127 plot(ans(1,:), x_predict(1,:));
128 xlabel('Time [s]');
129 ylabel('Current d-axis [A]');
130 title('Current d-axis');
131 legend('Measured $i_d$', 'Estimated $i_d$', 'Interpreter',
        'latex');
132
133 figure(2)
134 plot(ans(1,:), iq);
135 hold on
136 plot(ans(1,:), x_predict(2,:));
137 xlabel('Time [s]');
138 ylabel('Current q-axis [A]');
139 title('Current q-axis');
140 legend('Measured $i_q$', 'Estimated $i_q$', 'Interpreter',
        'latex');
141
142 figure(3)
143 plot(ans(1,:),x_predict(3,:));
144 hold on
145 plot(ans(1,:),Flux_mean);
146 xlabel('Time [s]');
147 ylabel('Flux [Wb]');
148 title('Estimated Flux');
149 legend('Estimated Flux','Moving Mean', 'Interpreter', '
        latex');
150
151 figure(4)
152 plot(ans(1,:), x_predict(4,:));
153 hold on
154 plot(ans(1,:),Rs_mean);
155 xlabel('Time [s]');
156 ylabel('Resistance [Ohm]');
157 title('Estimated Resistance');
158 legend('Estimated Resistance', 'Moving Mean', 'Interpreter'
        , 'latex');
```

Matlab Code UKF

Main UKF-Code

```
1
2  %Load data from simulation
3  load('parameters.mat')
4
5  %Parameters, ITB-A61
6  n = 4; %Number of states
7  Ls = 0.03264e-3; %Nominal Induction
8  Ts = 1.25e-04; %Time Step
9
10 %Process Noise Matrix, can be tuned
11 Q = diag([1, 1, 1, 1]);
12
13 %Measurement Noise Matrix, can be tuned
14 R = diag([1, 1]);
15
16 %State Transition Function
17 f = @(x,u)[(1-(Ts*x(3))/Ls)*x(1) + Ts*u(3)*x(2) + (Ts*u(1))
       /Ls;
18                 -Ts*u(3)*x(1) + (1-(Ts*x(3))/Ls)*x(2) + (Ts*u
                       (2))/Ls + (-Ts*x(4)*u(3))/Ls;
19                 x(3);
20                 x(4)];
21
22 %Observation Matrix
23 h = @(x)[x(1);
24           x(2)];
25
26 %Initial conditions for state and covariance
27 x0 = [0, 0, 0, 0]';
28 P0 = diag([1, 1, 1, 1]); %Can be tuned
29
30 %Loop variable
31 N = length(time);
32
33 %Memory for storing estimates and measurements
34 xV = zeros(n,N);
35 zV = zeros(2,N);
36
37 %Settings for Butterworth filter applied on id, iq, vd, vq,
       we
38 [b,a] = butter(3,0.05,'low');
39
```

```
%Input parameters:
% vd = d-axis voltage
% vq = q-axis voltage
% we =  electrical angular velocity

%State parameters:
% id = d-axis current
% iq = q-axis current

for k = 1:N

    z(1,k) = id(k);
    z(2,k) = iq(k);

    u(1,k) = vd(k);
    u(2,k) = vq(k);
    u(3,k) = we(k);

    %Use initial conditions for first loop, (i = 1)
    %else, use previous estimate

    if k == 1
        [x,P] = ukf(f,x0,P0,h,z(:,k),Q,R,u(:,k)); %Main UKF-function
    else
        [x,P] = ukf(f,x,P,h,z(:,k),Q,R,u(:,k)); %Main UKF-function
    end

    xV(:,k) = x;
    zV(:,k) = z(:,k);
end

%Settings for moving mean of Flux and Resistance
Rs_mean = movmean(xV(3,:),2000);
Flux_mean = movmean(xV(4,:),2000);

figure(1)
plot(ans(1,:), z(1,:));
hold on
plot(ans(1,:), xV(1,:));
xlabel('Time [s]');
ylabel('Current d-axis [A]');
title('Current d-axis');
legend('Measured $i_d$', 'Estimated $i_d$', 'Interpreter',
    'latex');
```

```
83
84  figure(2)
85  plot(ans(1,:), z(2,:));
86  hold on
87  plot(ans(1,:), xV(2,:));
88  xlabel('Time [s]');
89  ylabel('Current q-axis [A]');
90  title('Current q-axis');
91  legend('Measured $i_q$', 'Estimated $i_q$', 'Interpreter',
        'latex');
92
93
94  figure(3)
95  plot(ans(1,:), xV(3,:));
96  hold on
97  plot(ans(1,:),Rs_mean)
98  xlabel('Time [s]');
99  ylabel('Resistance [Ohm]');
100 title('Estimated Resistance');
101 legend('Estimated Resistance', 'Moving Mean', 'Interpreter'
        , 'latex');
102
103 figure(4)
104 plot(ans(1,:),xV(4,:));
105 hold on
106 plot(ans(1,:),Flux_mean);
107 xlabel('Time [s]');
108 ylabel('Flux [Wb]');
109 title('Estimated Flux');
110 legend('Estimated Flux','Moving Mean', 'Interpreter', '
        latex');
```

UKF-Function

```
function [x,P]=ukf(f,x,P,h,z,Q,R,u)

    L = numel(x); %Number of states
    m = numel(z); %Number of measurements

    %Weights
    alpha=10e-3;
    ki=0;
    beta=2;
    lambda=alpha^2*(L+ki)-L;
    c=L+lambda;
    Wm=[lambda/c 0.5/c+zeros(1,2*L)];
    Wc=Wm;
    Wc(1)=Wc(1)+(1-alpha^2+beta);
    c=sqrt(c);

    %Sigma points, see sigmas.m
    X = sigmas(x,P,c);

    %Unscented transformation of process, see ut.m
    [x1,X1,P1,X2] = ut(f,X,Wm,Wc,L,Q,u);

    %Unscented transformation of measurements, see ut1.m
    [z1,~,P2,Z2] = ut1(h,X1,Wm,Wc,m,R);

    %Transformed cross-covariance
    P12 = X2*diag(Wc)*Z2';

    %Kalman gain
    K = P12*inv(P2);

    %State update
    x = x1+K*(z-z1);

    %Covariance update
    P = P1-K*P2*K';
end
```

Sigma Points

```
function X=sigmas(x,P,c)
    %Sigma points around reference point
    %Inputs:
    %       x: reference point
    %       P: covariance
    %       c: coefficient
    %Output:
    %       X: Sigma points

    A = c*chol(P)'; %chol is the Cholesky decomposition
    Y = x(:,ones(1,numel(x)));
    X = [x Y+A Y-A];
end
```

Unscented Transform

```
function [y,Y,P,Y1]=ut(f,X,Wm,Wc,n,R,u)
    %Unscented Transformation
    %Input:
    %         f: nonlinear map
    %         X: sigma points
    %        Wm: weights for mean
    %        Wc: weights for covraiance
    %         n: numer of outputs of f
    %         R: additive covariance
    %Output:
    %         y: transformed mean
    %         Y: transformed sampling points
    %         P: transformed covariance
    %        Y1: transformed deviations

    L=size(X,2);
    y=zeros(n,1);
    Y=zeros(n,L);
    for k=1:L
        Y(:,k)=f(X(:,k),u);
        y=y+Wm(k)*Y(:,k);
    end
    Y1=Y-y(:,ones(1,L));
    P=Y1*diag(Wc)*Y1'+R;
end
```

Unscented Transform of Measurements

```
function [y,Y,P,Y1]=ut1(f,X,Wm,Wc,n,R)
    %Unscented Transformation
    %Input:
    %         f: nonlinear map
    %         X: sigma points
    %        Wm: weights for mean
    %        Wc: weights for covraiance
    %         n: numer of outputs of f
    %         R: additive covariance
    %Output:
    %         y: transformed mean
    %         Y: transformed sampling points
    %         P: transformed covariance
    %        Y1: transformed deviations

    L=size(X,2);
    y=zeros(n,1);
    Y=zeros(n,L);
    for k=1:L
        Y(:,k)=f(X(:,k));
        y=y+Wm(k)*Y(:,k);
    end
    Y1=Y-y(:,ones(1,L));
    P=Y1*diag(Wc)*Y1'+R;
end
```

A.0.1 Simulation Steps

1. Open the folder where the chosen filter is located.

2. Run the file *motor_parameters.m*. For more details, see Appendix Section Matlab code

3. Open the simulation file *simulation_model.slx*. Choose test case, set the correct simulation time and run the simulation.

4. Run the chosen filter for parameter estimation, appropriate graphs will be generated.

A.0.2 Real World Steps

1. Connect the USB cable from DEWEsoft to the lab computer and start the DEWEsoft application.

2. Connect the tool to the computer with a micro-usb cable.

3. Start a web browser and visit the site for the tool interface, on this site the tightening program can be programmed.
4. In DEWEsoft, browse to the tab "Measurements" and go to the tab called "CAN", make sure that the channels are activated.
5. Arm the measurement function.
6. Place the tool head on the screw joint and press the trigger.
7. Stop the measurement when the tightening is complete.
8. Export the measurements to appropriate format.

Bibliography

[1] CMSIS-DSP. Vector clarke transform, 2022. URL https://www.keil.com/pack/doc/CMSIS/DSP/html/group__clarke.html. [Online; accessed January, 2023].

[2] CMSIS-DSP. Vector park transform, 2022. URL https://www.keil.com/pack/doc/CMSIS/DSP/html/group__park.html. [Online; accessed January, 2023].

[3] NXP Community. Module 2: Pmsm and foc theory, 2017. URL https://statics.teams.cdn.office.net/evergreen-assets/safelinks/1/atp-safelinks.html. [Online; accessed January, 2023].

[4] Fouad Giri. *AC Electric Motors Control: Advanced Design Techniques and Applications*.

[5] Alain Glumineau and Jesús de León Morales. *Sensorless AC Electric Motor Control*.

[6] Fredrik Gustafsson. *Statistical Sensor Fusion*.

[7] Lennart Harnefors. *Control of variable speed drives applied signal processing and control*.

[8] Iyad Hashlamon. A new adaptive extended kalman filter for a class of nonlinear systems. *Journal of Applied and Computational Mechanics*, 6:1–12, 2020.

[9] Co Huynh, Liping Zheng, and Dipjyoti Acharya. Losses in high speed permanent magnet machines used in microturbine applications. *Journal of Engineering for Gas Turbines and Power*, 131:1–5, 2009.

[10] Xiaoliang Jiang, Pindong Sun, and Z.Q.Zhu. Modeling and simulation of parameter identification for pmsm based on ekf. In *2010 Internationale Conference on Computer, Mechatronics, Control and Electronic Engineering*, pages 345–348, Changchun, China, 2010.

[11] Simon J. Julier and Jeffrey K. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, 2004.

[12] Sang-Hoon Kim. *Electric Motor Control : DC, AC, and BLDC Motors*. Joe Hayton, 2017. ISBN 978-0-12-812138-2.

[13] Vadim Kim. How to design 10 khz filter. (using butterworth filter design). URL https://www.egr.msu.edu/classes/ece480/capstone/fall11/group02/web/Documents/How%20to%20Design%2010%20kHz%20filter-Vadim.pdf. [Online; accessed May, 2023].

[14] E.E. Kriezis, T.D. Tsiboukis, S.M. Panas, and J.A Tegopoulos. Eddy current, theory and applications. *Proceedings of the IEEE*, 80:1559–1589, 1992.

[15] Mathworks. Field-oriented control: Developed filed-oriented control algorithms using simulation, 2023. URL https://se.mathworks.com/solutions/electrification/field-oriented-control.html. [Online; accessed March, 2023].

[16] Stephan Meier. Theoretical design of surface-mounted permanent magnet motors with fieldweakening capability, 2002. URL https://www.emetor.com/static/meier2002.pdf. [Online; accessed May, 2023].

[17] Victor M. Moreno and Albert Pigazo. *Kalman Filter Recent Advances and Applications*.

[18] Maria Isabel Ribeiro. Kalman and extended kalman filters: Concept, derivation and properties, 2004.

[19] Marco Taboga. Cholesky decomposition, 2021. URL https://www.statlect.com/matrix-algebra/Cholesky-decomposition. [Online; accessed March, 2023].

[20] Sana Toumi, Mohamed Benbouzid, and Mohamed Faouzi Mimouni. Modeling and simulation of a pmsg-based marine current turbine system with inter-turn faults, 2022. URL https://www.intechopen.com/online-first/1131675. [Online; accessed March, 2023].

[21] Rudolph van der Merwe and Eric A. Wan. The square-root unscented kalman filter for state and parameter-estimation. In *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings*, pages 3461–3464, Salt Lake City, UT, USA, 2001.

[22] Paul Waid and Conrad U.Bruuner. Energy-efficiency policy opportunities for electric motor-driven systems, 2011.

[23] Simon Delamere Wilson, Paul Stewart, and Benjamin P. Taylor. Methods of resistance estimation in permanent magnet synchronous motors for real-time thermal management. *IEEE Transactions on Enegy conversion*, 25(3): 698–707, 2010.