CODEN:LUTEDX/(TEIE-5469)/1-88/(2021)

Industrial Electrical Engineering and Automation

Online Parameter Estimation in Electric Motors

Abizar Sadikot

Division of Industrial Electrical Engineering and Automation
Faculty of Engineering, Lund University

ONLINE PARAMETER ESTIMATION IN ELECTRIC MOTORS

In collaboration with HUSQVARNA AB

Author: Abizar Sadikot

October 25, 2021

Academic Supervisor: Avo Reinap
Industrial Supervisor: Martin Larsén

Industriell elektroteknik och automation
Lunds tekniska högskola

Abstract

The permanent magnet synchronous machine (PMSM) can be used for a wide range of applications. In this case, the motor that is examined is inspired of Husqvarna'shand-held electrical tools where the motor is part of its electrical chainsaw. The idea is to investigate and simulate different algorithms for online parameter estimation for the PMSM. Consequently, one of the algorithms is selected as the best one for this application. The motor parameters that are examined by the online parameter estimators are mainly the stator resistance and the inductance. The stator resistance is linked to the temperature of the motor and the stator inductance is also of interest for online estimation. Although the inductance is not directly linked to any physical quantities, large deviations deviations of the inductance usually indicate faults in the motor. By estimating these parameters, faulty operating points of the motor and critical damage can be stopped. The algorithms researched are; the recursive least square (RLS) with forgetting factor, extended Kalman filter (EKF) and the model reference adaptive system (MRAS). They are explained and applied to a non-salient PMSM, meaning that the stator inductance is not dependent on the rotor position. The simulations are done in Matlab/simulink with ten different test cases, emulating different operating points of the motors to see the convergence capability of the algorithms. The thesis also treats the subject of convergence time, and very lightly, the computational complexity of the algorithms. Moreover, throughout the simulation process, it is more evident that the EKF is the most suitable algorithm for online estimation in this application. This is because it handles all the different test cases well and converges to reasonable values within the margin of error.

Acknowledgments

This thesis is my last assignment at Lunds tekniska högskola (LTH). It is made in cooperation with Husqvarna AB.

Firstly I would like to thank my supervisor at Husqvarna, Martin Larsén for choosing me and believing in my abilities for this project. I also would like to acknowledge his support in each step of the project and his commitment to help. Secondly I would like to thank my supervisor at LTH, Avo Reinap for also helping me throughout the project and giving my helpful guidance in the many steps of this project.

I would also like extend my deepest gratitude to each and every one who in some way have helped me throughout my time at LTH by giving me the help I needed.

Lastly I would like to thank my family; my mom Yasmin Sadikot, dad Mohammed Sadikot, sister Minaz Arsiwala, brother-in-law Huzefa Arsiwala and my niece Aalizah Arsiwala, for always believing in me and always supporting me in any way I needed. Without them, this is not possible.

Contents

List of Figures

List of Tables

Acronyms

Conv - Convergence

Covid-19 - Corona Virus Disease 2019

DC - Direct Current

DSP - Digital Signal Processor

EKF - Extended Kalman Filter

FOC - Field-Oriented Control

MRAS - Model Reference Adaptive System

Nm/Kg - Newton meter per kilogram

OPEA - Online Parameter Estimation Algorithm

PI - Proportional Integral

PMSM - Permanent Magnet Synchronous Motor

PWM - Pulse Width Modulation

RLS - Recursive Least Square

SVM - Space Vector Modulation

ZOH - Zero Order Hold

Nomenclature

i_{abc} - Currents in a, b, c coordinates

$i_{\alpha\beta}$ - Currents in α, β coordinates

i_{dq} - Currents in d, q coordinates

u_{abc} - Voltages in a, b, c coordinates

$u_{\alpha\beta}$ - Voltages in α, β coordinates

u_{dq} - Voltages in d, q coordinates

R_s - Stator resistance

L_s - Stator inductance

ψ_M - Magnetic flux

θ_{el} - Electrical position of the rotor

θ_r - Mechanical position of the rotor

ω_{el} - Electrical speed of the rotor

ω_r - Mechanical speed of the rotor

p - pole-pair number

T_s - Sample time

1 Introduction

This chapter introduces the topics of this thesis and gives some brief background information. The scope of the thesis is also be presented and a short section of the methodology.

1.1 Background

The permanent magnet synchronous motor (PMSM) can be used for a wide range of applications. Important areas for the PMSM are large scale power generation as well as applications for various electrical drives [1]. Another big field for the PMSM is servo applications, where there is a need for very accurate feedback of units such as torque, speed and position.

Two major advantages of a PMSM are capabilities of high torque density and a high efficiency. A typical range for the PMSM is within 1-30 Nm/kg, which may be compared to combustion engines featuring 1-3 Nm/kg. The efficiency of PMSM, if well constructed, can reach 98% efficiency, which compared to combustion engines is significantly higher.

The motor that is examined in this work is inspired of Husqvarna's hand-held electrical tools which is driven by batteries. The Idea is to investigate and simulate different algorithms for online parameter estimation for the PMSM. Husqvarna is a global leader in the field of outdoor power equipment and in this case the motor that is examined will be used in an electrical chainsaw. A previous thesis work has been done with this motor in [28], where the goal was to implement a good strategy for sensorless control of the PMSM. This thesis will further work on that and implement algorithms for online parameter estimation.

1.2 Problem definition

The motor parameters that are examined by the online parameter estimators are mainly the stator resistance and the inductance. The stator resistance is linked to the motor's temperature state, which is why it is helpful to gain information about the parameter online. The inductance is not directly connected to any physical quantity, however great deviations of the inductance indicate faults in the motor.

Parameter variations occur inevitably in electric motors. These parameter variations can be classified in three different categories, namely; static, reversible and irreversible parameter variations.

Static parameter variations occur during the production phase of the motor. In order to handle these variations, a robust control system or a good offline parameter estimation can be used.

Reversible parameter variations occur during operation and do mainly depend on the variations in motor temperature. This in turn affects the motor characteristics and performance. Here a good online parameter estimation can be used to compensate for these variations, meaning that the control system can be used to avoid unwanted operating points.

Irreversible parameter variations occur when the motor is faulty and there are permanent damages. These are important to find as well for diagnostic purposes.

The field of online parameter estimation has been studied extensively and there are numerous reports available on this topic. There are many different approaches that this thesis can take and the questions that need to be examined while studying this field are:

- What online parameter estimation algorithms are available and how easy are they to implement for this PMSM application?
- How well and fast do these algorithms converge to the final result, in terms of accuracy and settling time?
- How feasible is it to implement these algorithms in terms of complexity and computational efficiency?

1.3 Scope

The work is arranged between theoretical preparation and modeling. The first half consists of a literature study, online parameter estimation algorithm (OPEA) description and selection. The second half consists of the model based implementation in Matlab-Simulink and the assessment of the selected OPEA's. There are some limitations that need to be set in order to achieve relevant results and to keep the research suitable for a master's research topic. The main area of this thesis is based on reports and studies previously done on this topic. There are many different parameter estimation methods available but this thesis focuses on three selected methods that are to be simulated. If more estimation methods are found to be relevant, they are to be selectively discarded. This is done for the purpose of comparing different methods with regard to the questions stated in section 1.2 and also to keep the project within the projected time frame.

It is also important to keep the methods relevant for PMSM applications, thus it is important that the chosen methods be suitable for such motors. Moreover, it is necessary for the methods to be able to implemented in MATLAB/Simulink since this is the environment in which simulations of the methods are to be done.

This project will only reach the simulation phase of the selected methods and algorithms. Because of the ongoing Covid-19 pandemic, the access to laboratory and physical motors are limited.

Furthermore, the selected methods will also have to be applicable to field-oriented control(FOC) system since this how the previous implementations have been done.

1.4 Methodology

The majority of this thesis project is based on papers and previous work related to the topic of online parameter estimation in PMSMs. This thesis will also be a continuation of a previous thesis work which treated the subject of sensorless control. Therefore, this thesis will continue with implementing sensorless control system. However, tests on the estimation methods will be done with regular control and sensorless control.

Moreover, the chosen algorithms and methods for online estimation will be simulated in MATLAB/Simulink. In this section, there will also be a continuation of previous Simulink work conducted by the author of the previous thesis work. The simulation method and the work surrounding the simulations will be presented in this section.

The simulation will have to emulate the environment at which the chainsaw operates in. This is done in order to give accurate and proper results of the studied methods.

The results will be presented from the simulink environment with graphs and tables, where the OPEA's will be evaluated in terms of accuracy and settling time. Settling time and accuracy will be the quantified performance indicators of the the OPEA's and the computational complexity will be a soft performance indicator. The goal is to find the best suited algorithm for this application.

1.5 Outline

The thesis is divided into 7 chapters. Chapter 1, introduction, introduces the problem and some background information of the project. Chapter 2, theory, the reader is presented with the theory behind the PMSM and how it is modelled. The focus is on the modelling part since this thesis uses mathematical models of the motor throughout. There is also be a section in the theory which presents the control system.

Chapter 3 is the literature study. This is where the theory and mathematical background of the online estimation methods is presented. The section also presents how well the methods are suited for this application in terms of implementation capability and complexity.

Chapter 4 presents the simulation environment and how the simulations are built. It explains the different implementations of the motor and the different implementations of the parameter estimation methods.

Chapter 5 presents the different results from the estimation methods. Here both sensorless control and regular control are tested.

Chapter 6 discusses and analyzes the results in terms of how well the introductory questions are answered and how this thesis work can be improved upon in the future.

2 Theory

This chapter goes through the PMSM and explains the theory behind the physical and mathematical model of the PMSM. It shows the different transforms used to describe the dynamics of the model and what will be used in the parameter estimation part. It also gives the reader an understanding of the control system that is used in this system and the difference between sensorless and sensor based control scheme.

2.1 Permanent-magnet synchronous motor

2.1.1 General PMSM & Drive

The basic design of a PMSM usually consists of a rotor which has permanent magnets in it. The magnets create a magnetic flux which is the fundamental principle that contributes to the rotational movement of the motor. The stator usually is an iron core with copper windings that create a rotational magnetic field. It is this rotating magnetic field that is in sync with the rotor's magnetic flux, hence the name synchronous. The PMSM is usually driven by a three phase system. However, this three phase representation as a, b, c-currents and -voltages can be transformed using Clarke and Park transformations. This enables easier control of the motor and simpler mathematical model[9]. The control system used in this work is field-oriented control (FOC), which will be further explained in sections down below. There are two categories of the PMSM, the salient one with interior magnets that results in rotor saliency and reluctance, and the non-salient one with surface mounted magnets that does not result in saliency and reluctance[10]. In this thesis the PMSM that will be studied is a non-salient one, which means that the stator inductance is constant and not dependent on the rotor position because the air gap is uniformly even.

2.1.2 Three phase system

The permanent magnet synchronous motor is a synchronous motor driven by a three phase system, which creates the rotational magnetic field in the stator. This system is characterized by having three identical sinusoidal voltage signals separated each by 120 degrees phase shift. A three phase system is common in electrical applications due to the rotational magnetic field it creates along with the constant torque it produces. This is related to the fact that the active power produced by a three phase system is also constant[2]. Another unique property of a three phase system is that the sum of the instantaneous values are zero, this can be seen visually in the 2.1.

$$i_a + i_b + i_c = 0 \tag{2.1}$$

$$u_a + u_b + u_c = 0 \tag{2.2}$$

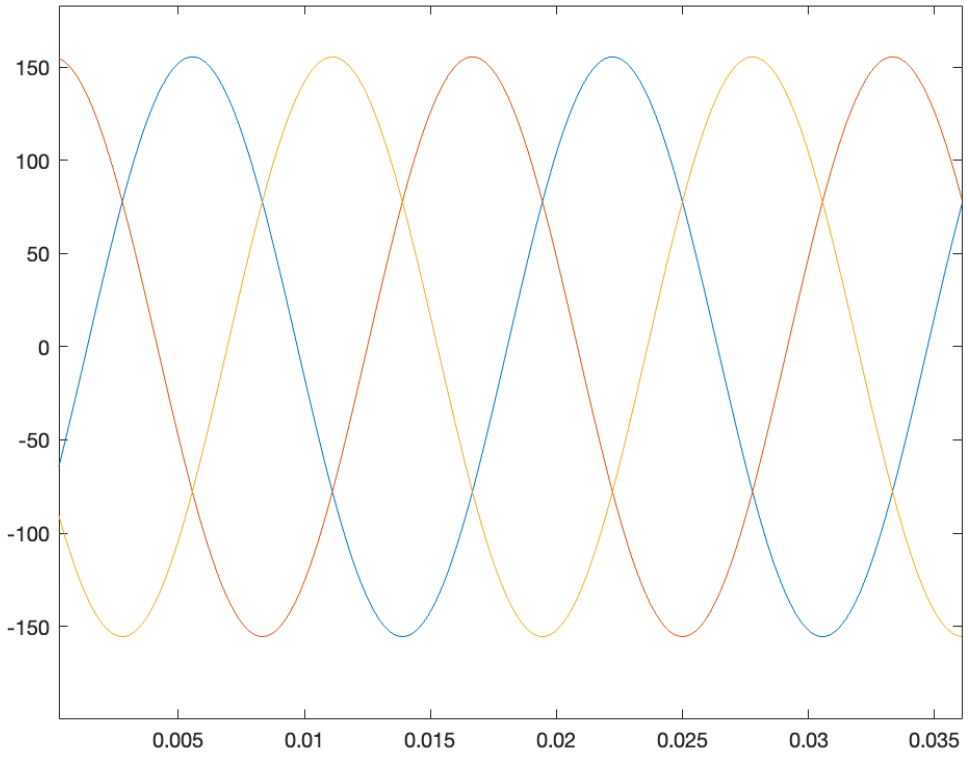


Figure 2.1: Visual representation of three phase system

2.1.3 Mathematical model of PMSM

model in *abc*-frame

The mathematical model of the PMSM is not unlike any other electrical motor. The three phases that drive the motor can be individually represented by the equations below[28].

$$u_a = R_s i_a + \frac{d}{dt}(L_s i_a + \psi_{Ma}) \tag{2.3}$$

$$u_b = R_s i_b + \frac{d}{dt}(L_s i_b + \psi_{Mb}) \tag{2.4}$$

$$u_c = R_s i_c + \frac{d}{dt}(L_s i_c + \psi_{Mc}) \tag{2.5}$$

Here u_{abc} represents the stator voltages, R_s is the stator resistance, i_x represents the different phase currents, L_s is the stator inductance and ψ_{Mabc} is the contribution of the rotor magnets to the stator flux linkage. The first term $R_s i_x$ of the phase equations

are the voltage drop in the stator winding resistance due to the stator currents. The second term represents the derivative of the flux on the stator windings. The $L_s i_x$ represents the flux given by the inductance and the derivative ψ_{Mabc}-term represents the back-EMF induced by the rotor magnets. This flux is dependent on the position of the rotor and is sinusoidal in its shape. Thereby, the equations can be rewritten in the form given below[28].

$$u_a = R_s i_a + \frac{d}{dt}(L_s i_a) + \omega_{el}\frac{d}{d\theta_{el}}\psi_{Ma} \tag{2.6}$$

$$u_b = R_s i_b + \frac{d}{dt}(L_s i_b) + \omega_{el}\frac{d}{d\theta_{el}}\psi_{Mb} \tag{2.7}$$

$$u_c = R_s i_c + \frac{d}{dt}(L_s i_c) + \omega_{el}\frac{d}{d\theta_{el}}\psi_{Mc} \tag{2.8}$$

Here ω_{el} is the electrical rotational speed of the rotor and θ_{el} is the electrical position of the rotor. The relationship between electrical and mechanical speed and position is given below where p is the pole-pair number.

$$\omega_{el} = p\omega_r \tag{2.9}$$

$$\theta_{el} = p\theta_r \tag{2.10}$$

model in $\alpha\beta$-frame

The three phase representation can be transformed into two equivalent axes using Clarke transform, going from *abc*-axes to $\alpha\beta$-axes. This is done to reduce the complexity of the mathematical model and to simplify the implementation of FOC. The frame of reference is still the stator reference, meaning the mathematical model has a stationary reference frame with a rotating magnetic field. The Clarke transform is defined as stated below[20].

$$\begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} = \sqrt{\frac{2}{3}} \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} \tag{2.11}$$

The Clarke transformation gives the two equations.

$$u_\alpha = R_s i_\alpha + L_s \frac{d}{dt} i_a - \psi_{M\alpha} sin(\theta_{el})\omega_{el} \tag{2.12}$$

$$u_\beta = R_s i_\beta + L_s \frac{d}{dt} i_a - \psi_{M\beta} cos(\theta_{el})\omega_{el} \tag{2.13}$$

model in dq-frame

The $\alpha\beta$-frame is in the stator reference frame. The benefit of the park transformation is that it's given in dq-axes, which are rigidly coupled to the rotor, thus they rotate with the rotor. The d-axis is the *direct* axis which is aligned with the maximum of the rotor flux. The q-axis is the *quadrature* axis. This one is 90 degrees rotated in the positive direction with respect to the electrical position. This transformation is dependent on the angle between the d-axis and the α-axis of the stator. The transformation is defined as given below[5].

$$\begin{bmatrix} i_d \\ i_q \end{bmatrix} = \begin{bmatrix} cos(\theta_{el}) & sin(\theta_{el}) \\ -sin(\theta_{el}) & cos(\theta_{el}) \end{bmatrix} \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} \tag{2.14}$$

By rewriting the equations given in 2.12 and 2.13 with the park transformation and then to rewrite them as dynamic differential equations, they can be described by state-space equations. Differential equations are given below.

$$\frac{d}{dt} i_d = -\frac{R_s}{L_s} i_d + \omega_{el} i_q + \frac{u_d}{L_s} \tag{2.15}$$

$$\frac{d}{dt} i_q = -\frac{R_s}{L_s} i_q - \omega_{el} i_d + \frac{u_q - \omega_{el}\psi_M}{L_s} \tag{2.16}$$

The differential equations are rewritten in as state space equations in 2.18 with the form as given in 2.17

$$\begin{aligned} \dot{x}(t) &= A(x) + B(u) \\ y(t) &= C(x) \end{aligned} \tag{2.17}$$

$$\underbrace{\begin{bmatrix} \dot{i}_d \\ i_q \end{bmatrix}}_{\dot{x}} = \underbrace{\begin{bmatrix} -\frac{R_s}{L_s} & \omega_{el} \\ -\omega_{el} & -\frac{R_s}{L_s} \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} i_d \\ i_d \end{bmatrix}}_{x} + \underbrace{\begin{bmatrix} \frac{1}{L_s} & 0 \\ 0 & \frac{1}{L_s} \end{bmatrix}}_{B} \underbrace{\begin{bmatrix} u_d \\ u_q - \omega_{el}\psi_M \end{bmatrix}}_{u} \tag{2.18}$$

$$\underbrace{\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}}_{y} = \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}_{C} \underbrace{\begin{bmatrix} i_d \\ i_q \end{bmatrix}}_{x} \tag{2.19}$$

Discretization of PMSM model

In order to use the PMSM model in the simulation or for example in a DSP (digital signal processor) it is necessary to discretize the model. The estimation methods that

will be used later in the thesis are discrete models and in some cases, it is more useful to use the signals from the motor model instead of the actual motor signals.

To begin with, the state space model can be ZOH (zero order hold) sampled with sampling period h[29]. This gives the discretized state-space model as 2.20.

$$\begin{aligned} \dot{x}(kh+h) &= A_d(kh) + B_d(kh) \\ y(kh+h) &= C_d(kh) \end{aligned} \tag{2.20}$$

Here the discretized state space matrices are defined as followed.

$$A_d = e^{Ah} \tag{2.21}$$

$$B_d = \int_0^h e^{As} ds B \tag{2.22}$$

$$C_d = C \tag{2.23}$$

The computation of the discretized state-space matrices can be simplified by using the matrix Ψ as given in 2.24.

$$\Psi = \int_0^h e^{As} ds \tag{2.24}$$

The matrices are then calculated as 2.25 and 2.26

$$A_d = I + A\Psi \tag{2.25}$$

$$B_d = \Psi B \tag{2.26}$$

By using Euler forward difference with one step time shifted backwards and the discretized state-space matrices. The complete discretized state-space model of the PMSM is given in 2.27 [20].

$$\underbrace{\begin{bmatrix} i_d(k) \\ i_q(k) \end{bmatrix}}_{x(k)} = \underbrace{\begin{bmatrix} -\frac{R_s}{L_s}h + 1 & \omega_{el}h \\ -\omega_{el}h & -\frac{R_s}{L_s}h + 1 \end{bmatrix}}_{A_d} \underbrace{\begin{bmatrix} i_d(k-1) \\ i_d(k-1) \end{bmatrix}}_{x(k-1)} + \underbrace{\begin{bmatrix} \frac{1}{L_s}h & 0 \\ 0 & \frac{1}{L_s}h \end{bmatrix}}_{Bd} \underbrace{\begin{bmatrix} u_d(k-1) \\ (u_q - \omega_{el}\psi_M)(k-1) \end{bmatrix}}_{\bar{u}(k-1)} \tag{2.27}$$

2.1.4 Reference motor

The motor used in this thesis have the specifications and parameter values as given in table 2.1[28].

Table 2.1: Table of specifications of the motor

Nr of poles in the rotor	14
Nr of slots in the stator	12
Rated power output	1,2 kW
Rated torque output	1,5 Nm
Rated speed	8500 rpm
No load speed	11000 rpm
Rated DC link voltage	30 V
Maximum current	60 A

2.2 Field-oriented Control

2.2.1 FOC implemented with sensors

The control system that is applied in this motor is called field-oriented control. The idea behind this control strategy is that it controls the motor current in the dq-frame. The benefit of this is that the signals behave like a DC machine. This means that the signals provided by the control system are DC signals. Another great benefit of this control strategy is that the d- and q-axis current can be controlled individually[6]. The d-axis current is proportional to the flux and the q-axis is proportional to the torque, which consequently means that those two units can be controlled individually. Ideally, if the torque is to be maximized with reference to the total power, the current in the d-axis is to be zero for the reference machine[6].

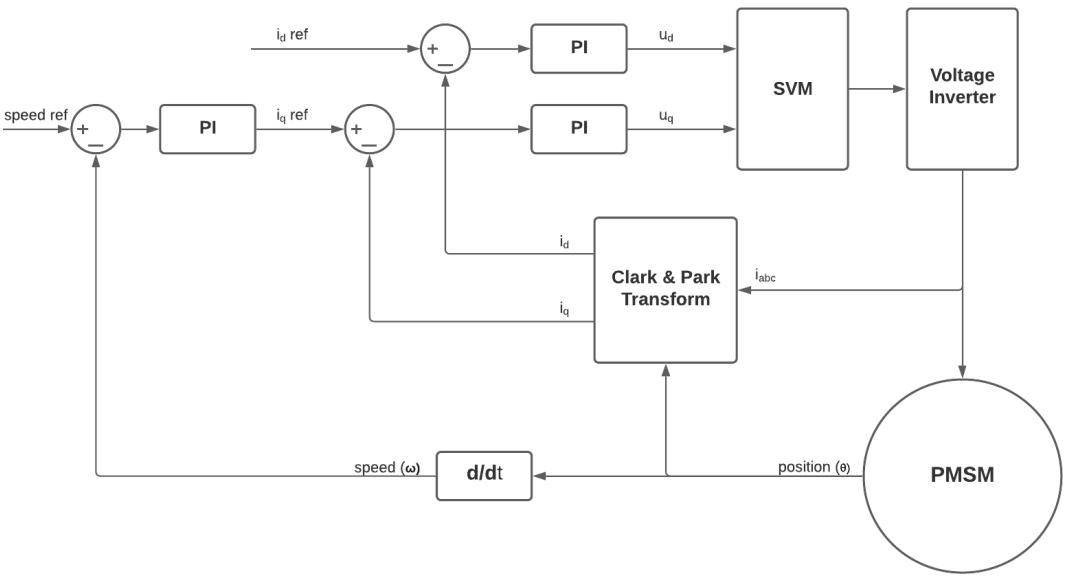


Figure 2.2: Field-oriented control scheme implemented using position sensor

Figure 2.2 shows the field-oriented control scheme visually. From the PMSM, the

position and speed is retrieved by sensors. This is then fed back to two PI-controllers. The speed controller are two cascaded PI-controllers because the speed reference need to produce a reference value for the i_q current. The dq currents are then fed through the PI-controllers to produce the dq voltages. The voltages are then used to produce the required signals to the three phase PMSM. The space vector modulation (SVM) produces PWM signals to the voltage inverter, which in turn sends the right three phase signal to the motor.The three phase currents are then Clarke and Park transformed back into dq currents, and the feedback loop is complete. It is also visible on the block scheme that dq current are controlled separately. Ideally, as stated, the d-current is zero. In cases when there is need for more speed, the d-current can be controlled to a value different from zero but this will come at the cost of lowering torque. This is called field weakening control [1].

2.2.2 Sensorless control

In this thesis, the control scheme is not using sensor to retrieve the position and speed. Instead, an estimation method is used to achieve those signals. It is visible in 2.3, where both speed and position are retrieved by only using the current and voltage signals. This thesis deploys OPEA's on both sensorless and position feedback FOC schemes. This is to get a better basis when evaluating the different methods. The control scheme used in this case uses model reference adaptive system (MRAS) when estimating speed and position.

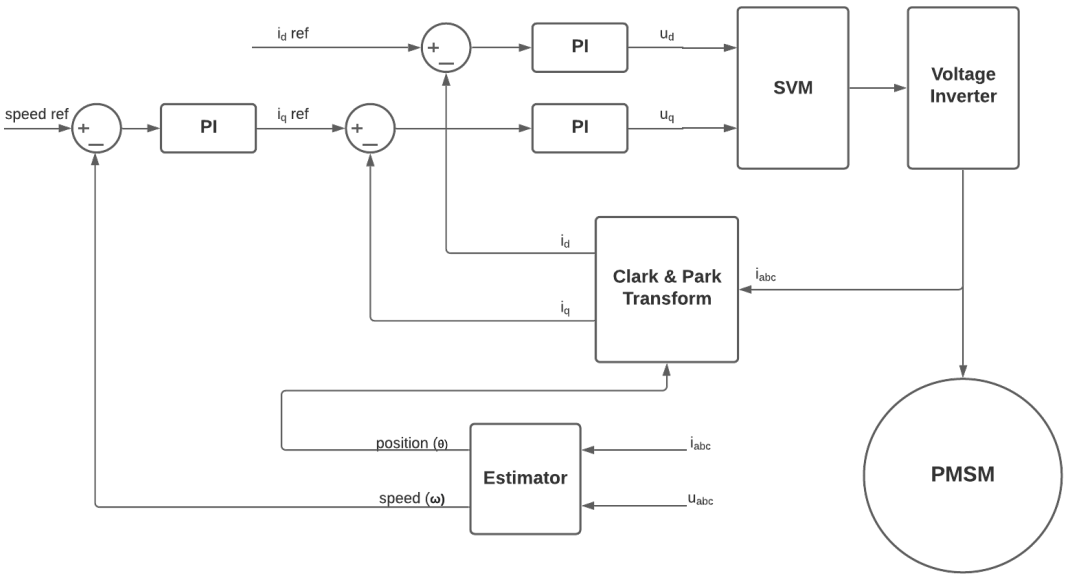


Figure 2.3: Field-oriented control scheme implemented with speed and position estimator (Sensorless control)

3 Literature study

In this chapter the methods for online parameter estimation are presented. The theoretical background derived from papers that have been published are also presented, as well as the implementation capabilities in the the PMSM. There are many algorithms that have been published and studied, these are evaluated before the simulation step. The requirements presented in the introduction are considered when analyzing the methods. Furthermore, only methods that are applicable and feasible to implement based on the theoretical data available will be simulated.

3.1 Extended Kalman Filter - EKF

The extended kalman filter (EKF) is an extension of the kalman filter used in many applications[28]. The regular kalman filter is a minimum variance estimator, given that the model is linear and that noise is additive[23]. The extended part includes non-linear dynamic systems which is possible by linearizing the non-linear system using first order Taylor expansion[23]. The jacobian of the non linear system function is used for linearization in the current time step[22]. The EKF is widely used in PMSM applications for sensorless control, obtaining information from noisy environments and for parameter estimation[3]. This work will focus on the latter.

3.1.1 Mathematical model of the EKF

The extended kalman filter uses non-linear systems in the state space form given below:

$$\begin{aligned} \dot{x}(t) &= f(x(t), u(t)) + w(t) \\ y(t) &= h(x(t)) + v(t) \end{aligned} \tag{3.1}$$

$x(t)$ is the state vector, $u(t)$ is the input vector and $w(t)$ and $v(t)$ are both zero-mean white Gaussian noise [13]. The input vector and the state vectors will be explained further in a later section.

The EKF algorithm is divided into two main parts. The prediction step and the correction step[14]. The mathematical equations for each step are given below. The equations presented give the algorithm in discrete time. This is also how it is implemented in the real application and the simulations as well, which will be treated in later chapters.

- The prediction step:

$$\hat{x}_{k|k-1} = \hat{x}_{k-1|k-1} + (f(\hat{x}_{k-1|k-1}, u_k))T_s \tag{3.2}$$

$$P_{k|k-1} = P_{k-1|k-1} + (F_{k-1}P_{k-1|k-1} + P_{k-1|k-1}F_{k-1}^T)T_s + Q \tag{3.3}$$

  In this step the algorithm predicts the next state ($\hat{x}_{k|k-1}$) based on the previous state ($\hat{x}_{k-1|k-1}$) and the current input (u_k) into the system. T_s is the discrete time step. Also, the P matrix, which is the covariance matrix, is calculated. The Q matrix is the noise covariance, R is the disturbance covariance and F is the jacobian of the non-linear state-space function, which is defined in 3.7. The equations denoted with $k|k-1$ means the term at time step k with information retrieved from time step $k-1$.

- The correction step:

$$K_k = P_{k|k-1}H^T(HP_{k|k-1}H^T + R)^{-1} \tag{3.4}$$

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(y_k - H\hat{x}_{k|k-1}) \tag{3.5}$$

$$P_{k|k} = P_{k|k-1} - K_kHP_{k|k-1} \tag{3.6}$$

  In this step the covaraiance matrix ($P_{k|k}$) along with the state estimation ($\hat{x}_{k|k}$) is corrected using feedback from the system. The kalman gain (K_k) is also calculated in this step. The *jacobians* used in the prediction and correction step are defined as given below[31].

$$F(x(t)) = \frac{\partial \hat{x}}{\partial x}|_{x=x(t)} \tag{3.7}$$

$$H(x(t)) = \frac{\partial h}{\partial x}|_{x=x(t)} \tag{3.8}$$

3.1.2 EKF implementation

The EKF state-space equation should be given in form presented in 3.1. This means that the state vector is is defined as $x(t) = [id, iq, a, b]^T$, where id and iq are the currents given from the motor, $a = R/L_s$ and $b = 1/L_s$ [3].

This yields the non linear function as:

$$f(x(t), u(t)) = \begin{bmatrix} -ai_d + \omega i_q + bu_d \\ -ai_q - \omega i_d + b(u_d - \psi_M\omega) \\ 0 \\ 0 \end{bmatrix} \tag{3.9}$$

The assumption is made that the parameters a and b are changing slowly, hence the function value of 0[3]. The output is given by $y(t)$ and the output matrix is given below.

$$y(t) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} x(t) \tag{3.10}$$

Further, to implement the EKF it is important to set the initial conditions of the state variables correct[15]. In this case it will be given known values from the datasheet. The resistance and the inductance is specified for the motor. The currents are set to 0. Moreover, it is important to specify the covariance matrices Q, R and P_0. They are set by a trial-and-error method since there is no proven good method for setting them[3].

3.1.3 Applicability

The extended kalman filter is a very good method for parameter estimation, especially in noisy environments. Further benefits of the extended kalman filter is that it is very robust and is not heavily influenced by parameter variations[31]. It has been used in PMSM applications in many projects and could be a candidate to investigate further in simulation. One of the drawbacks of implementing the EKF is the high computational cost of the algorithm[21]. The process, which is mathematically presented in the previous section, involves many matrix calculations and especially computationally heavy matrix inversion. This leads to higher demand on the processor which might be a disadvantage. Another factor to take into consideration is the trial-and-error method for defining the covariance matrices. These need to be individually tested for each motor it is implemented in [31]. There is, as stated before, no obvious way of deciding these and the choice of the matrices affect the convergence feature of the entire method. It is a very time consuming task and should be a factor to take into account.

3.2 Recursive least square - RLS

The least square is a method where the parameters of a linear model are estimated in such way that the square of the difference between the observed values and the computed values is minimized by a loss function. The idea is to have the model written in linear regression form, see section further down. The loss function is given by 3.11 [25].

$$V(\hat{\theta}, n) = \frac{1}{2} \sum_{i=1}^{n} (y(i) - \varphi^T(i)\hat{\theta})^2 \tag{3.11}$$

where θ is the parameter vector, n is the size of the data array. $y(i)$ are the observed values and $\varphi^T(i)\hat{\theta}$ are the computed values.

The recursive least square algorithm is modified for the use of online estimation[19]. The advantages of the RLS are that it is simple in the construction for example

compared to the EKF. The RLS algorithm is not that computationally complex either, which is to prefer in this application[30]. The overall structure in RLS is to update the value of the estimated parameters recursively based on the error between the estimated output and the actual output of the model[11]. The limitation of RLS is that the parameters are considered constant, however in this case the parameters such as stator resistance and inductance could be time-varying. In this case the RLS with forgetting factor is better. The idea is to introduce a forgetting factor that gives more weight to newer data and lower weight to old data[20]. This is to keep newer data more relevant for parameter estimation. The loss function is then updated with the factor λ^{n-i}, where $0 < \lambda \leq 1$.The updated loss function is then given in 3.12 [19]

$$V(\hat{\theta}, n) = \frac{1}{2}\sum_{i=1}^{n} \lambda^{n-i}(y(i) - \varphi^T(i)\hat{\theta})^2 \tag{3.12}$$

3.2.1 Mathematical model of the RLS with forgetting factor

The recursive least square method with forgetting factor is given by minimizing the the loss function 3.12 with respect to $\hat{\theta}$[19].

The RLS with forgetting factor is given in 3.13[19].

$$\hat{\theta}(n) = \hat{\theta}(n-1) + K(n)(y(i) - \varphi^T(i)\hat{\theta}(n-1)) \tag{3.13}$$

$$K(n) = P(n)\varphi(n) = P(n-1)\varphi(n)(\lambda I - \varphi^T(n)P(n-1)\varphi(n))^{-1} \tag{3.14}$$

$$P(n) = \frac{P(n-1)(I - K(n)\varphi^T(n))}{\lambda} \tag{3.15}$$

$\hat{\theta}(n)$ is the current parameter vector that is estimated, $K(n)$ is the update gain and $P(n)$ is the covariance matrix.

3.2.2 RLS Implementation

in order to implement the RLS with forgetting factor for the PMSM, the discretized state-space model in 2.27 must be rewritten in linear regression form. The linear regression form is given by:

$$y(n) = \varphi(n)^T\theta(n) \tag{3.16}$$

where $y(n)$ is the measured current output. $\varphi(n)^T$ is the regressor vector. The regressor vector contains information from previous time step. In this case it represents the currents and voltages from PMSM from time step $t-1$. $\theta(n)$ is the parameter vector, or in this case the parameter matrix that needs to be determined with the RLS algorithm[20].

By substituting equation 3.16 with the discretized state space model parameters, the following equation is obtained as given in 3.17

$$\begin{bmatrix} i_d(n) & i_q(n) \end{bmatrix} = \begin{bmatrix} i_d(n-1) & i_q(n-1) & u_d(n-1) & \bar{u_q}(n-1) \end{bmatrix} \begin{bmatrix} a_{11} & a_{21} \\ a_{12} & a_{22} \\ b_{11} & b_{21} \\ b_{12} & b_{22} \end{bmatrix} \tag{3.17}$$

The RLS will estimates the parameters in the last matrix in 3.17. The motor parameters are obtained as followed, with $T_s = h$ being the sampling time [20].

$$R_s = \frac{2 - a_{11} - a_{22}}{b_{11} - b_{22}} \tag{3.18}$$

$$L_s = \frac{T_s}{b_{11}} = \frac{T_s}{b_{22}} \tag{3.19}$$

3.2.3 Applicability

The RLS algorithm with forgetting factor is widely used in applications where parameters are to be estimated. The relative simple calculations with simple matrix algebra (compared to the EKF) and proven exponential convergence rate for constant parameters is also a benefit of the RLS[25]. With the case of the PMSM used in this situation, it can be seen in previous section that the mathematical models work for the RLS algorithm. The rewriting of the PMSM in linear regression form allows for the parameters to be estimated.

Drawbacks of the RLS with forgetting factor is the potential unwanted increase in the covariance matrix. The covariance matrix at each time-step is divided by the forgetting factor. In the normal RLS algorithm, the matrix P goes towards zero as the parameters are estimated. However in this case, with the PMSM, the rate at which the covaraiance matrix goes to zero is much slower because the forgetting factor usually is < 1[25]. Another drawback in the RLS with forgetting factor, is the potential lack of excitation in the signals that are received by the algorithm[7]. The excitation of signals simply mean the availability of new dynamic information from the signal. If there is poor excitation, the "old information" is discarded because of the forgetting factor and lack of new information in the signal could lead to an exponential increase in the covariance matrix. This could result in a very sensitive estimator, which leads to extremely wrong computational and numerical error. This "error" is called estimator wind-up[19].

There are some suggested methods to avoid estimator wind-up. Conditional updating is one suggested method, meaning that the RLS algorithm only updates the values when there is enough excitation in the signal. Although this is a simple method to avoid wind-up problem the condition at which the RLS updates its values is very sensitive. If the condition is too strict the RLS algorithm will give poor estimation and bigger error compared to actual parameter values. On the other hand, if the conditions are too soft, there can be, again, wind-up problems. Another suggested

method is to use boundaries on the P matrix. This means that there should be a constant value on the sum of the diagonal elements of the matrix. The drawback to this method is that there could be errors that are too big in the estimation and also that the covariance matrix never goes towards zero[19]. There is also a method of resetting the covariance matrix when there is low excitation in the signal[25].

The capabilities of the RLS algorithm with forgetting factor is one that should be investigated further in the simulations for this PMSM as it is a simpler method than the one suggested by the EKF. And if the potential for implementation is rather simple as well.

3.3 Model reference adaptive system - MRAS

The idea with model reference adaptive system (MRAS) is to have two independent models that contain the same parameters that are to be estimated. These two models output the same vector. The error between these two vectors is then fed back to an adaptive system which adjusts parameters in the adjustable model in order to minimize the error[12]. Ideally the error tends towards zero, which indicates a perfect adjusted model in terms of the reference model, see figure 3.1. There are many ways the adjustable model and the adaption scheme can be constructed. In this case it is simplest to build the models based on the state space models already given in previous sections. The adaption mechanism will be presented in a section further down.

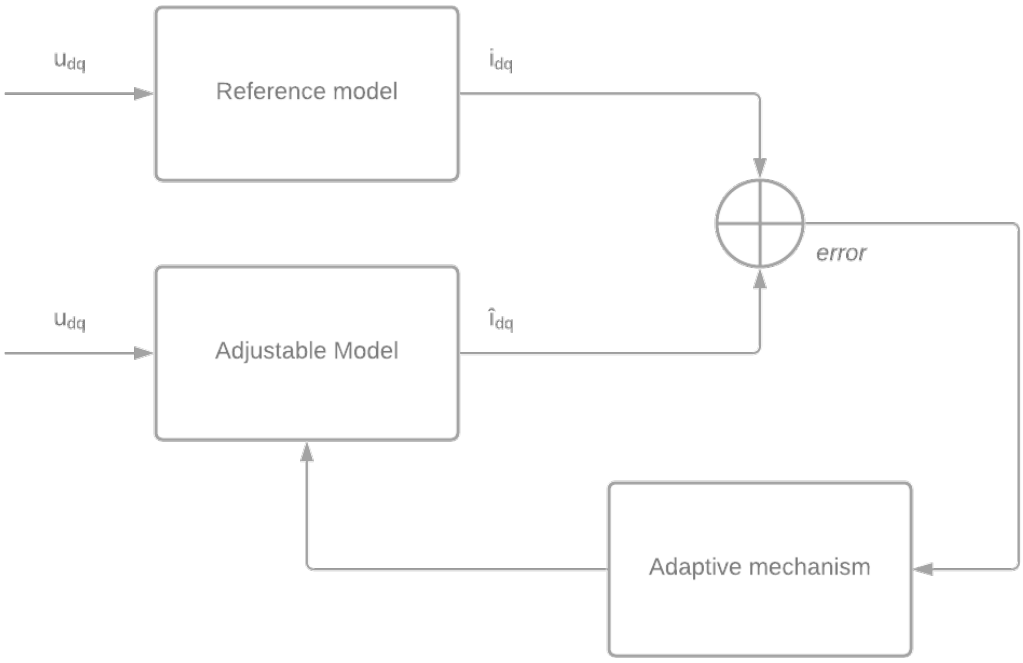


Figure 3.1: MRAS scheme, where i_{dq} are the currents from the motor and $\hat{i}_{dq}$ are the estimated currents given from the motor model

3.3.1 Mathematical model of MRAS & implementation

Reference model and adjustable model

The mathematical design of the MRAS can be done in many different ways. In this case, it is done by having the reference model be the actual motor. From the motor, the *dq* currents and voltages are available. The adjustable model can then be designed as a similar motor as the one used in the reference model[16]. The model equations are given in both, differential equations (see 2.15 and 2.16), and as state-space equation (see 2.18). The implementation is straight forward and does not require much computational power from the motor model. Thereby the adjustable model for parameter estimation is given by equation 3.20. The parameters denoted with $\hat{\gamma}(hat)$ are parameters estimated by the model.

$$\begin{bmatrix} \dot{\hat{i}_d} \\ \dot{\hat{i}_q} \end{bmatrix} = \begin{bmatrix} -\frac{\hat{R}_s}{\hat{L}_s} & \omega_{el} \\ -\omega_{el} & -\frac{\hat{R}_s}{\hat{L}_s} \end{bmatrix} \begin{bmatrix} \hat{i_d} \\ \hat{i_q} \end{bmatrix} + \begin{bmatrix} \frac{1}{\hat{L}_s} & 0 \\ 0 & \frac{1}{\hat{L}_s} \end{bmatrix} \begin{bmatrix} u_d \\ u_q - \omega_{el}\psi_M \end{bmatrix} \tag{3.20}$$

Adaptive mechanism

The adaptive scheme can be constructed in many different ways. The important aspects while designing the adaptive mechanism in this case are the convergence capability, the convergence time and the computational complexity of the mechanism.

The error dynamics $\dot{e}$ of the motor can be described by the equations below, see 3.21. This is achieved by taking the equations given in 2.18 and subtracting it with 3.20[18].

$$\begin{bmatrix} \dot{e_d} \\ e_q \end{bmatrix} = \begin{bmatrix} -\frac{\hat{R}_s}{L_s} & \omega_{el} \\ -\omega_{el} & -\frac{\hat{R}_s}{L_s} \end{bmatrix} \begin{bmatrix} e_d \\ e_d \end{bmatrix} + \begin{bmatrix} -\frac{i_d}{L_s} \\ -\frac{i_q}{L_s} \end{bmatrix} \left[R - \hat{R}\right] \tag{3.21}$$

Here again, the terms denoted with $\hat{\gamma}(hat)$ are estimated parameters from the adjustable motor model. With further maths and algebraic calculations, the adaptive law, based on the Popov's hyper stability is given below, see 3.22 [16].

$$\frac{\hat{R}_s}{\hat{L}_s} = \frac{R_s}{L_s} - K_{Ri}\int_0^t (e_d\hat{i_d} + e_q\hat{i_q}) - K_{Rp}(e_d\hat{i_d} + e_q\hat{i_q}) \tag{3.22}$$

In the same way the adaptive law for the stator inductance can be derived, see equation 3.23[16].

$$\frac{1}{\hat{L}_s} = \frac{1}{L_s} + K_{Li}\int_0^t (u_d e_d + u_q e_q) + K_{Lp}(u_d e_d + u_q e_q) \tag{3.23}$$

3.3.2 Applicability

One of the main benefits of the MRAS estimation method is that it is very simple to implement. Both in terms of implementation of the algorithm and the computational complexity[12]. The algorithm only uses simple differential equations when designing the motor model in the adjustable model and the adaptive law based on Popov's stability theorem uses a simple PI-controller.

There are other adaptive laws that can be implemented. One common one is to use an adaptive law based on Lyapunov stability. The adaptive law is very similar to the one used in Popov's case, however the Popov's case is more flexible in performance, thus it is more desirable. In [16], the Lyapunov method does not perform as well as the Popov method in the experimental verification. The convergence time is much slower, which in the case of the PMSM is not wanted.

The drawbacks of the MRAS estimation method is that it does not work very well with parameter variations [8]. The adaptive laws for the parameters are dependent on the initial condition of the parameters to be estimated [27][28]. If the parameters are time varying, the MRAS method may have difficulty in presenting accurate results.

3.4 Other methods

There exists other methods as well, which are proven to work as online parameter estimation methods in PMSM applications. One of them is signal injection method. The idea of this method is to inject known signals in to the motor, which will produce new sets of steady state equations for estimating parameters[4]. For example, the injected signal could be different d-current signals which will produce different sets of d-axis equations as given in 2.15.

The injected signal could be in many forms, sinusoidal, square or even triangle waves. The reasoning is to produce more information of the system from the injected signal. When estimating the parameters, most of the methods makes use of the RLS algorithm. The same on as presented in this paper. [17] uses a sinusoidal injection method which proves to be successful in estimating parameters. [4] uses different dc values of i_d.

This method builds upon the one presented in the RLS section, and will thus not require much extra work in the simulation process. Therefore it will also be tested while simulating the RLS method. The drawbacks of choosing signal injection method is that the injected signal should not affect the motor and its operating point to such extent that it does not perform as intended[17]. The choice of the injected d-axis current should minimize the torque change as much as possible but at the same time give a good enough and fast enough parameter estimation.

In terms of computational complexity and estimation capability, this method has been proven to work in certain PMSM application with good result[32]. However, it needs to be analyzed whether it is suitable for this chainsaw application or not.

Another estimation method that have been tested to work with online parameter estimation is neural network. Like all the other methods, this one also has many different implementation variations. The one used in [26] is a variant of the MRAS estimation method. Here, the adjustable model is a neural network instead of the state space form of the PMSM. The error dynamics is also different, where the model discussed in previous section uses the current error in dq frame, this one uses the magnetic flux. Another type of neural network is the back propagation neural network.

3.5 Discussion

The studied methods have all been proven to work in estimating the parameters of a PMSM online. When it comes to the EKF it has very good capabilities when it comes to noisy environments and is very robust. Although that comes with the cost of high computational cost and time consuming tuning of the parameters this has been proven to work in different types of operating points and is a good candidate for the chainsaw application.

With the RLS algorithm, there have been many proven successful implementations. The points that are of advantage for this method is the simpler computational complexity and easy implementation. Things to be considered for this method is that there need to sufficient excitation in the signal for the algorithm to not have ill convergence. Signal injection is a method that could solve the problem with low excitation, thereby this can also be tested simultaneously.

Further, the MRAS method is a much simpler algorithm both in terms of implementation and computational complexity. Thus it is desirable to test this method. Important factors to analyze while testing this is how it estimates parameters while there are parameter variations and unusual operating points that the chainsaw might be in.

To summarize, all previous mentioned methods of online parameter estimation will be tested in MATLAB/simulink environment. That is because they are different in how computationally demanding they are, and the work in different environments. Best case scenario would be to have the algorithm with the least demand on the processing unit to be converging to the correct parameter values, In this case, that would be the MRAS method. Moreover, it is important to try other methods where MRAS might fail. Further, the neural network will not be tested for this purpose as it is more demanding in implementation and it is not desirable in this case for offline learning.

4 Simulation

This chapter specifies and presents the values used in the different simulations. The motor, the control system and the sensorless control system. This gives the reader a better understanding of how each part is connected. Moreover, each estimation method is explained and how they are implemented. Since this thesis builds upon a previous work, it refers much of the simulation work to that thesis. Also, the emphasis is on the estimation methods.

4.1 Simulink overview

Figure 4.1 shows the overview of the simulation work space. The blue one is the FOC system and the green one below is the MRAS speed and position estimator. Further the top grey box is the motor model and the red one has the three different OPEA's that are given in chapter 3. Moreover, the solver is set to auto and the maximum step size is set to $5e-05$.

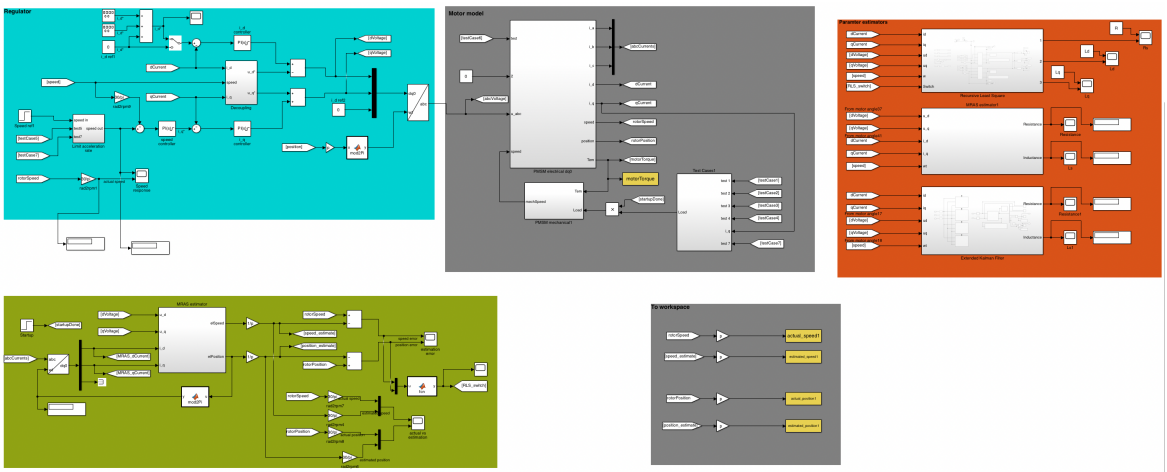


Figure 4.1: overview of the simulink environment

4.2 Permanent magnet synchronous motor

To begin with, the motor is explained in the simulink environment. The PMSM uses the differential equations given in 2.15 and 2.16. In the motor block the differential equations are implemented with blocks with the parameters of the motor defined as given below in table 4.1.

Table 4.1: Motor parameters

Stator Resistance R_s	0.0087 Ω
Stator inductance L_s	1.9e-5 H
Magnetic flux linkage ψ_M	0.0024 Weber
Pole pair nr	7

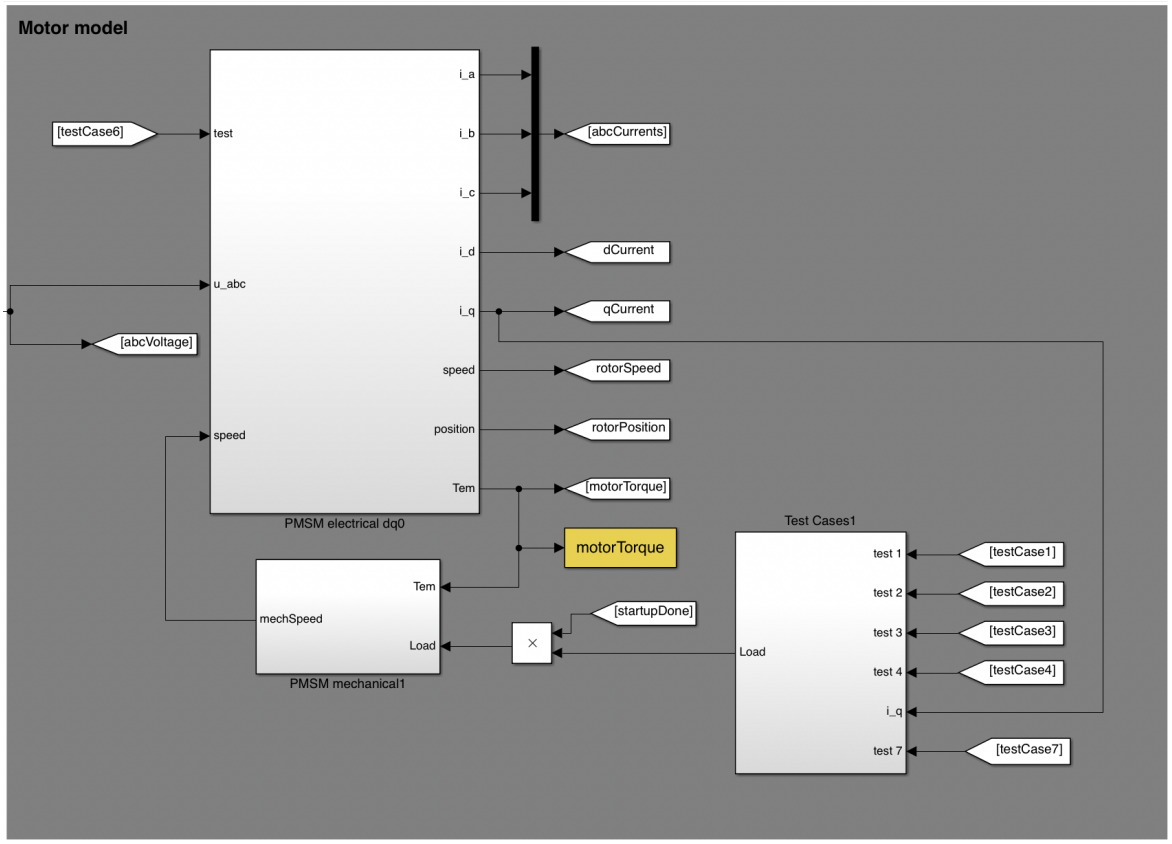


Figure 4.2: Simulink block of the motor model

In figure 4.2 the model for the motor can be seen. It has two main inputs, which can be seen on the left side of the biggest block. The input voltage u_{abc}, which inside the block will be transformed to dq voltages. There is no space vector modulation and inverter step in the simulation as it is not needed for this purpose. The Other input is the feedback of the speed. The other two blocks are there for test cases which will be treated later.

The outputs from the model are stated on the labels. The information retrieved from this model are the current in both dq coordinates and in abc coordinates. Further, the rotor speed and position are given along with the motor torque produced. However, speed and position will also be estimated by another estimator for the sensorless control.

4.3 Field oriented control

Next section of the simulation model is the control system. The model is based upon the model given in 2.2. The simulink model is presented in figure 4.3. It is visible that the FOC scheme in the simulink model consists of three PI controller blocks. They are constructed and tuned in a previous thesis work. In order to understand the tuning process, it is described in detail in [28]. There is also a decoupling block, which is there to compensate for the back-EMF produced by the rotor. There is also a speed input signal, which can be chosen to come from both the motor directly or the estimation block. The reference i_d-current can be set here as well. It is usually fixed to zero but can have any value. To compare it to the i_q-current, which is chosen depending on the speed reference. The speed PI controller generates i_q-current reference depending on the speed reference. Then voltages are transformed into *abc*-current before being fed into the motor model.

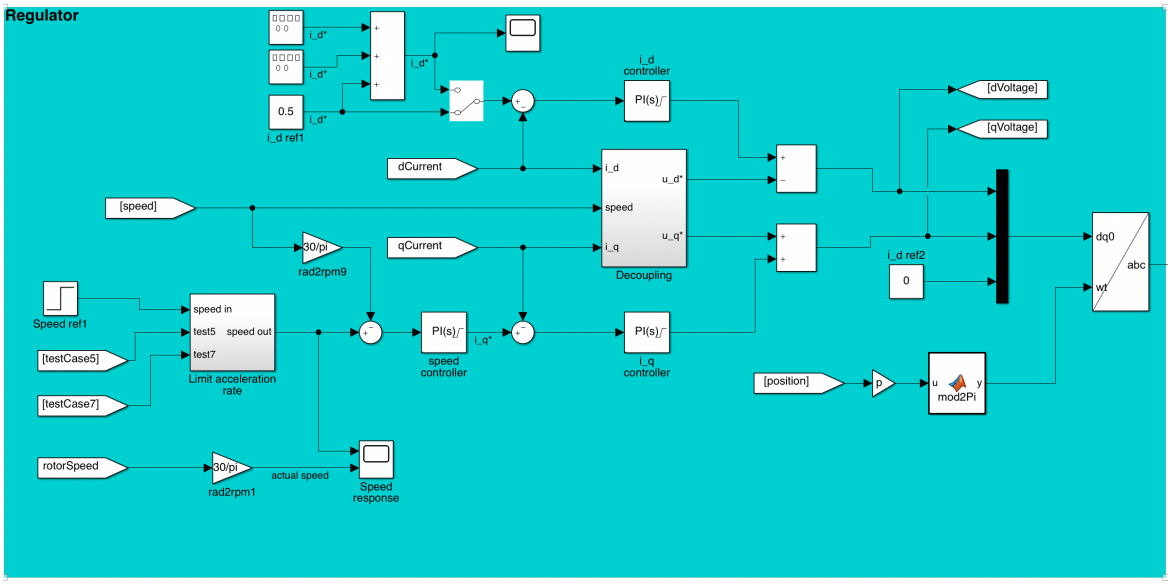


Figure 4.3: Simulink block of the control system

4.4 MRAS Speed and position estimator

The sensorless control means the control scheme will retrieve the information about position and speed through the estimator. In this section, the estimator is an MRAS estimator. The principle for this estimator is the same as the one explained in the previous section. However, the speed and position estimator is implemented and presented as a thesis work in [28], thus this thesis will not elaborate further on the speed and position estimation process.

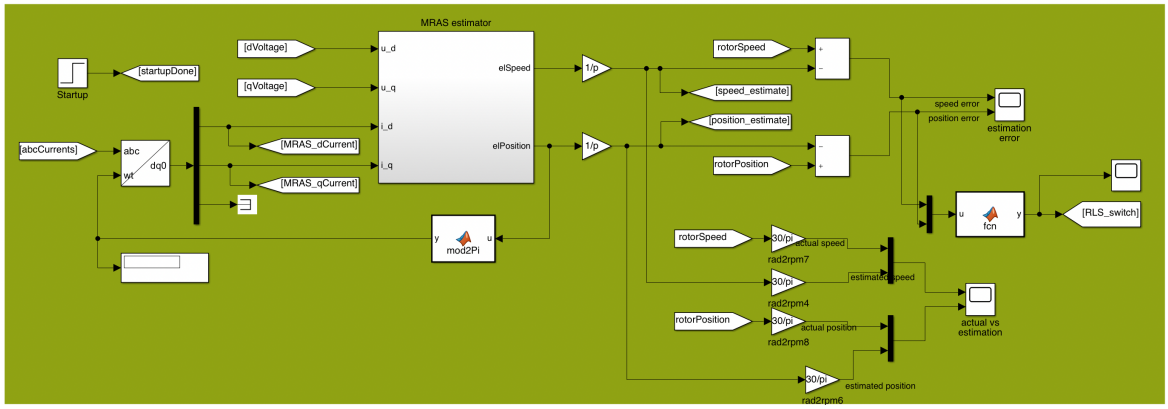


Figure 4.4: Simulink block of the control system

4.5 Parameter estimation blocks

The parameter estimation blocks is where the estimation methods are implemented. The first one is the RLS, the second MRAS and the third one is the EKF estimator. All of the blocks use both currents and voltages as inputs, along with the motor speed ω. This means that the simulation will be affected by sensorless control since the accuracy of the position and speed estimation will determine the efficiency and accuracy of the estimator. Here the main parameters to be estimated are the stator resistance R_s and the inductance L_s. Both estimated and the actual value of the parameters are plotted in a scope.

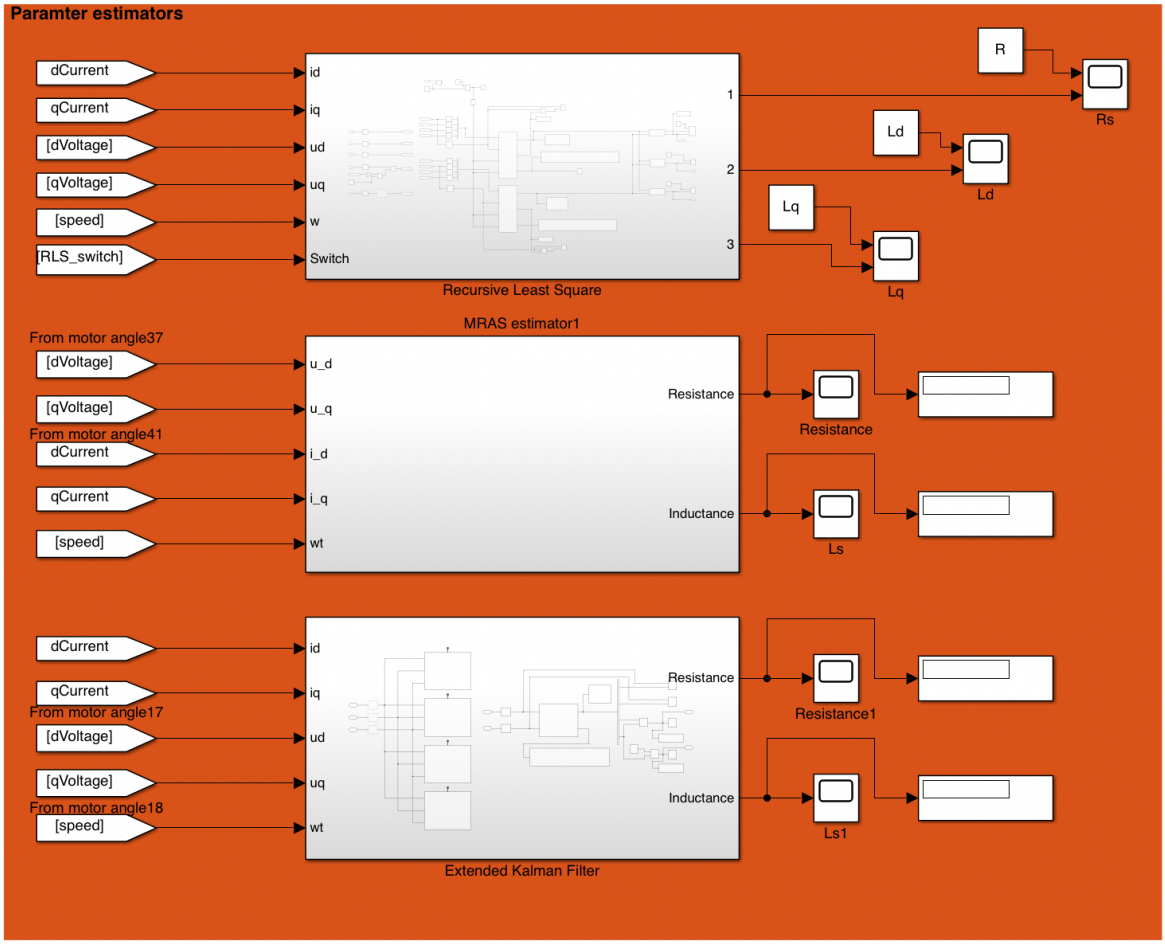


Figure 4.5: Simulink block of the control system

4.5.1 EKF in Simulink

The extended kalman filter is implemented using the 'System identification toolbox' which is provided by Mathworks as a ready algorithm. It is configurable to the user's needs and there are many calculations and models that need to be determined before the simulation can be executed. The block model of the EKF is given in figure 4.6

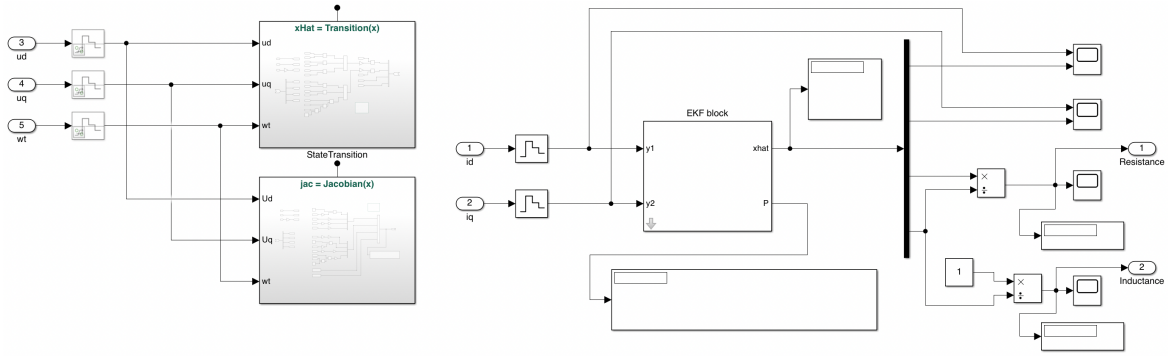


Figure 4.6: Simulink implementation of the extended kalman filter

The first step is to configure the state transition function and the jacobian of that

transition function. It can be done both algebraically as an m-file or graphically as a simulink function. In this case it was done as a simulink function where the transition function was created in the upper left block and its jacobian is beneath it. The inputs to both functions are the dq-voltages and the speed ω_{el}. The state transition function is defined in equation 3.9, however, in this case it written in discrete form, see 4.1.

$$x(k+1) = x(k) + f(x_k, u_k)T_s \tag{4.1}$$

Here, $f(x_k, u_k)$ is the state transition function and T_s is the sample time. x_k is the current state[24].

The jacobian of the state transition function, which is defined in 3.7 is also implemented in discrete time with the same formula as given in 4.1. The jacobian is given below 4.2.

$$F(x(t)) = \frac{\partial \hat{x}}{\partial x}|_{x=x(t)} = \begin{bmatrix} 1 + \frac{R_s}{L_s}T_s & \omega_{el}T_s & -i_d T_s & u_d T_s \\ -\omega_{el}T_s & 1 + \frac{R_s}{L_s}T_s & -i_q T_s & (u_q - \omega_{el}\psi_M)T_s \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.2}$$

The inputs to the EKF block are the $y(t)$ signals of the system. This is stated in 3.10, where it can be seen that y_1 is i_d and y_2 is i_q. These signals come directly from the motor. Further configurations in the EKF block require to decide matrices P_0, Q and R. These are the covariance matrices. As stated in the section 3.1.3, there are no methodical ways of deciding them. In this application it was decided by trial-and-error until the convergence became correct and fast enough. The covariance matrices are given below.

$$P_0 = \begin{bmatrix} 10^{-2} & 0 & 0 & 0 \\ 0 & 10^2 & 0 & 0 \\ 0 & 0 & 10^3 & 0 \\ 0 & 0 & 0 & 10^5 \end{bmatrix} \tag{4.3}$$

$$R = \begin{bmatrix} 10^{-2} & 0 \\ 0 & 10 \end{bmatrix} \tag{4.4}$$

$$Q = \begin{bmatrix} 10^{-2} & 0 & 0 & 0 \\ 0 & 10^2 & 0 & 0 \\ 0 & 0 & 10^5 & 0 \\ 0 & 0 & 0 & 10^6 \end{bmatrix} \tag{4.5}$$

Another important aspect in the implementation of the EKF is to set the initial state value. In this case, the states are set as given in 4.6.

$$\begin{bmatrix} 1_d & i_q & a & b \end{bmatrix}^T = \begin{bmatrix} 0 & 0 & \frac{R_s}{L_s} & \frac{1}{L_s} \end{bmatrix}^T \tag{4.6}$$

4.5.2 Recursive least square in simulink

The RLS algorithm is implemented in simulink with the same toolbox as the EKF, the 'System identification toolbox'. Here it is also configurable to include a forgetting factor in the algorithm. The structure of the RLS implementation is the same as given in section 3.2.2 and the regression model as given 3.17. The only difference is that in 3.17, the parameter variables are stored in a matrix, while in the implementation in simulink, it is done with the two estimators, where each estimator generates a vector. The generated vectors are given in 4.7 and 4.8. Figure 4.7 shows the implementation of one RLS block. The regressors, are the currents and voltages with one time step delay. As seen with the z^{-1} blocks.

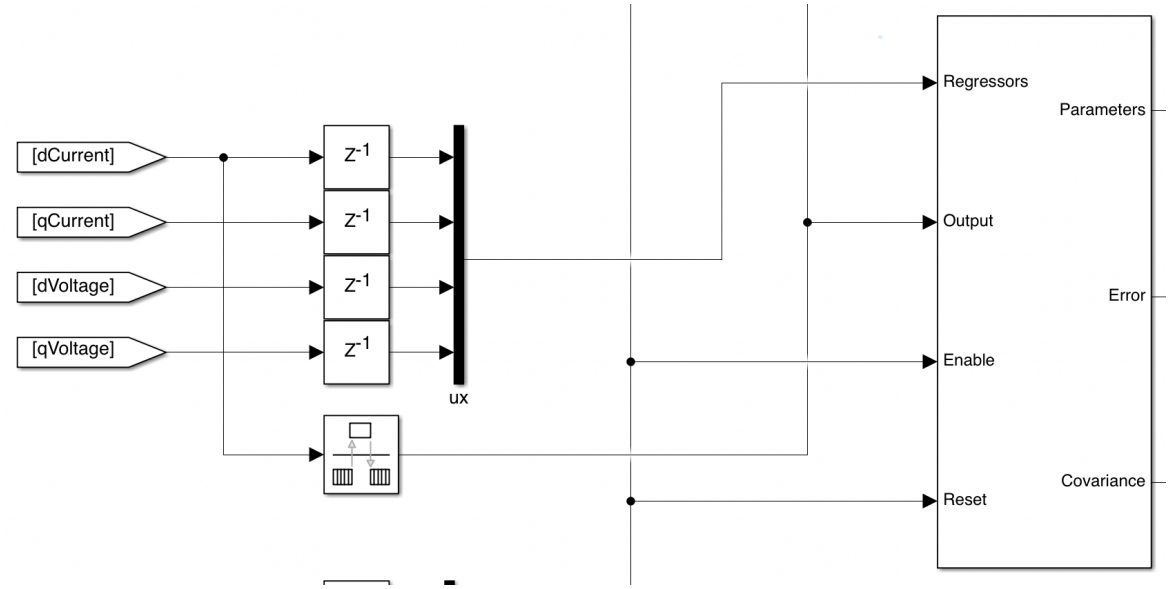


Figure 4.7: RLS Imlpementation in simulink

$$\theta_1 = \begin{bmatrix} a_{11} & a_{12} & b_{11} & b_{12} \end{bmatrix}^T \qquad (4.7) \qquad \theta_2 = \begin{bmatrix} a_{21} & a_{22} & b_{21} & b_{22} \end{bmatrix}^T \qquad (4.8)$$

Furthermore, the implementation of the RLS block requires the user to define the initial state estimates and the initial covariance matrix value. In this case, since the parameters in both θ_1 and θ_2 are derived from the motor's discrete time state space equation given in 2.27, the initial states are stated in 4.9 and 4.10. The forgetting factor λ is decided with trial-and-error and was decided as $\lambda = 1 - h$, with h being the sample time.

$$\theta_1 = \begin{bmatrix} -\frac{R_s}{L_s}h + 1 & \omega_{el}h & \frac{1}{L_s} & 0 \end{bmatrix}^T \qquad (4.9) \qquad \theta_2 = \begin{bmatrix} \omega_{el}h & -\frac{R_s}{L_s}h + 1 & 0 & \frac{1}{L_s} \end{bmatrix}^T \qquad (4.10)$$

4.5.3 MRAS

The MRAS implementation is also very much alike the block model presented in figure 3.1. As can be seen in 4.8, the block to left is the motor model. The inputs to this model are the *dq* voltages, which are given from the controller. The motor model then generates *dq* currents.

The currents, both from the motor itself and the motor model model, will be fed into the adaption scheme. The voltages will also be input to the block. The adaption law's for both the stator resistance R_s and the inductance L_s are given in equations 3.22 and 3.23. It is clear that they are PI-controller, and in this case they are implemented with a PI-block from the matlab library. The resistance and inductance are fed back to the motor model and will continually be updated until the current difference between the motor and the model is zero.

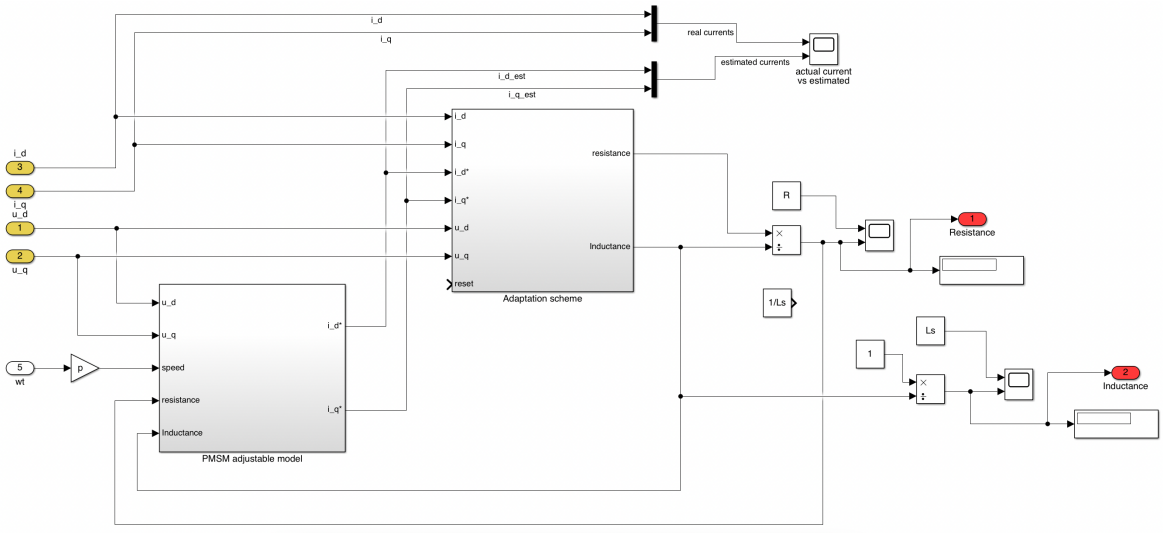


Figure 4.8: MRAS Imlpementation in simulink

The configuration for the MRAS scheme, apart from building the blocks, is to configure the PI-parameters for both resistance and inductance estimation. They are chosen to the values given below.

$$K_{Rp} = 5 \qquad (4.11) \qquad K_{Ri} = 0.1 \qquad (4.12)$$

$$K_{Lp} = 5 \qquad (4.13) \qquad K_{Li} = 0.01 \qquad (4.14)$$

5 Results

In this chapter the results from the simulations are presented. The results are done with different test cases where some cases are are tried both with sensorless control and with regular control and some only with sensorless control. In addition, the simulation results are only commented in this section. A deeper analysis of the results is presented in the discussion section.

5.1 Regular control scheme - No load

The first step of the simulation is to drive the motor up to a reference speed with no load applied. This will give an indication of how the estimators work on basic performance. This is a minimum requirement on the estimators. The reference speed is set to 8500rpm and in this case it is also done with no limitation on the acceleration. Here, the controller's step response time will also be tested. Two simulations tests are done, where one applies control with sensors and the other one will use sensorless control. This is to show how estimators work differently with different control schemes. Figure 5.1 shows the dq-current, speed and torque for this case. There is a big overshoot in i_q up to 60A in the beginning, which is the transient phase. The rest of the simulation the i_q is at around 1.7A and the i_d current is set to zero. There is also a torque in the beginning measuring 1.5Nm which later goes down.

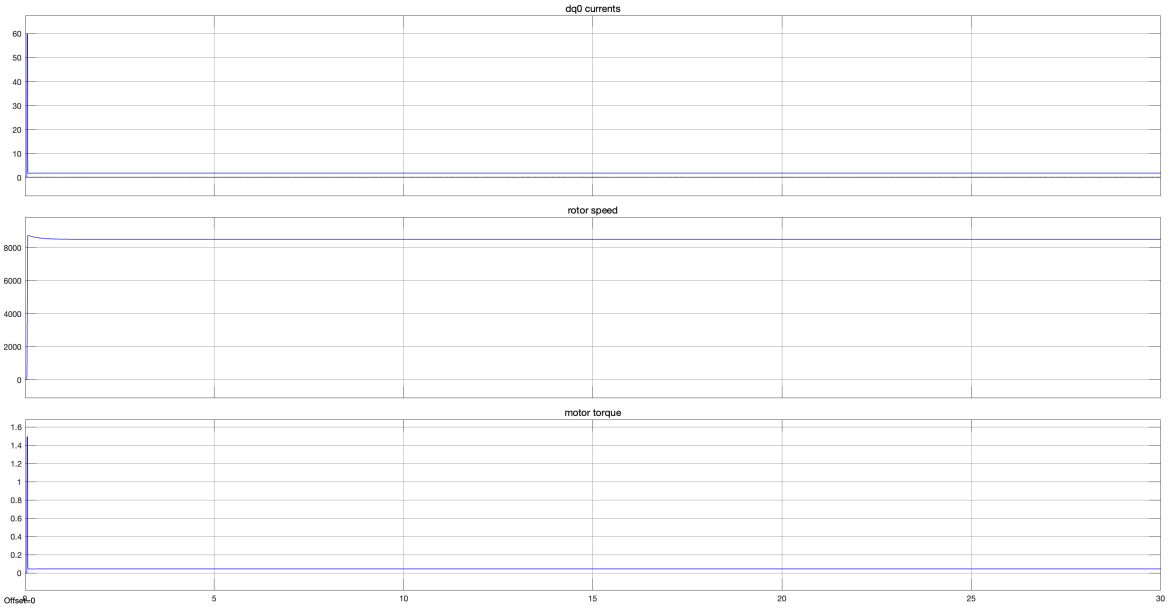
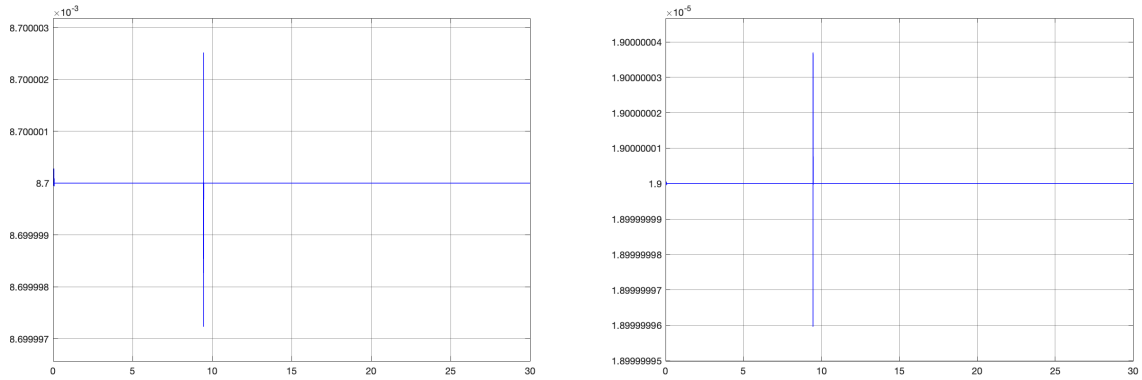


Figure 5.1: Graphs of dq-currents in *ampere* (Red: d-current, Blue: q-current), speed in rpm and torque in Nm with position feedback and no load applied

MRAS

The graphs show the MRAS estimation method without sensorless control scheme. The results give no error in the identification. This is because the simulation in matlab uses the same motor model equations in both the estimator and the actual plant motor. This is an ideal case and this will almost never be emulated when running the estimation on an actual motor.

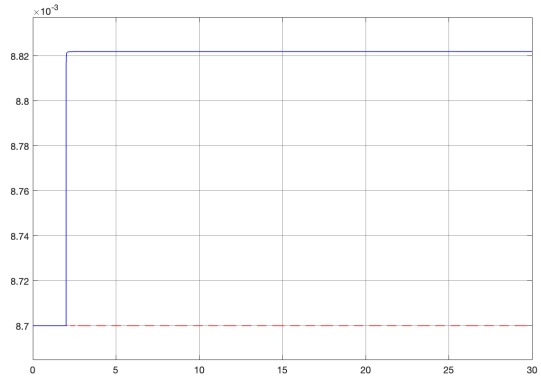


(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

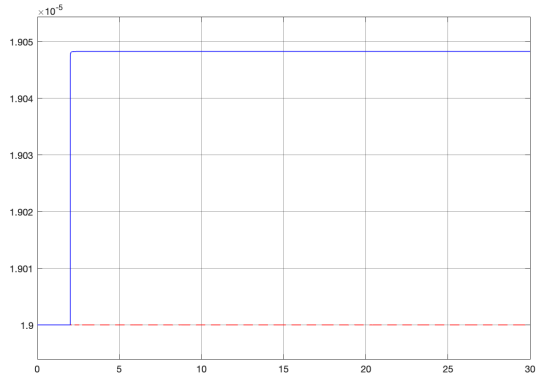
Figure 5.2: MRAS estimation with sensor control. Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

RLS

Figure 5.3 shows the RLS estimation with a sensor control scheme. The blue lines are the estimated lines and the red line is the actual value. It is clear that the estimator works for both the resistance and the inductance. The convergence time is 6ms which is very fast and the convergence value is very close to the actual value with a difference of only 1.38%. For the inductance the convergence time is 20ms with an error of 0.2%. The RLS algorithm also does not experience any wind-up problems as the convergence value is constant throughout the entire 30 second simulation on both parameters. It should be noted that the RLS algorithm is activated 2 seconds after the motor starts running in order to reduce the effects of the transient currents from the motor.



(a) Resistance R_s (Ω)

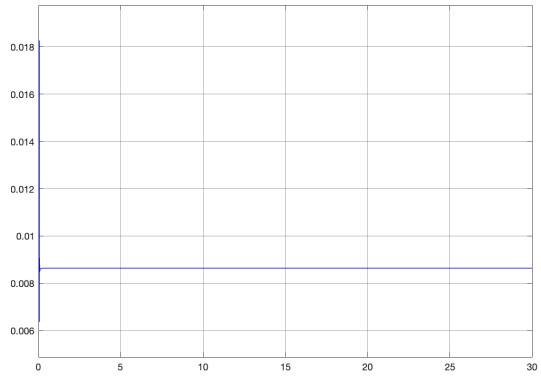


(b) Inductance L_s (H)

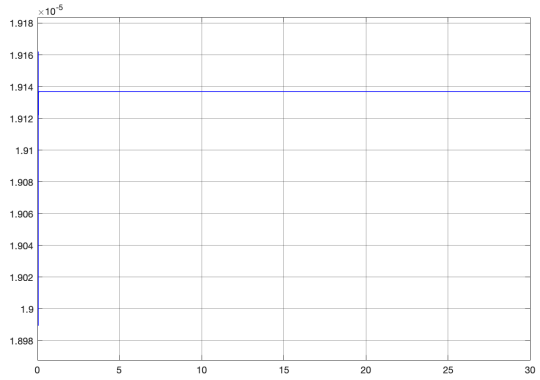
Figure 5.3: RLS estimation with sensor control. (Red: Actual values, Blue: Estimated) Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

EKF

The last estimator to be tested is the EKF estimator. Here, it is also clear that the estimator works very well both in terms of time and value it converges to. The big overshoot in the beginning is due to the transient currents resulting from the motor dynamics. The values also stay constant during the 30 second simulation. In the resistance estimation, the settling time is around 0.1s and the inductance settles with 0.08s. The accuracy of the estimations are only deviating around 0.6% for the resistance and 0.7% for the inductance.



(a) Resistance R_s (Ω)



(b) Inductance L_s (H)

Figure 5.4: EKF estimation with sensor control. Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

5.2 sensorless control - No load

This section simulates the motor the same way as it is done in the previous test. The motor runs up to a reference speed of 8500rpm and stays there. However, in this case, the position and speed is estimated by an MRAS position estimator. In figure 5.5, it is seen that there is no difference in how the currents, speed and torque behaves

compared to the position feedback. There are small errors in the position estimation but this will be treated in section 5.3.1.

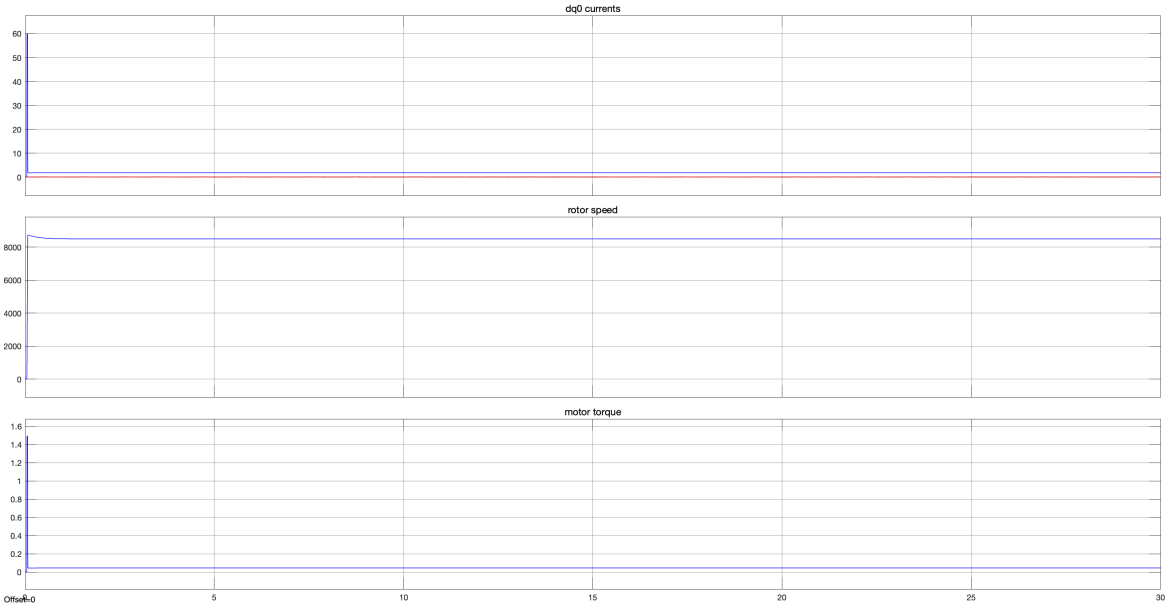
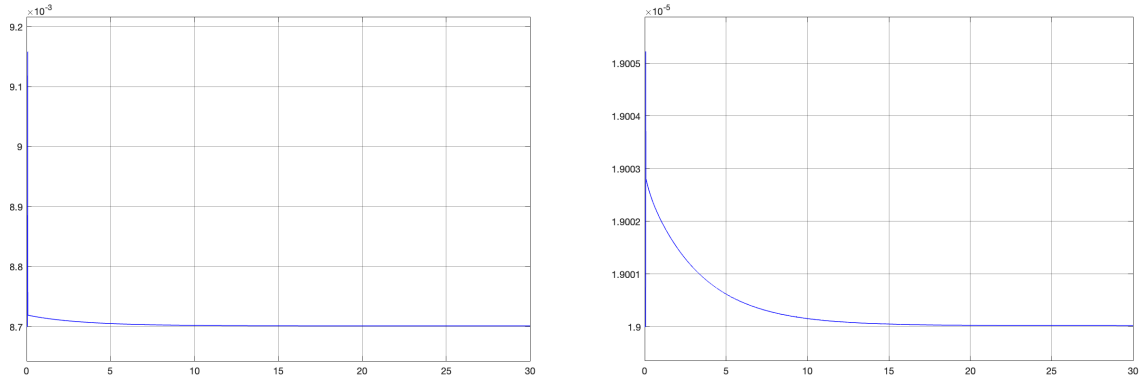


Figure 5.5: Graphs of dq-currents in *ampere* (Red: d-current, Blue: q-current), speed in rpm and torque in Nm with sensorless control and no load applied

MRAS

The MRAS estimation works very well with sensorless control as well. The results have an overshoot in the beginning and the convergence is a little slower than the previous test but the results are as expected. The convergence error for both parameters are lower than 0.1% and the convergence time is 10s for the resistance and around 15s for the inductance, which is significantly higher than with the actual position feedback from the motor.

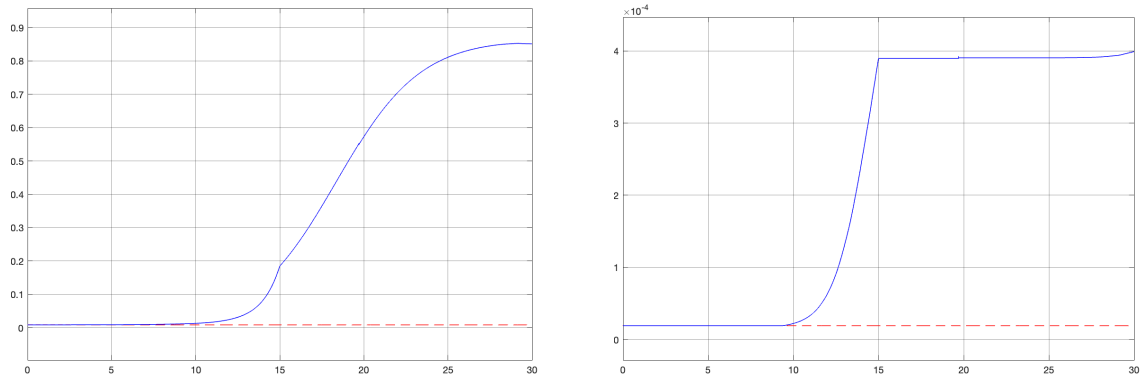


(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.6: MRAS estimation with sensorless control. Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

RLS

The results from the RLS estimator can be seen in figure 5.7. From the graphs it is obvious that in the sensorless case the signals do not converge to the expected value. In the resistance case it is off by a factor bigger than 100. The same thing goes for the inductance, where it too converges after to the wrong value after approximately 10s. This is a case of wind-up problem, where the covariance matrix increases to such extent that the slightest error becomes magnified. It is also caused by the small error in the position and speed estimation, as seen in figure 5.8. There it can be seen that the position error slowly decreases and reaches a value close to zero at around 20s.



(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.7: RLS estimation with sensorless control. (Red: Actual value, Blue: Estimated) Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

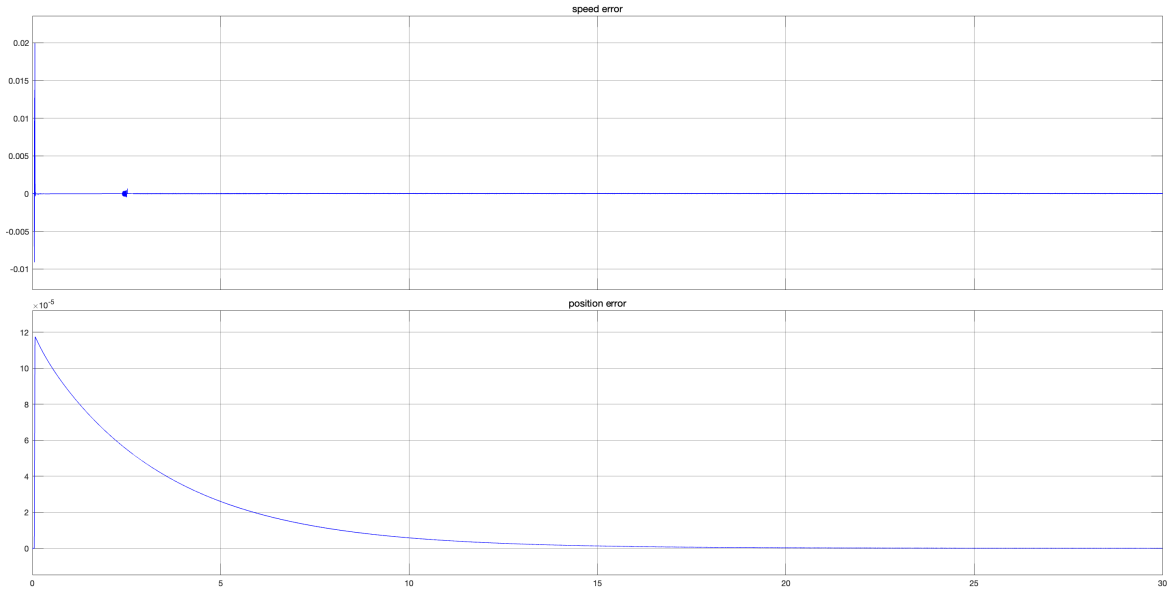
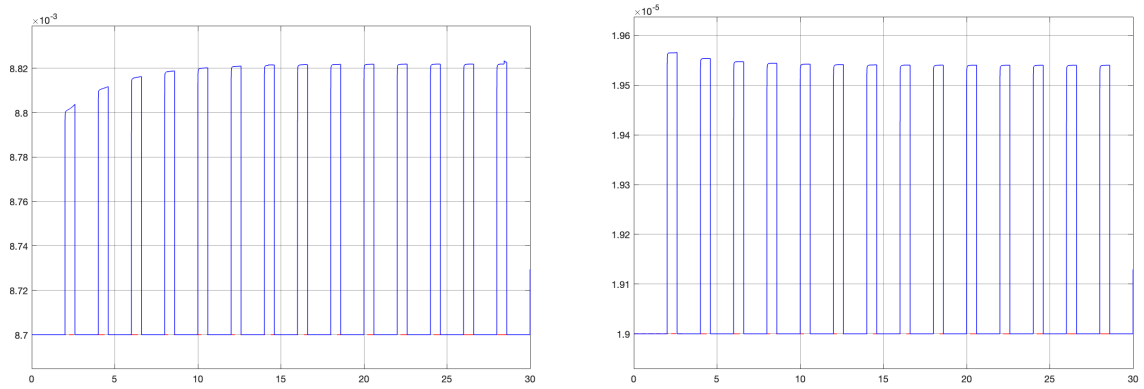


Figure 5.8: Estimation error of speed and position from the MRAS estimator given in mechanical speed ω_r (rad/s) and mechanical position θ_r (rad)

There can be adjustments made as mentioned in [19]. One of the methods presented to avoid wind-up is to reset the RLS covariance matrix when it increases to large

numbers. By injecting pulses with 1 second intervals to the RLS estimator, and reset each time the pulse goes down to zero, better results can be achieved. This is shown in figure 5.9, where the estimation of the parameters are the value when the pulse is "high". By looking at the graphs, the parameters converge faster when the error in the position estimation goes towards zero. It is also seen that the values of the pulsating RLS estimator gives a result similar to the one in the previous test both in terms of convergence error and convergence time.

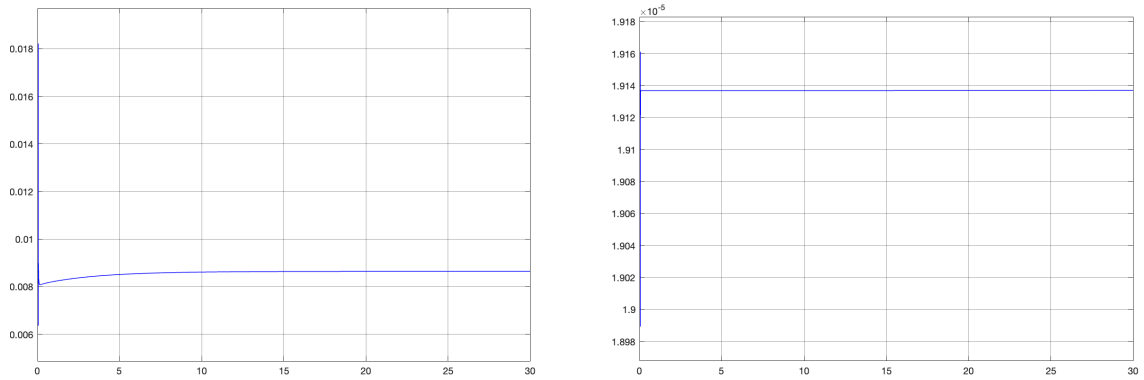


(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.9: RLS estimation with sensorless control and pulse injection. (Red: Actual value, Blue: Estimated) Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

EKF

The Results from the EKF estimator in the sensorless case are in terms of the convergence time around 9s for the resistance and for the inductance it is around 0.07s. The estimation error for the parameters are the same as they are in the previous case. Compared to the convergence time in the position feedback case, this one is slower, which is the case with the rest of the estimators in the sensorless case as well.



(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.10: EKF estimation with sensorless control. Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

5.3 Test cases

In this section, different test cases are introduced to try and test the estimators at different operating points. This is to ensure the stability and robustness of the estimators when the motor is not running on constant speed with no load applied to it. In real application, this operating point is rarely achieved. The test cases are taken from the previous thesis work, where they were created to emulate the different operating points of the motor. Description of the test cases can also be found in [28]. Further, most of the test cases will use sensorless control scheme since this is how the real application is implemented. However, in test case 8, the MRAS position estimator is not able to handle inductance variation during drive. Therefore, in this case actual position feedback from the motor is used.

5.3.1 Test case 1 - step load

In this simulation, once the motor reaches steady state on the reference speed, a load of 1 Nm is added for 1s and then removed. This repeats every 1.5s. The idea for this test case is to see how the estimators work with load applied periodically on the motor. This is a normal operating point for the motor. In figure 5.11, it is shown how the test case changes the currents and torque. The i_q current switches between 1.78A and 42A when the load is applied, which is also seen in the torque graph. The speed stays at the reference speed of 8500.

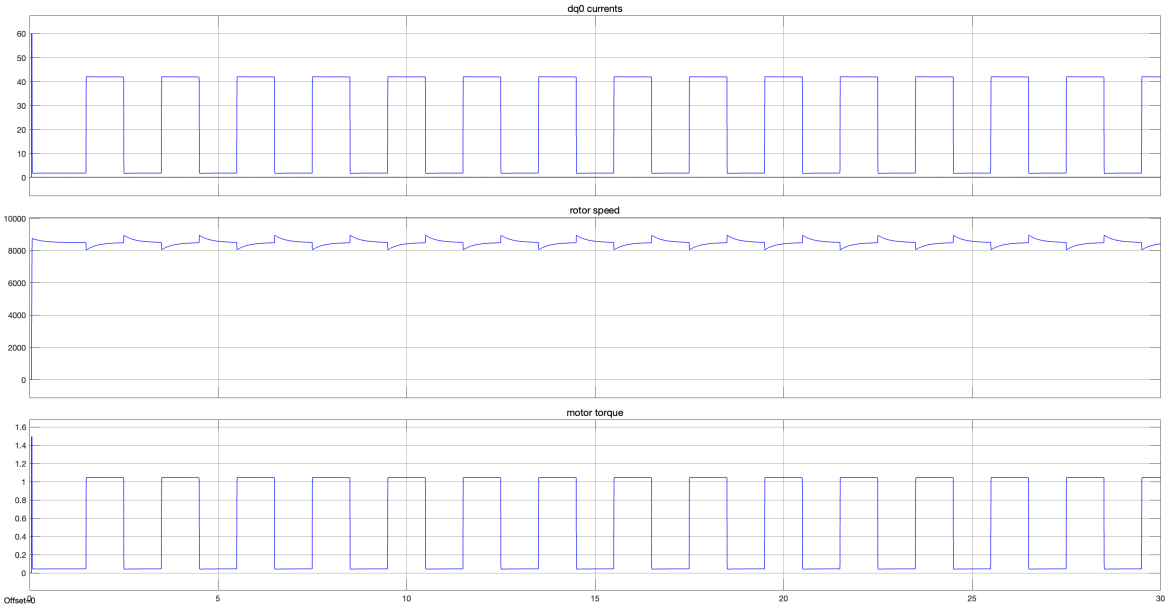
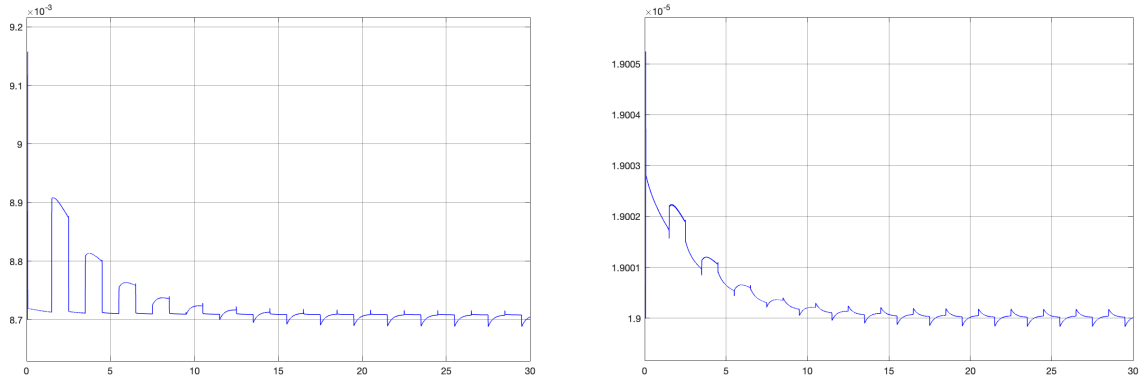


Figure 5.11: Graphs of dq-currents in *ampere* (Red: d-current, Blue: q-current), speed in rpm and torque in Nm with test case 1

MRAS

As figure 5.12 shows, the MRAS estimation converges to the right value in both resistance and inductance. The convergence time in this case is the same as it is in the previous cases, which gives a steady value at around the 10 second mark. Furthermore, the oscillation of the values are due to the periodic load applied. When a load is applied the i_q current switches the value of 42A from its original value of around 1.78A. Since the estimators uses the current signals to estimate the parameters, this oscillation of could be due to the high changes in currents. Both parameters settles to a value after approximately 11s with an error of less than 0.1%.

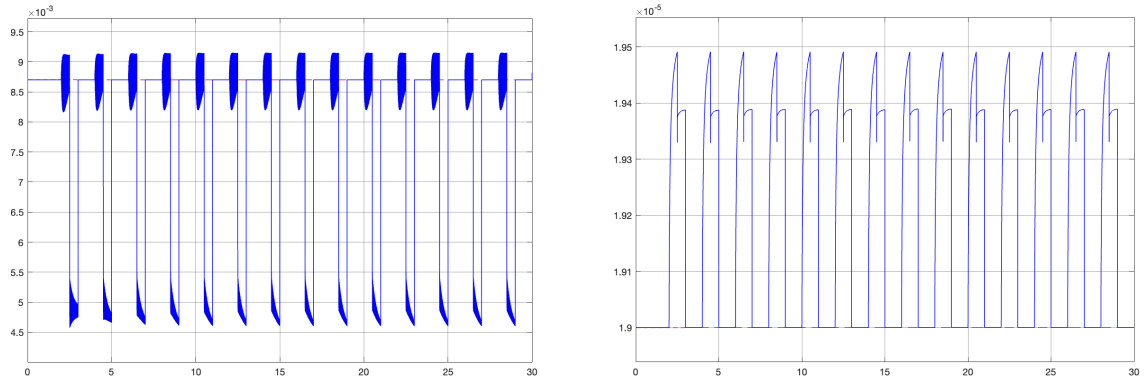


(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.12: MRAS estimation with test case 1. Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

RLS

A pulse is given to the RLS estimator every second the same way as it is done in the previous case. This is done throughout all the test cases as well unless something else is stated. With the RLS algorithm it is obvious that it cannot handle online parameter estimation while it has a load added periodically. Especially when it comes to estimating the resistance. The resistance estimation oscillates first around the value at which it is expected, around resistance value of 0.0087, then when the load is added, the estimated value drops around 30%. This is because the i_q current is changed to a much higher value of 42A. The same result is given in the inductance estimation, although it is not as bad as the resistance. However, it never reaches a convergence state and therefore the results are not sufficiently good.

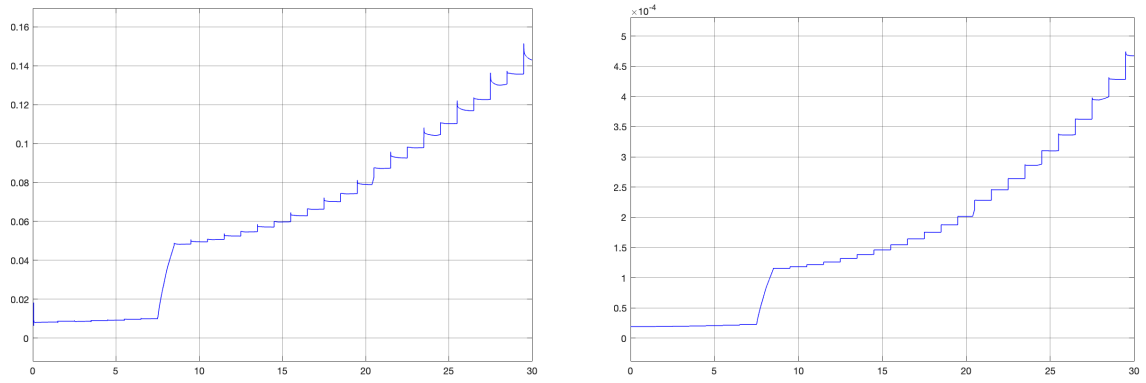


(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.13: RLS estimation with test case 1. (Red: Actual value, Blue: Estimated) Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

EKF

The EKF estimator clearly does not work well with the step load estimation. For the first 7 seconds of the simulation, the estimation of the parameters are rather stagnant, however they start to diverge afterwards, which is seen in figure 5.14. This happens in both cases. One way to prevent this is to excite the i_d current, which is one of the inputs to the EKF estimator. This is done due to the fact that the signal provides more information about the system dynamics. The excitation signal could vary in many different ways, in this case the i_d current is chosen to have staircase shape with a frequency of 200Hz and four levels with an amplitude of 0.15A. The i_d current chosen is given below in figure 5.15.



(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.14: EKF estimation with test case 1. Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

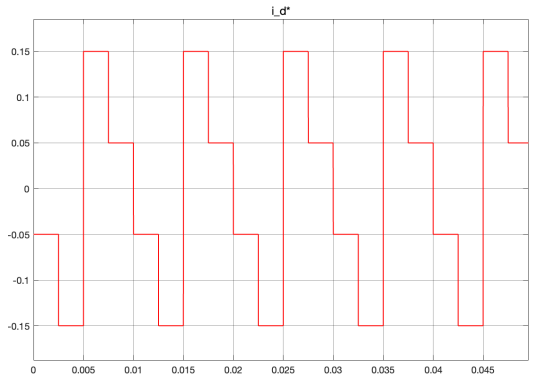
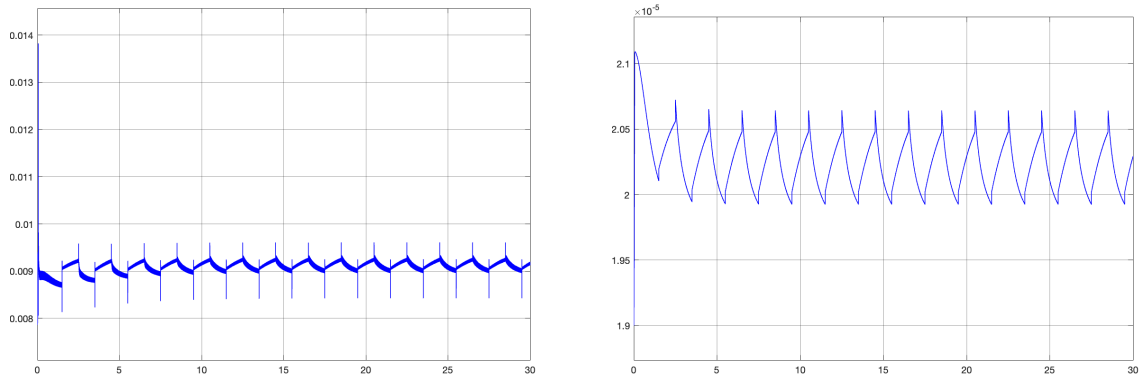


Figure 5.15: Excited i_d (A) current signal fed to the motor system

With the implementation of the excitation signal, the results from the EKF can be seen in 5.16. The results are much better even though there are oscillations in the estimation, they are centered around the expected value of both parameters. The resistance settles around 4s and has an error of approximately 2.5% and the inductance settles after 5s with and error of 6%. The resistance estimation overshoots when the currents load is added or removed, but this can be neglected by low-pass filtering the estimation signal. The same thing could be done with the inductance estimation.



(a) Resistance R_s (Ω)

(b) Inductance L_s (H)

Figure 5.16: EKF estimation with test case 1 and excited i_d signal. Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

5.3.2 Test case 2 - Periodic load

Test case 2 is similar to test case 1. The difference is that the load is added in the shape of a sinusoidal signal with an amplitude of 0.2 Nm. The idea is to show how the estimators work when the load is not constant. Figure 5.17 shows how there is a periodic change in the i_q current as well as in speed and torque.

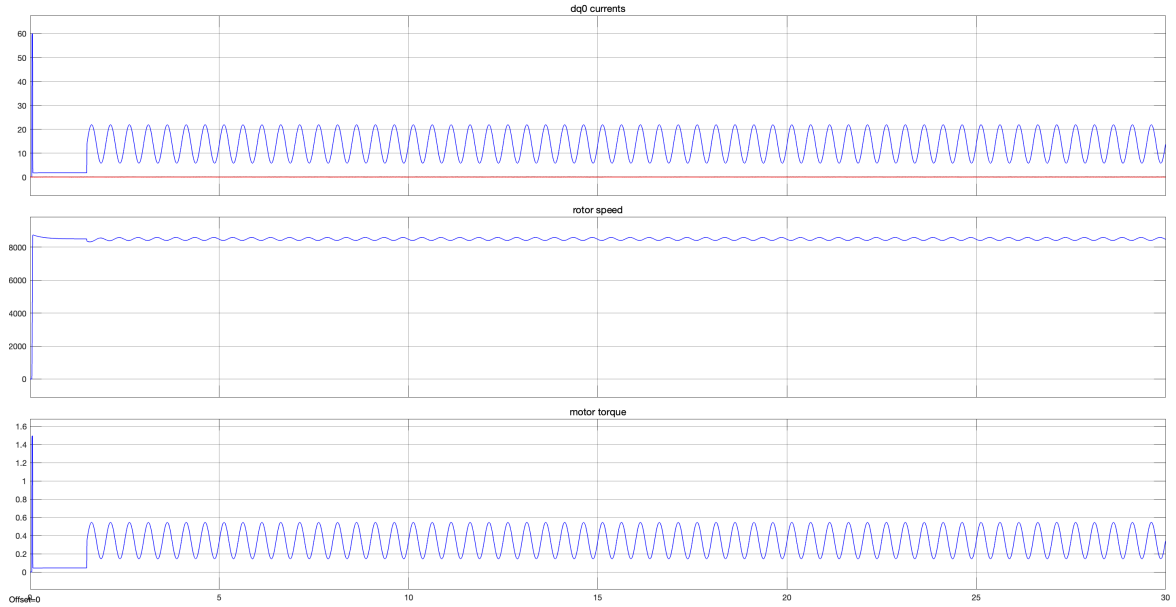
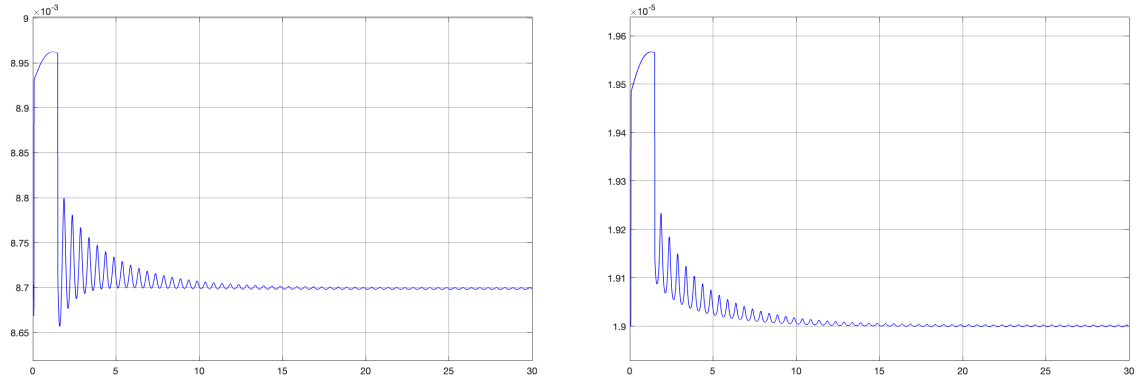


Figure 5.17: Graphs of dq-currents in *ampere* (Red: d-current, Blue: q-current), speed in rpm and torque in Nm with test case 2

MRAS

The results from the MRAS estimation show a slight sinusoidal disturbance in the estimation. It is specially visible in the resistance estimation. The current is affected when applying the load, this is the same as in the previous case. Therefore there is a slight disturbance, however it is very small and the estimation converges towards the actual value in both parameters. The settling time for the resistance is 5s and for the inductance it is 10s. The estimation error is 0.5% and 0.1% for the parameters.

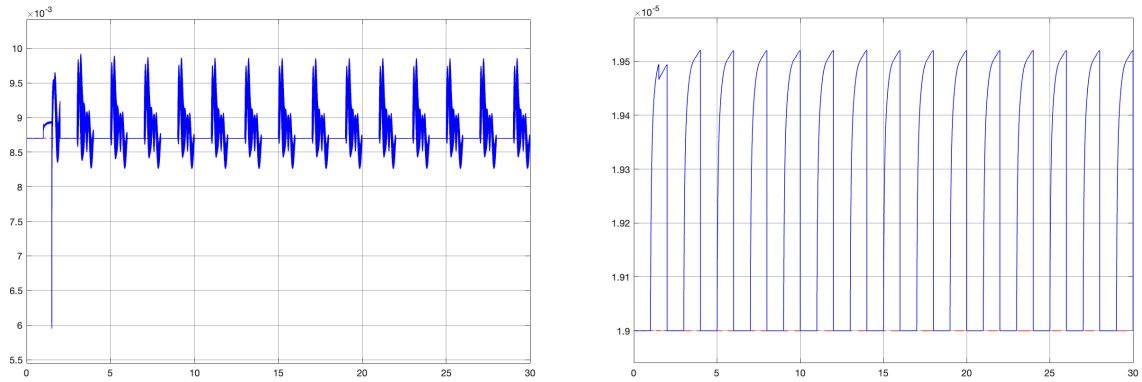


(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.18: MRAS estimation with test case 2. Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

RLS

In the case of the RLS with a periodic load, the estimation method does not work. This test case is also simulated with the pulsating RLS method. It is seen that the estimation oscillates when the load is applied, and there is no clear way of determining the estimation value for either resistance and inductance since neither converges properly. The resistance estimation is more noisy due to the excited i_d signal as given in previous section and shown in figure 5.15.

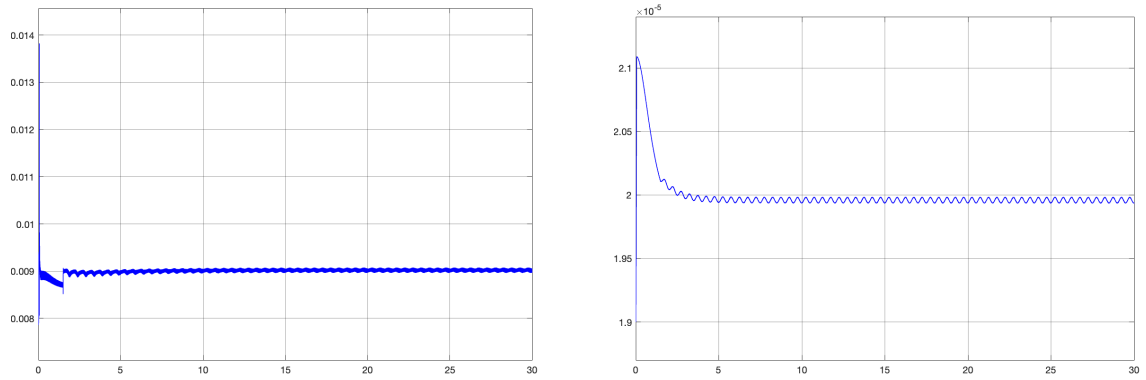


(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.19: RLS estimation with test case 2. (Red: Actual value, Blue: Estimated) Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

EKF

The EKF has no major issue when it comes to estimating the motor parameters with periodic load. Like the other cases, there are small oscillations, which are more prevalent in the resistance case, however, it is deemed negligible for the convergence. The biggest drawback from the EKF is the offset in the convergence value for both parameters. They are slightly higher than the expected values of the parameters. The resistance settles after 0.5s, however the initial estimation after around 2s is not deviating much from the final convergence value. The final convergence value has an error of 3.45%. The inductance settles after 3s with an error of 4.95%.



(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.20: EKF estimation with test case 2. Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

5.3.3 Test case 3 - Step load in transient state

This test will have a load added to the motor when it is in the transient state. The load is 1Nm and is added when the motor has reached a speed of 4000 rpm. The reference speed is set to 8500 rpm. This shows how the estimators work when there is a disturbance before a steady state has been reached. The load that is added is seen in figure 5.21, where the i_q current has a steady value of 42A after the transient state as well as a constant torque of 1Nm. There is also no overshoot in the speed estimation for this case.

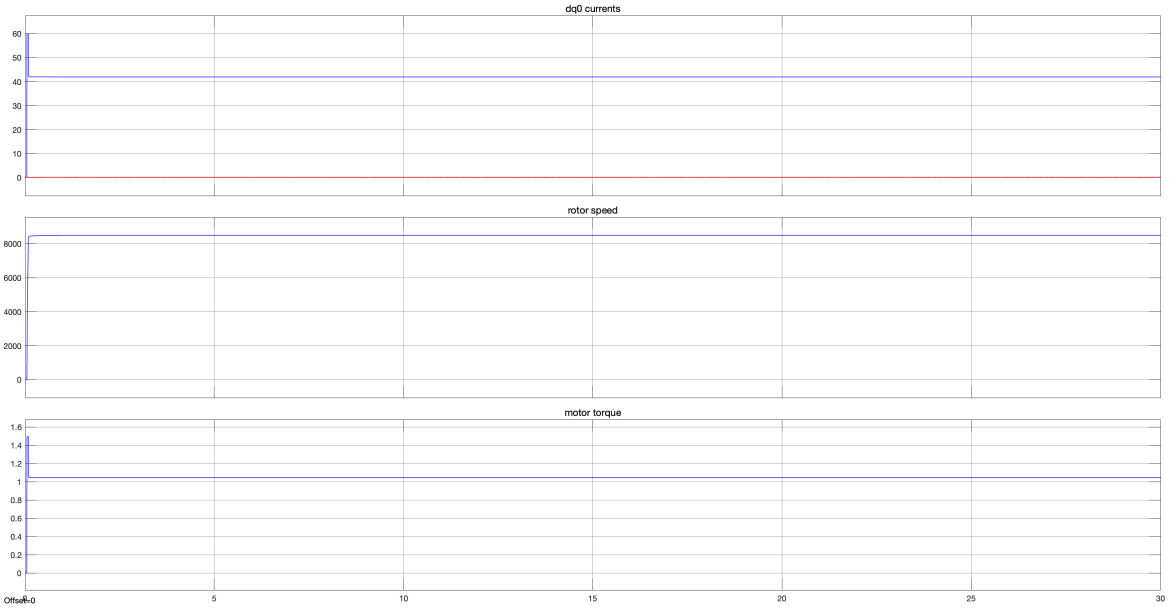
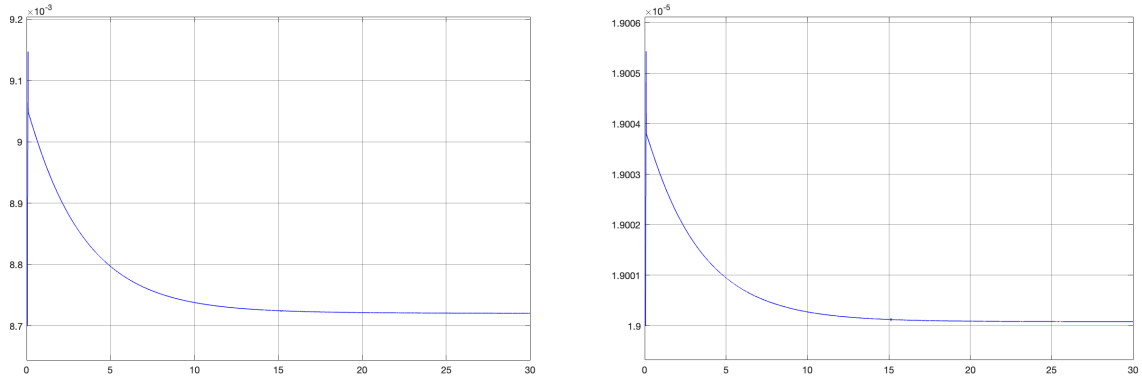


Figure 5.21: Graphs of dq-currents in $ampere$ (Red: d-current, Blue: q-current), speed in rpm and torque in Nm with test case 3

MRAS

When it comes to having a load applied in the transient state, there is no major issue on the estimation for the MRAS estimator. Both parameter values converge in the expected time towards the expected value. The transient state of the motor is not long and the controllers that are implemented work very well, thus the load from the start does not affect the estimation convergence. However, because there is a load in the beginning there is a higher current from the motor, thus leading to a higher overshoot in the beginning. The settling time for both parameters in this test is around 17s with an error of less than 0.3%.



(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

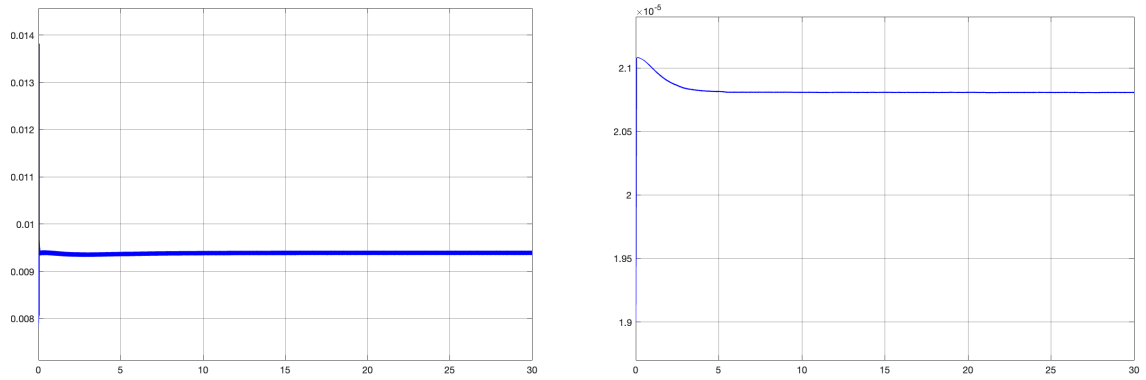
Figure 5.22: MRAS estimation with test case 3. Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

RLS

Because the RLS algorithm only works well when it is in steady state, the test on how it estimates during the transient phase is not applicable for this test. Consequently, this is seen a major drawback of this estimation model.

EKF

In figure 5.23, the EKF performs very well when a load is applied in transient phase. The convergence time is faster than the MRAS estimator and is is more stable. However, the setback is that the value the estimator converges to is slightly higher than the actual value. For the inductance estimation, the error in convergence value is more significant than it is for the resistance. The resistance reaches the final convergence value within 2s with 8% error and the inductance converges after 5s with 9.4% error.



(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.23: EKF estimation with test case 3. Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

5.3.4 Test case 4 - Start load

This test case will put a load on the motor from the start. It will be the highest possible load applicable for this motor and motor controller. It is determined by the highest possible current. In this test case, there is no difference in the current and torque graphs compared to the previous case. The small difference is that the i_q current is slightly higher to a value of around 50A ans the torque settles at around 1.25Nm. There is also a small overshoot in the speed estimation as seen in 5.24.

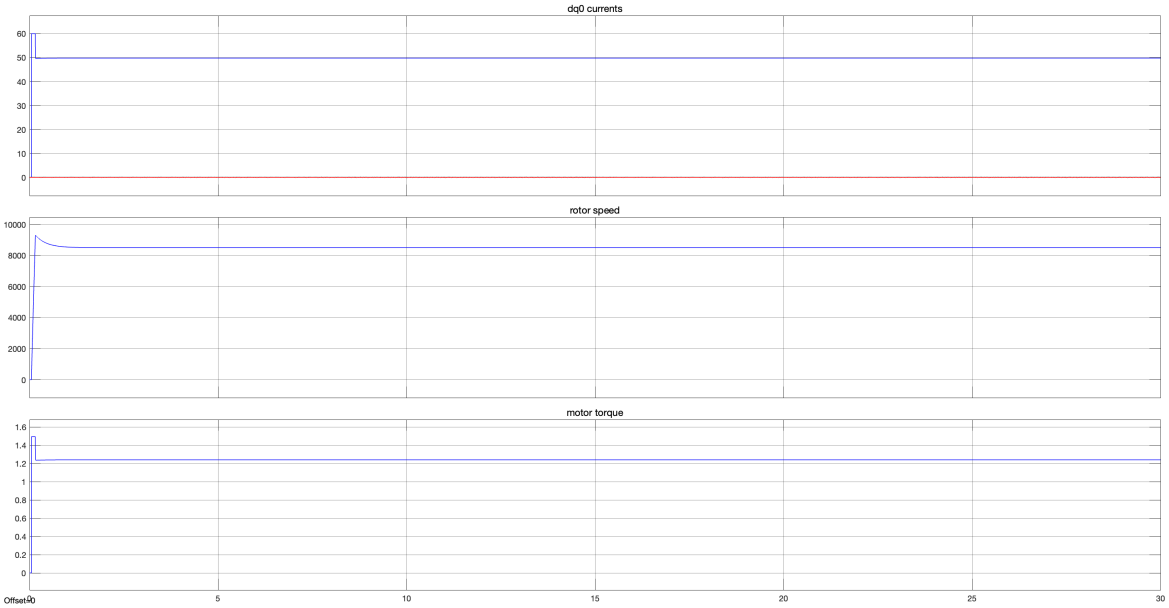
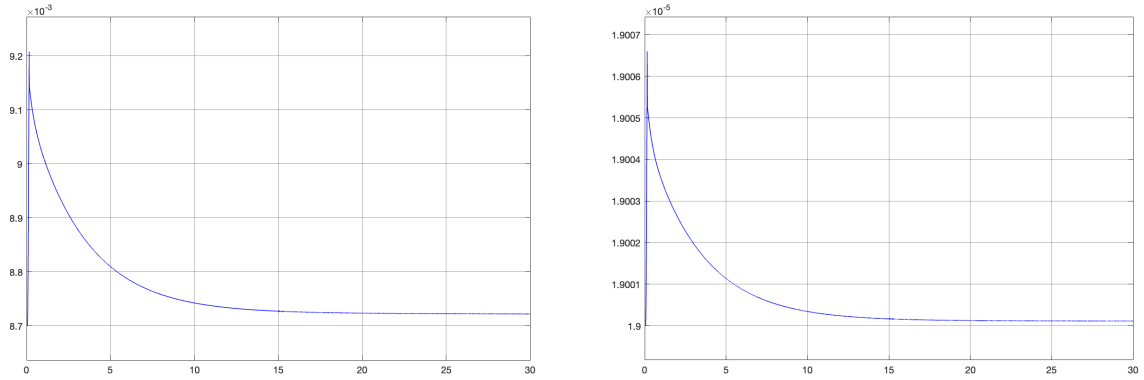


Figure 5.24: Graphs of dq-currents in *ampere* (Red: d-current, Blue: q-current), speed in rpm and torque in Nm with test case 4

MRAS

In the MRAS case, there is a high overshoot in the transient state of the R_s estimation but it quickly goes down as well. The overshoot in the inductance estimation converges slower but the overall estimation is as expected where both parameters converge around the 15s mark and the estimation error is below 0.3% for both estimations.

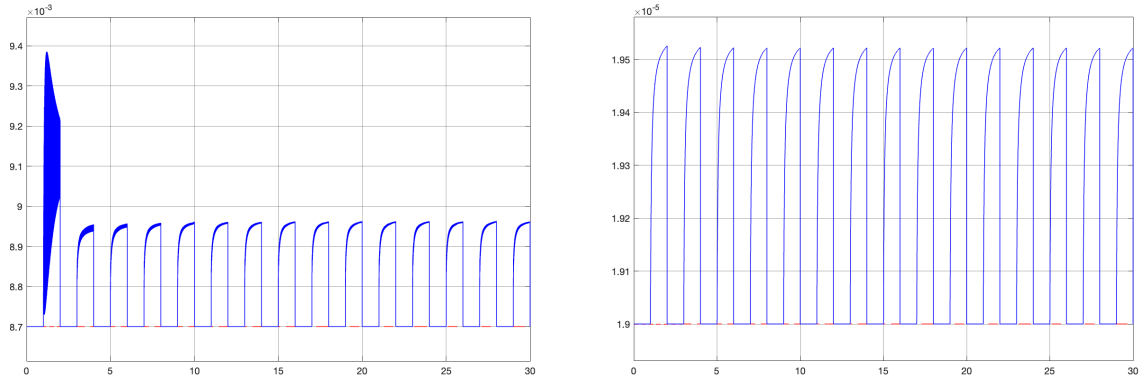


(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.25: MRAS estimation with test case 4. Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

RLS

For the RLS case, there is a big overshoot and big oscillation of the resistance estimation in the transient phase. However, after the transient phase, the convergence of the pulses stabilize and gives appropriate estimations of the resistance. The pulses settle after 0.8s and has an error of 6%. The inductance does not experience the same oscillations and has a constant estimation. The inductance value is also as expected with an error of 3.2%. Settling time is around 1s.

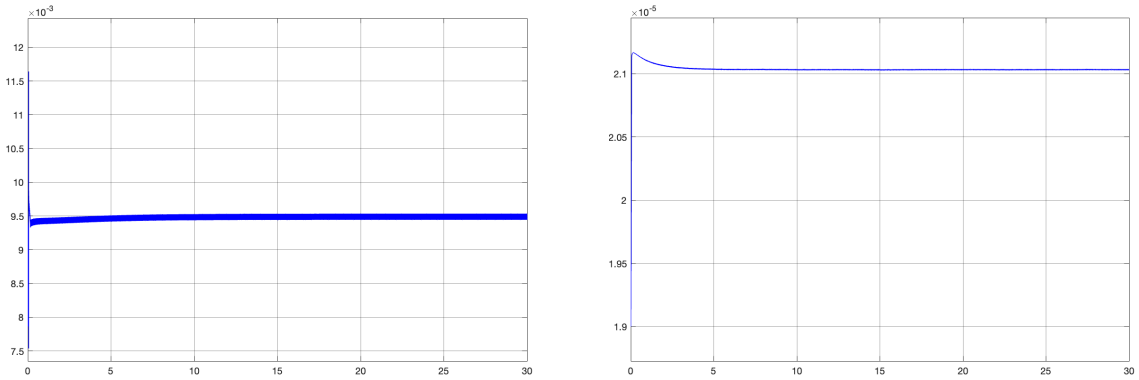


(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.26: RLS estimation with test case 4. (Red: Actual value, Blue: Estimated) Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

EKF

The EKF is very fast at converging to the right value on this case. The transient phase does not last long, and they are hardy visible in graphs in figure 5.27. However, the drawback is still that there is an offset in the estimated values by a margin of around 8% for the resistance and 11% for the inductance. Both parameters settle around the 3s mark.



(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.27: EKF estimation with test case 4. Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

5.3.5 Test case 5 - Acceleration limit

In this test case there is an acceleration limit put on the system when it is accelerating towards the reference speed. This is done to have a more controlled start-up of the motor. The acceleration speed limit is set to 30000 rpm/s. The acceleration limit gives a small oscillation in the transient phase as seen in both i_q current and the motor torque graphs in figure 5.24, however both settle after around 0.5s. The step up to the speed reference is seen to be slower than previous cases and there is also no overshoot in the step, indicating a more controlled start up.

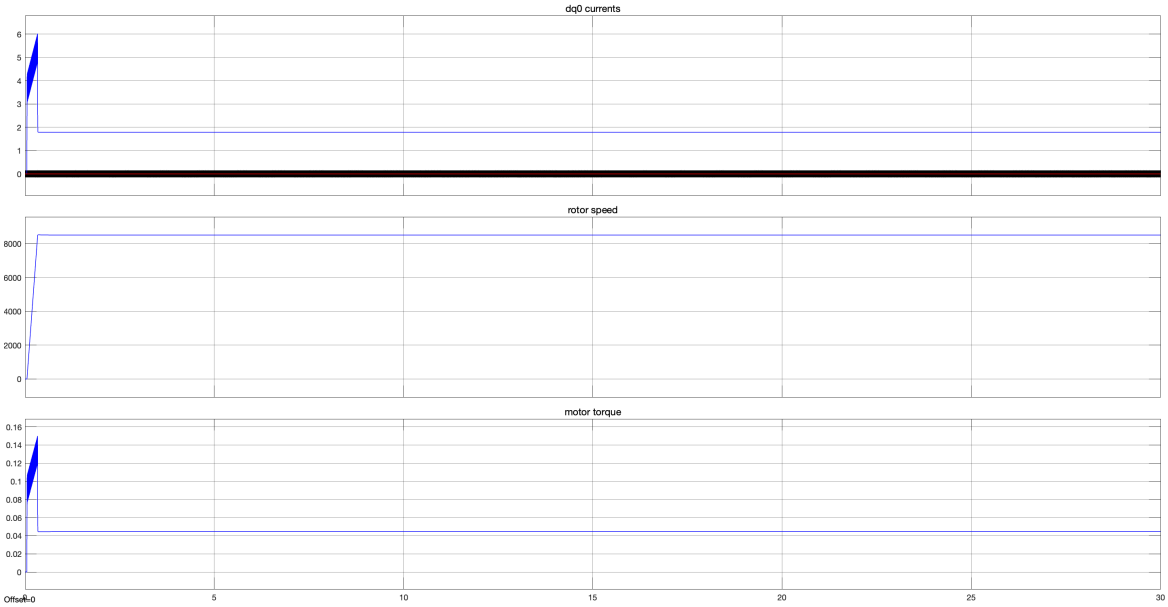
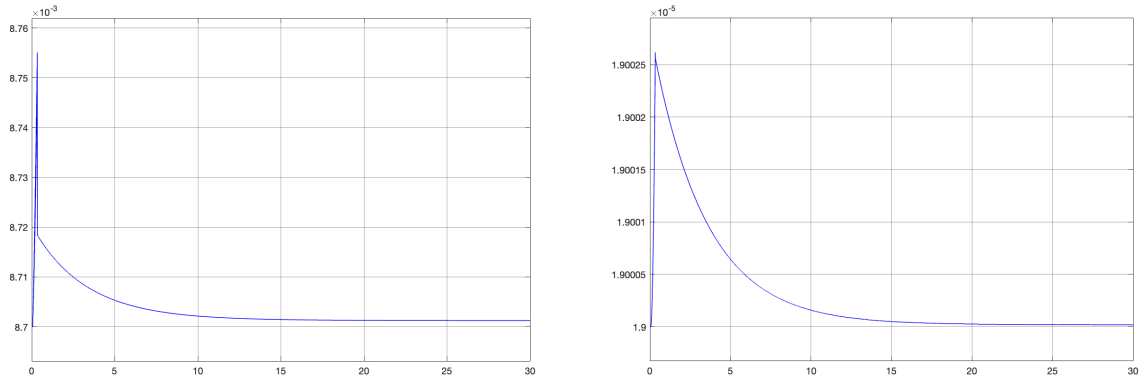


Figure 5.28: Graphs of dq-currents in *ampere* (Red: d-current, Blue: q-current), speed in rpm and torque in Nm with test case 5

MRAS

When it comes to acceleration limit, there is no major setback for the MRAS estimator. The resistance and inductance converge towards the actual value within $10 - 15$s and the estimation error is lower than 0.1% for both parameters.



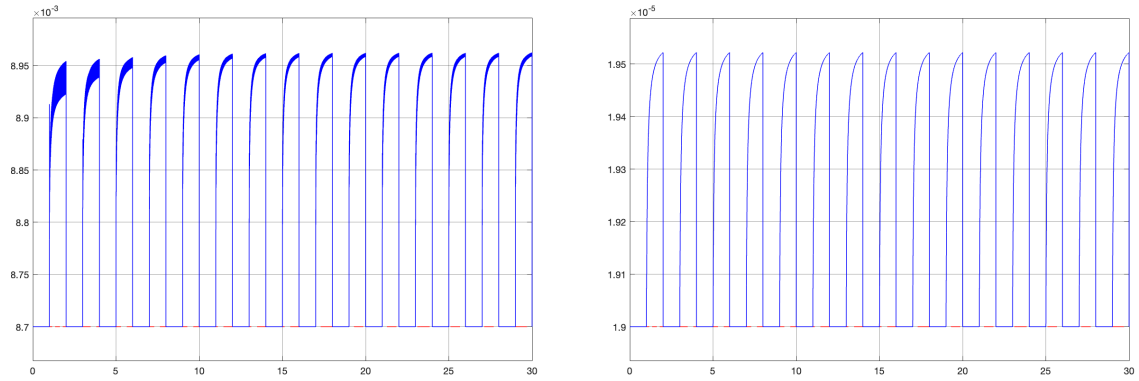
(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.29: MRAS estimation with test case 5. Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

RLS

With an acceleration limit, the RLS estimation has a bigger oscillation in the pulses of the resistance estimation. This can be seen in figure 5.30, on the first four pulses. After that the value stabilizes. The inductance however has a constant estimation

value throughout and is not much affected by the acceleration limit. Both parameters converge to its final value within 1s and the error is around 3% for both cases.

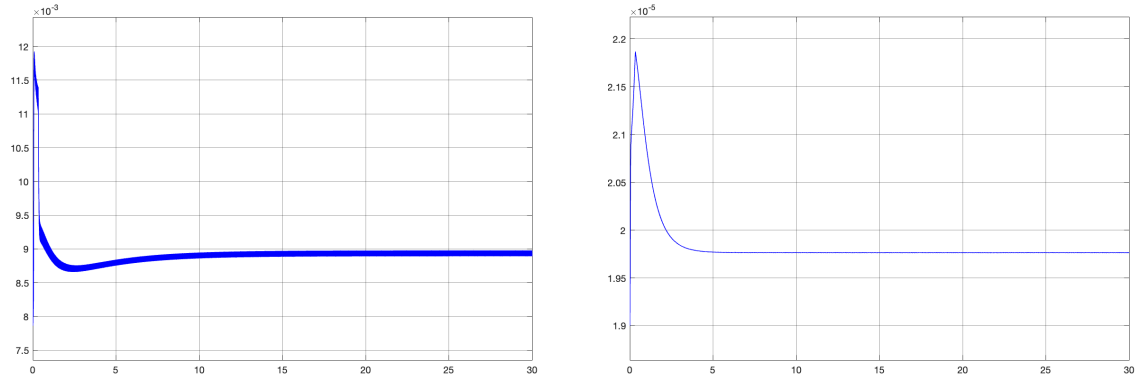


(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.30: RLS estimation with test case 5. (Red: Actual value, Blue: Estimated) Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

EKF

There is a big overshoot in the transient states of both parameters in the EKF estimation but they go down quickly as well. Under 1s for the resistance and 3s for the inductance. There are no major issues with the parameter estimation in this test case either, but there is still some small off sets in both R_s and L_s compared to the actual value. For the resistance case the error is around 1.7% and the inductance is 4.2%.



(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.31: EKF estimation with test case 5. Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

5.3.6 Test case 6 - Resistance Variation

In this test case there is change in the value of the resistance after 1.5s in the motor model. The idea is to test the estimators capability to estimate completely different values online. The resistance value is increased by 100%. This is one the most impor-

tant tests for the OPEA's since it tests the estimation capability. Current, speed and torque are the same as they are section 5.2, see figure 5.32.

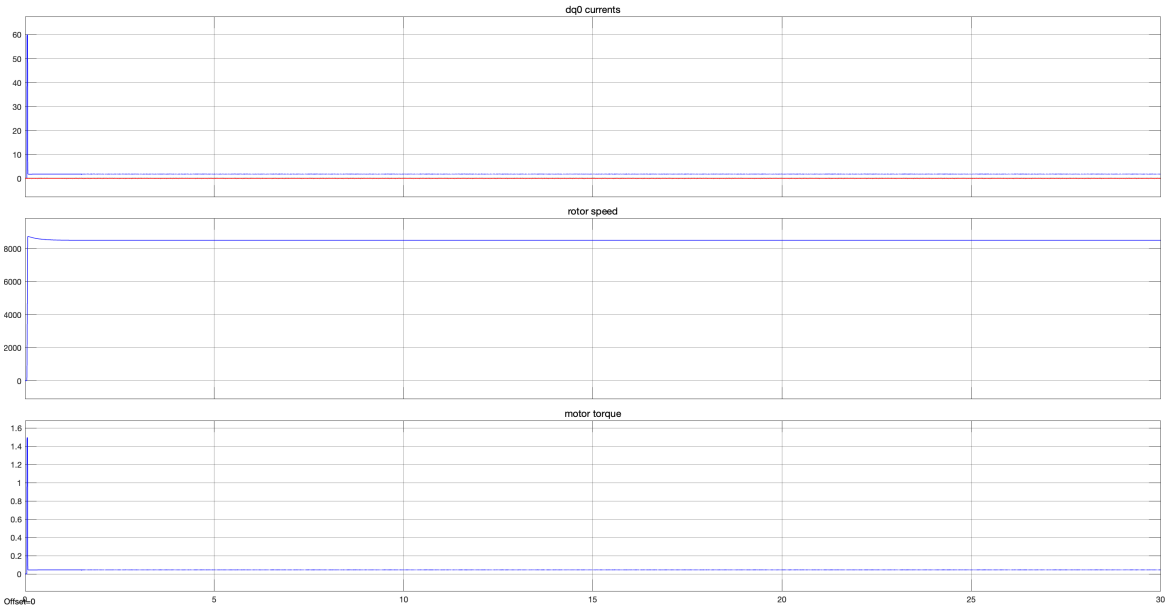
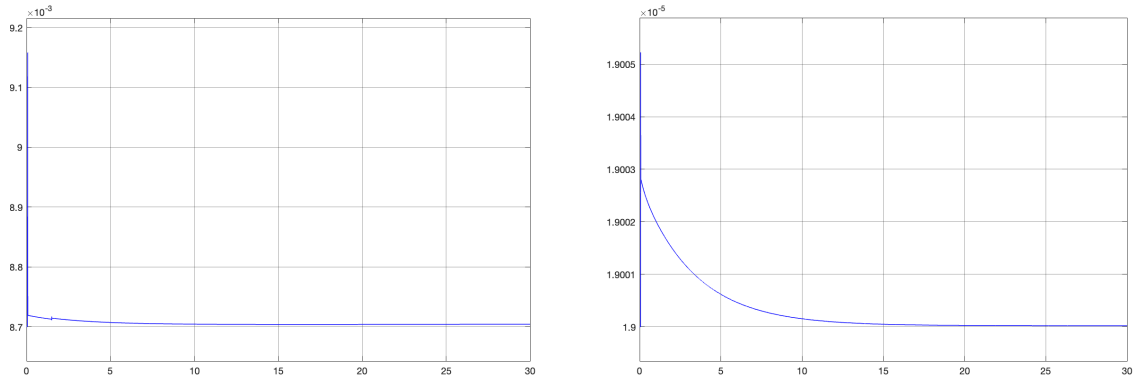


Figure 5.32: Graphs of dq-currents in *ampere* (Red: d-current, Blue: q-current), speed in rpm and torque in Nm with test case 6

MRAS

It is clear that the MRAS estimator cannot work when there is a step in the resistance value. As seen in 5.33, the convergence value of R_s is still the original value of 0.0087 when it should be the double 0.0174. As for the inductance estimation, it is not affected by the change in the resistance variation and it settles after 18s.

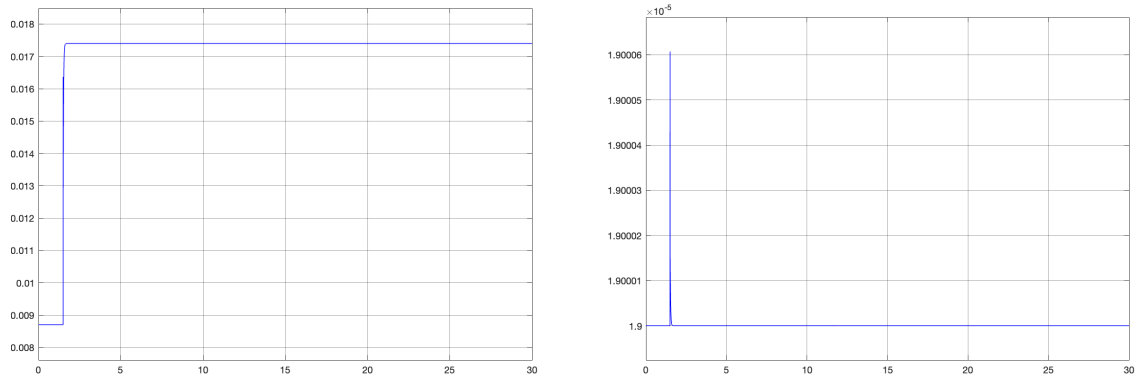


(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.33: MRAS estimation with test case 6. Actual values: $R_s = 8.7 * 10^{-3}$ Ω before 1.5s and $1.74 * 10^{-2}$ Ω after 1.5s, $L_s = 1.9 * 10^{-5}$ H

Figure 5.34 shows the MRAS estimation for with position feedback from the motor. There have also been slight changes to the PI parameters of the adaptive law in 3.22.

It is seen that this test case works with position feedback. The convergence time is 0.2s for the resistance and and the error is less than 0.5%. The figure also shows an overshoot of the inductance value after 1.5s. This is when the resistance change occurs. However, the overshoot is smaller than 0.01%, meaning it is negligible.

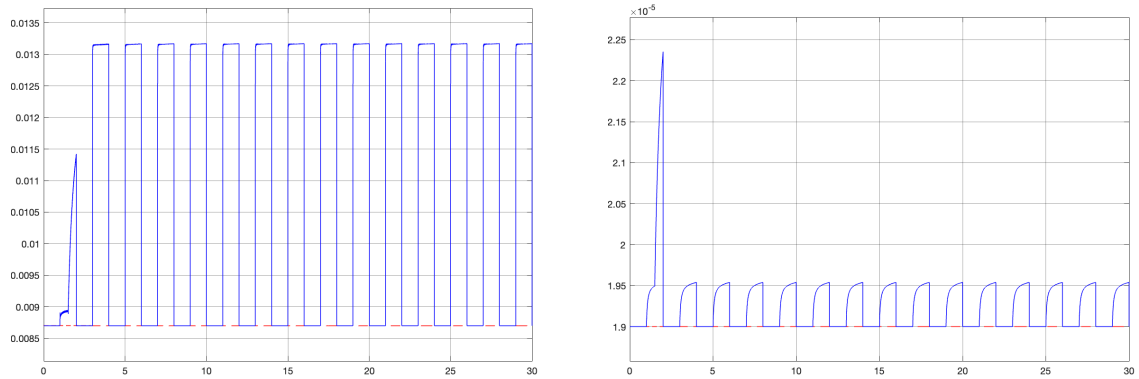


(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.34: MRAS estimation with test case 6 and actual position feedback from the motor. Actual values: $R_s = 8.7 * 10^{-3}$ Ω before 1.5s and $1.74 * 10^{-2}$ Ω after 1.5s, $L_s = 1.9 * 10^{-5}$ H

RLS

The RLS estimator, as the other cases, measures the parameters values in pulses. In the first pulse it can be seen that both resistance and inductance values are estimated somewhat correct to the actual value. After 50% the first pulse, the resistance and inductance shoots up. The second pulse and the pulses afterwards are stable and all converge. However, it is noteworthy that although the RLS estimator does react to the change in the resistance value, the convergence value has an offset of 23.5%. The inductance value stabilizes and converges to the right value with a smaller offset of 3%.

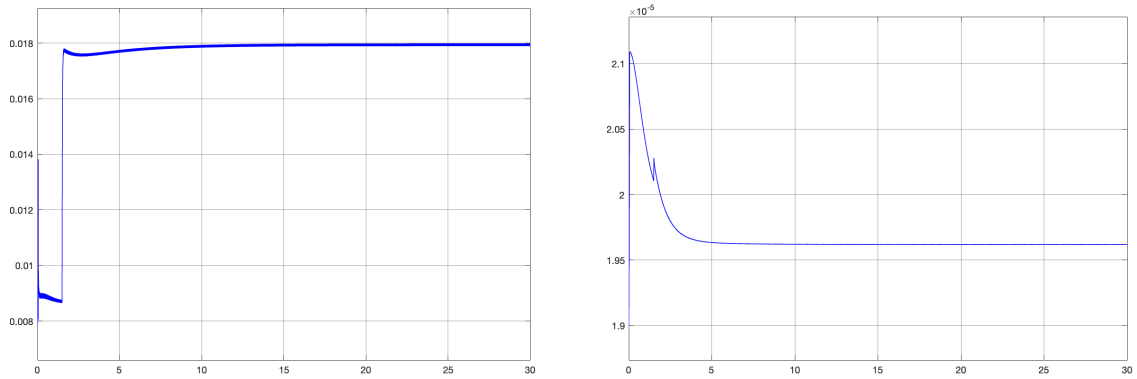


(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.35: RLS estimation with test case 6. (Red: Actual value, Blue: Estimated) Actual values: $R_s = 8.7 * 10^{-3}$ Ω before 1.5s and $1.74 * 10^{-2}$ Ω after 1.5s, $L_s = 1.9 * 10^{-5}$ H

EKF

The EKF is the only estimator that can handle online estimation when the resistance change during drive and with the sensorless control scheme. The convergence times is around 8s for the resistance and 5s for the inductance. Further, the convergence value is almost identical to the actual value with an offset of 1.7% in the resistance and 3.2% in the inductance. There is a small disturbance in the inductance estimation during the resistance change, at 1.5s time mark, but it is negligible for the overall estimation.



(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.36: EKF estimation with test case 6. Actual values: $R_s = 8.7 * 10^{-3}$ Ω before 1.5s and $1.74 * 10^{-2}$ Ω after 1.5s, $L_s = 1.9 * 10^{-5}$ H

5.3.7 Test case 7 - Low speed response

Test case 7 switches the reference speed periodically between 8500rpm and 150rpm. This is to test the low speed response of the estimators and to see if the estimators can handle lower speeds of the motor and variations in speed. In figure 5.37, in the speed graph it is seen that the speed reference is changed every second. When the speed steps up from lower speed to higher, there is a small increase in both i_q current and motor torque. The current increases with 10A for a small moment and the torque increases with 0.25Nm. However, during the step down in speed, there is a big momentous dip in both torque and current to negative values.

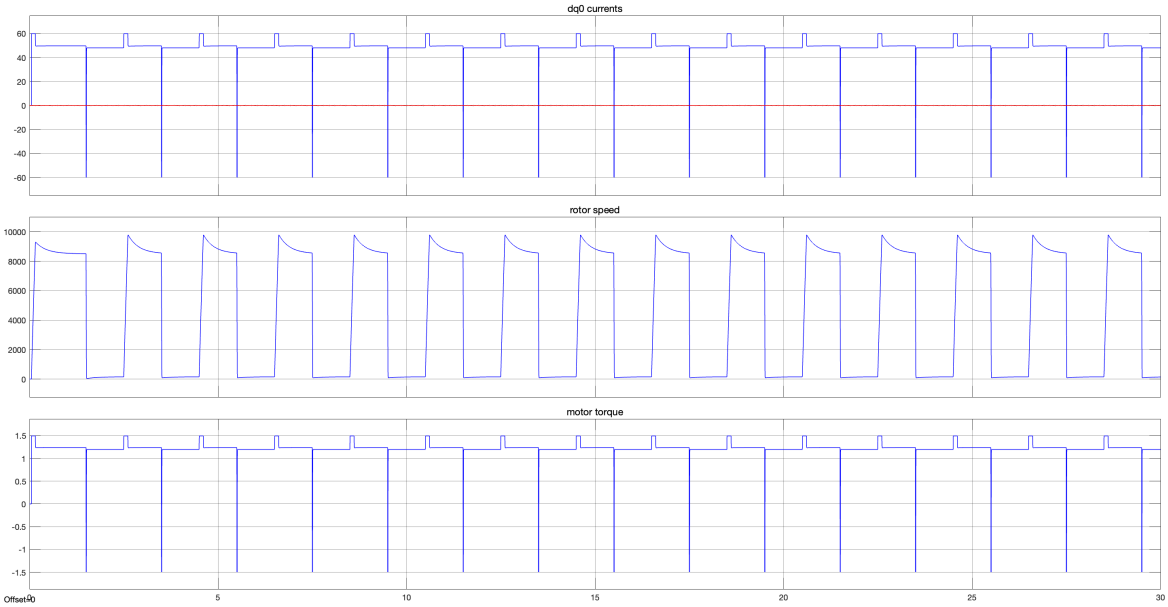
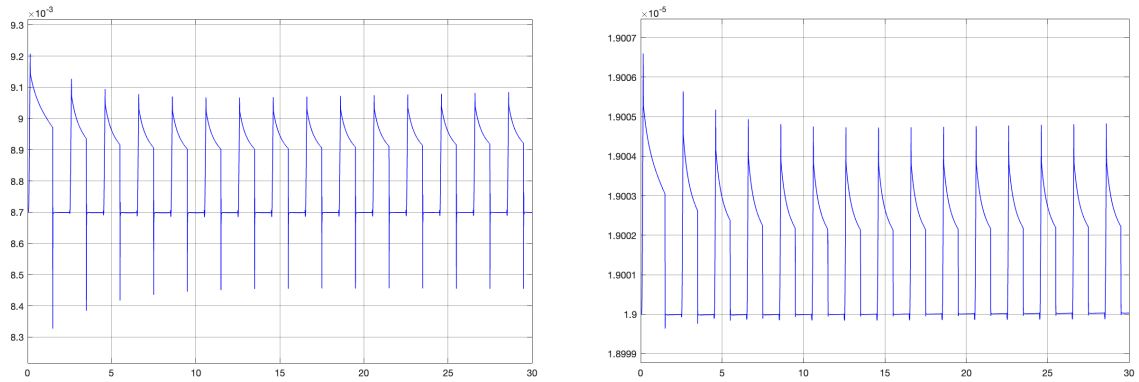


Figure 5.37: Graphs of dq-currents in *ampere* (Red: d-current, Blue: q-current), speed in rpm and torque in Nm with test case 7

MRAS

The MRAS estimator fluctuates and overshoots when the speed switches on both parameter estimations. Although the fluctuations jump quite a bit, they do not vary significantly enough for the result to be deemed faulty. There is an overshoot of around 5% for the resistance when the speed switching occurs and the values are still relevant. When it comes to the estimation of L_s, the jumps are very small and the overshoot has an error of less than 0.1%. The estimation error overall for the resistance is around 1.9% and less than 0.1% for the inductance.

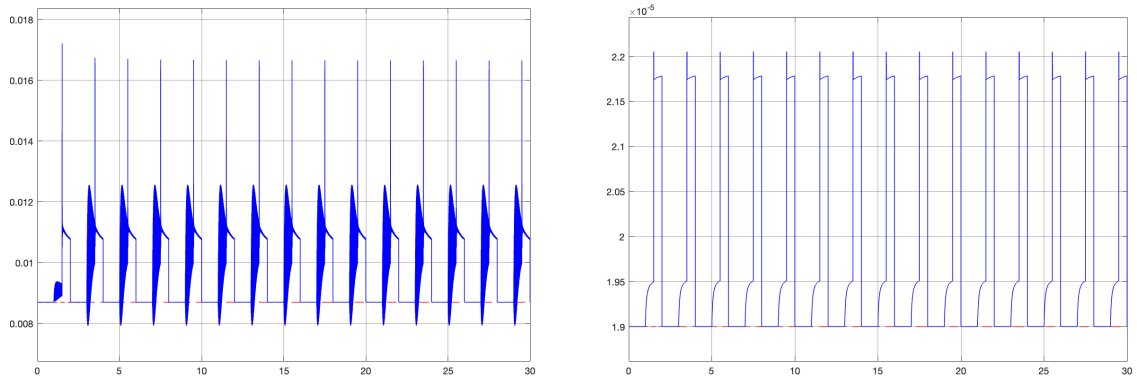


(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.38: MRAS estimation with test case 7. Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

RLS

The RLS estimator is very bad at handling low speed responses and the constant switching as well. Both in terms of resistance and inductance estimation, the result oscillate and overshoot at a very high amplitude of more than 100% for the resistance and around 17% for the inductance. This makes it quite hard to have a proper reading on the estimation. The resistance values also oscillates around the wrong value which makes it hard to low pass filter the signal as well. The inductance value shows no significant way either on proper readings.

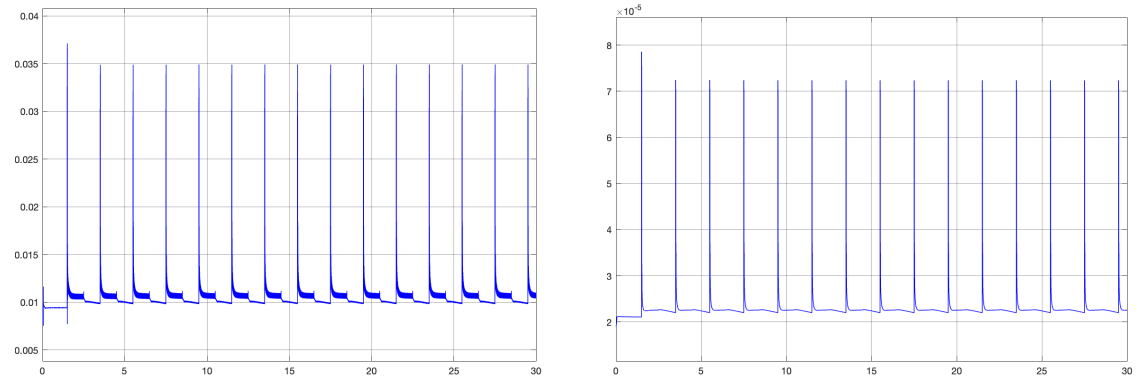


(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.39: RLS estimation with test case 7. (Red: Actual value, Blue: Estimated) Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

EKF

The EKF is relatively bad at estimating parameters on this case. There are big overshoots that occur when the switching takes place, and the there are small oscillations in the R_s estimation as well. Although there is a certain offset as well in both parameters, by low pass filtering the estimations, relevant parameter value can be read. For the resistance estimation the convergence value is reached around 2s with an error of 20%. The inductance settles after 0.5s with an error of 21%.



(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.40: EKF estimation with test case 7. Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H

5.3.8 Test case 8 - Inductance variation

In this test case, the inductance is increased by 100% after 1.5s. This test case shows how the well the estimators can give an updated value of the inductance. An increase of 100% is very high, and practically never occurs. A more realistic case would be an inductance change of around ±20%. However, 100% change is done in order to test the robustness of the estimators. A load is also applied from the start of the simulation to excite the system.

Further, it is seen in figure 5.41, that the MRAS position estimator cannot handle the inductance change. After 1.5s the speed error oscillates between a value of around 0.1 and the position estimator oscillates with an amplitude of around 0.5, with a constant offset of -2. This has not been the situation in previous test cases, therefore for this test case, actual position feedback is used in order to show how well the estimators perform.

Moreover, current, speed and torque behaves the same way as it does in test case 4 with the difference being that there is a small disturbance when in the inductance value is changed. However, this disturbance is only momentarily.

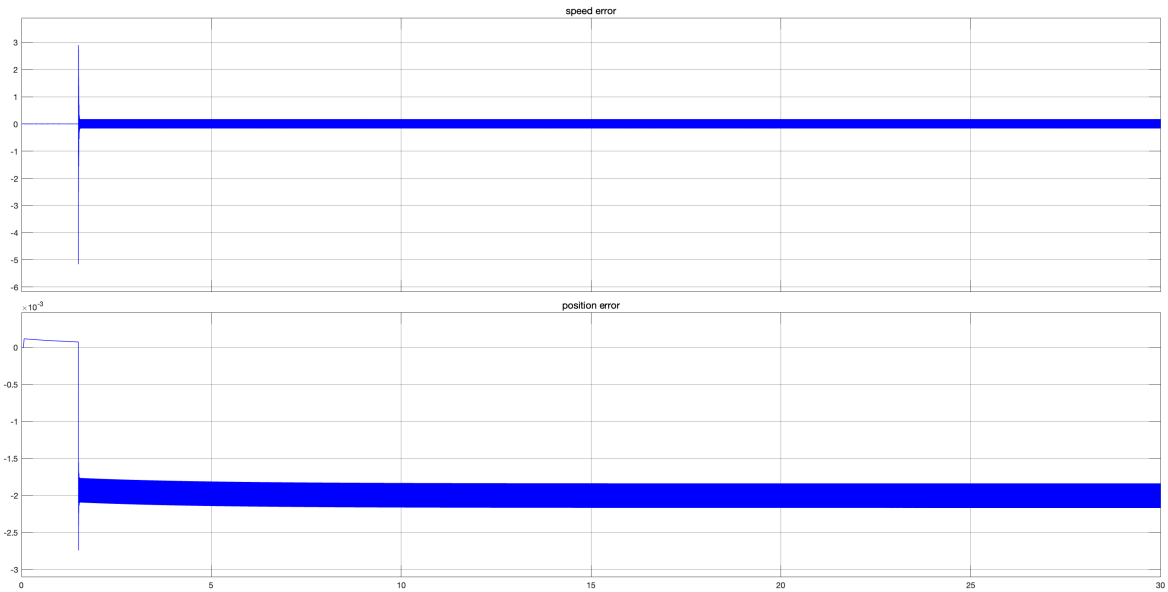


Figure 5.41: Estimation error of speed and position from the MRAS estimator with a change in the inductance. Speed error in mechanical speed ω_r (rad/s) and position error in mechanical position θ_r (rad)

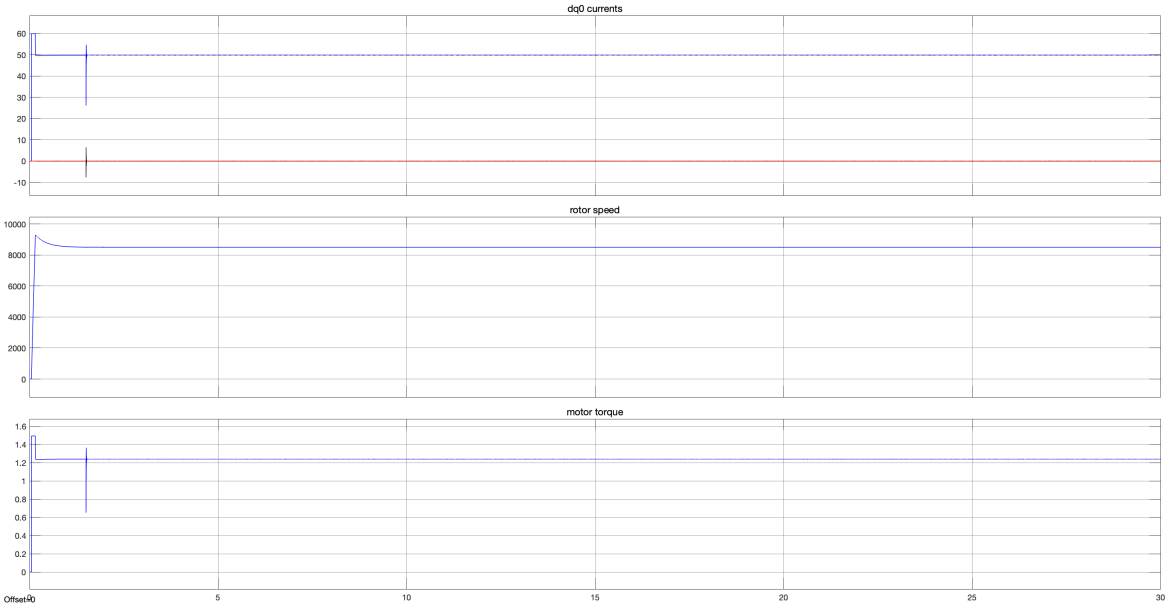
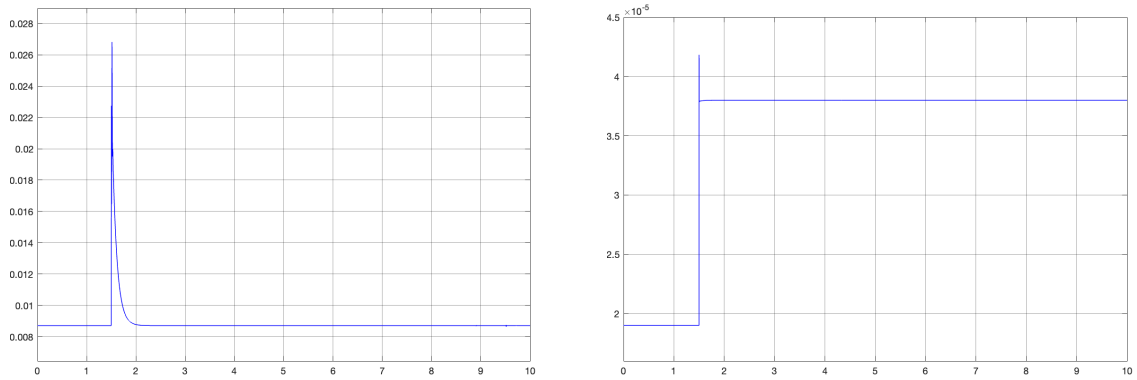


Figure 5.42: Graphs of dq-currents in *ampere* (Red: d-current, Blue: q-current), speed in rpm and torque in Nm with test case 8

MRAS

In test case 6, it is seen that with sensorless control, the resistance value does not update online. The same situation occurs for the inductance estimation. Thereby, for this test case, the MRAS estimators is using the actual position feedback. The resistance value has an overshoot of 210% relative to the actual value when the inductance change occurs. However, the resistance value settles again after 0.7s and only has an error of less than 0.1%. The same accuracy is the case with the inductance estimation and the settling time is around 1.5ms.



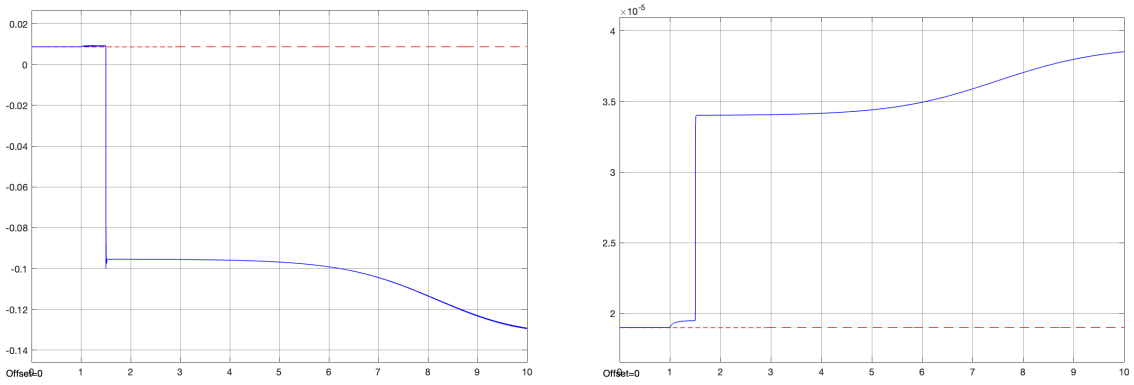
(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

Figure 5.43: MRAS estimation with test case 8. Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H before 1.5s and $3.8 * 10^{-4}$ after 1.5s

RLS

For the RLS estimation with inductance change, the resistance estimation is off by 1300% after 10s simulation. This indicates that the RLS estimator cannot handle online estimation with inductance change. Even during the initial convergence at time mark 1.5s, the estimation error is around 800%. Because the resistance estimation is off by this margin, it would not help to give a pulsating signal to the estimator as it is done in previous cases. It is also due to the fact that the initial convergence value after the inductance change is a negative value.

The inductance estimation however converges to the right value, with an offset of around 3%, however the settling time is around 10s.

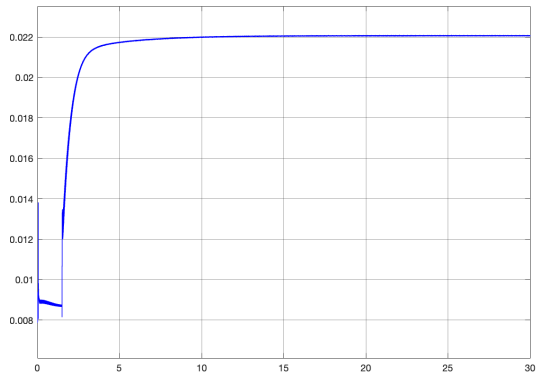


(a) Resistance R_s (Ω) **(b)** Inductance L_s (H)

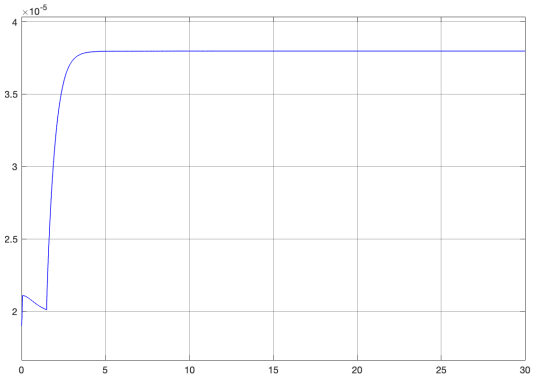
Figure 5.44: RLS estimation with test case 8. (Red: Actual value, Blue: Estimated) Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H before 1.5s and $3.8 * 10^{-4}$ after 1.5s

EKF

The EKF algorithm works well with the inductance change during operation. The estimated inductance value settles after around 3s with an error of less than 0.01% with the sensorless case. This is seen in figure 5.45. However, the resistance value also changes and gets an error of around 150%. Comparing this result with figure 5.46, the inductance value gets updated in the same way as in the sensorless case but the resistance value stays the same and only has an estimation error of 1.76%.

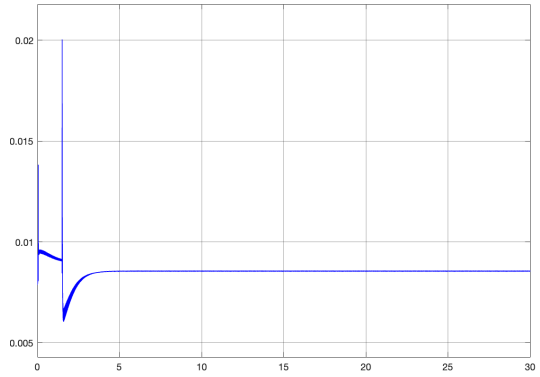


(a) Resistance R_s (Ω)

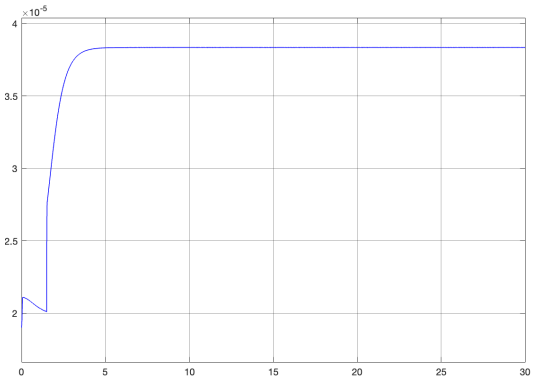


(b) Inductance L_s (H)

Figure 5.45: EKF estimation with test case 8 and sensorless control. Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H before 1.5s and $3.8 * 10^{-4}$ after 1.5s



(a) Resistance R_s (Ω)



(b) Inductance L_s (H)

Figure 5.46: EKF estimation with test case 8 and position feedback from the motor. Actual values: $R_s = 8.7 * 10^{-3}$ Ω, $L_s = 1.9 * 10^{-5}$ H before 1.5s and $3.8 * 10^{-4}$ after 1.5s

5.4 Summary tables

Below are the results of the estimators and the test cases presented in table form. For each OPEA there are two tables, one for the estimation error of the test cases and a second one for the convergence time. The results will further be discussed and analyzed in chapter 6.

MRAS

Table 5.1: Table of results from the test cases of the MRAS estimation with error percentages. **position feedback*

Test Cases	MRAS		
	Working	R_s *Error %*	L_s *Error %*
1 - Step Load	Yes	0.1	< 0.1
2 - Periodic Load	Yes	0.5	< 0.1
3 - Transient Load	Yes	0.3	< 0.1
4 - Start Load	Yes	0.3	< 0.1
5 - Acceleration Limit	Yes	< 0.1	< 0.1
6 - Resistance Variation	No	n/a	n/a
7 - Low Speed	Yes	1.9	< 0.1
8 - Inductance Variation*	Yes	0.1	0.1

Table 5.2: Table of results from the test cases of the MRAS estimation with the convergence time. **position feedback*

Test Cases	MRAS		
	Working	R_s *conv. time*	L_s *conv. time*
1 - Step Load	Yes	11s	11s
2 - Periodic Load	Yes	5s	10s
3 - Transient Load	Yes	17s	< 17s
4 - Start Load	Yes	15s	< 15s
5 - Acceleration Limit	Yes	< 15s	< 17s
6 - Resistance Variation	No	10s	18s
7 - Low Speed	Yes	n/a	$< n/a$
8 - Inductance Variation*	Yes	0.6s	1.5ms

RLS

Table 5.3: Table of results from the test cases of the RLS estimation with error percentages. **position feedback*

Test Cases	RLS		
	Working	R_s *Error %*	L_s *Error %*
1 - Step Load	No	n/a	n/a
2 - Periodic Load	No	n/a	n/a
3 - Transient Load	No	n/a	n/a
4 - Start Load	Yes	6.0	3.2
5 - Acceleration Limit	Yes	3	3.16
6 - Resistance Variation	Yes	23.56	3.16
7 - Low Speed	No	n/a	n/a
8 - Inductance Variation*	No	n/a	n/a

Table 5.4: Table of results from the test cases of the RLS estimation with the convergence time. **position feedback*

Test Cases	RLS		
	Working	R_s *conv. time*	L_s *conv. time*
1 - Step Load	No	n/a	n/a
2 - Periodic Load	No	n/a	n/a
3 - Transient Load	No	n/a	n/a
4 - Start Load	Yes	0.8s	1s
5 - Acceleration Limit	Yes	0.7s	1s
6 - Resistance Variation	Yes	< 0.1s	0.8s
7 - Low Speed	No	n/a	n/a
8 - Inductance Variation*	No	n/a	n/a

EKF

Table 5.5: Table of results from the test cases of the EKF estimation with error percentages. **position feedback*

Test Cases	EKF		
	Working	R_s *Error %*	L_s *Error %*
1 - Step Load	Yes	2.5	6.5
2 - Periodic Load	Yes	3.45	4.95
3 - Transient Load	Yes	8.0	9.4
4 - Start Load	Yes	8.0	11.0
5 - Acceleration Limit	Yes	1.72	4.2
6 - Resistance Variation	Yes	1.7	3.2
7 - Low Speed	Yes	20.7	21.05
8 - Inductance Variation*	Yes	1.67	< 0.01

Table 5.6: Table of results from the test cases of the EKF estimation with the convergence time. **position feedback*

Test Cases	EKF		
	Working	R_s *conv. time*	L_s *conv. time*
1 - Step Load	Yes	4s	5s
2 - Periodic Load	Yes	0.5s	3s
3 - Transient Load	Yes	2s	5s
4 - Start Load	Yes	3.5s	3s
5 - Acceleration Limit	Yes	11s	3
6 - Resistance Variation	Yes	8.5s	5s
7 - Low Speed	Yes	2s	0.5s
8 - Inductance Variation*	Yes	1.5s	3.5s

6 Discussion

This chapter comments on the results and gives an analysis of how to interpret the results. It also touches on the matter of how well the initial questions have been answered as well as how to improve results in future work

General discussion

The simulations of the different estimators showed how well the resistance and the inductance could be estimated. It is made very clear that the actual position feedback control scheme had the definite better overall convergence capability, both in terms of convergence value and the convergence time. It is also interesting to note that for this case, the MRAS estimator did not show any difference between the estimated value and actual value. This is because the simulation is dependent on a model of the motor which will act as an ideal model. Both the plant model and the adaptive model in the MRAS "box" in the simulation are the same. It is therefore hard to emulate real life application of the MRAS estimation for the position feedback case. The RLS estimation works very well with the position feedback control. The convergence time is almost negligible, and the value is reached instantly. This is the case for both R_s and L_s. The error in resistance estimation is only 1.3% and for the inductance it is only 0.2%. It is the same thing with the EKF as with the RLS. A very fast convergence time and a value very close to the actual value with almost no error.

Comparing the position feedback case to the sensorless case, both MRAS and the EKF converge to very good values on both parameters, though at a much slower time. The speed and position error caused by the MRAS position estimator in the sensorless case, as seen in 5.8, could be a factor in why the values converge slower. The same thing happens for the RLS case, the pulses converge to a more stable and accurate value after some seconds. When the position error of the MRAS estimator goes to zero, the estimation values of all parameter estimator reach a steady state and a value closer to the actual value.

Test cases - convergence time and estimation error

Table 5.1 to table 5.6 summarise how well each OPEA's worked with each test case in terms of estimation error and convergence time. The table's are the results from the sensorless control scheme on test cases $1-7$ and actual position feedback on test case 8.

The results from the test cases of the MRAS estimation scheme shows that it handles all the test cases expect the one where the resistance value is changed during operation. This is a crucial test to pass in order to be deemed a reliable parameter estimator. The

adaptive law for the resistance given in 3.22 shows that the adaptive law determines the resistance estimation based on the initial value of the resistance and the error in the current estimations.

Comparing this to test case 6 with position feedback from the motor, it is seen that the resistance estimation works very well. This means that the MRAS OPEA is not working for the case with MRAS position estimator. One of the factors for this error could be due to the slow diminishing of the position error from the estimator, as given in 5.8. However, actual position feedback from the motor is not always feasible, which means that a better position estimator could improve the results of the MRAS parameter estimator.

Further, it is also noteworthy that the estimation errors are very small for the test cases where the MRAS model work. This shows that it is performing good in conditions where there are disturbances and load applied during operation of the motor. One of the major benefits of the MRAS model is that it operates well during low speed, and the estimation error of the resistance is only around 1.9%. This can be compared to the EKF which has a error in that test case of around 20%. The RLS cannot handle this case at all.

Moreover, when it comes to the RLS case it is not very good at estimating the parameters in this sensorless application. There are many important test cases that it cannot handle and the results are not sufficiently good for the result to be handled. One interesting case that it handles is test case 6 where it does switch to a higher resistance value, but it has an estimation error of around 23%, which is far too high for the estimation to be deemed sufficiently good. One of the different approaches that can be taken in the RLS scheme is to try different solutions to the wind-up problem. In this thesis, the idea was to estimate the parameters in pulses, and only when the motor is at some sort of steady state. The pulses also reset the covariance matrix each time, which consequently leads to new estimations being done every pulse. The choice of forgetting factor can be varied in many different ways as well. In this case the forgetting factor was chosen with trial and error, which is not always the best case. There could be a better forgetting factor choice for the test cases to succeed and for the errors to be lower. In addition, there are other excitation signals that can be chosen for the RLS estimator. In this case, the signal injected into the i_d current is in a stair case shape, which was deemed to be the best one for this case. However, there are other types of signals as well that could be tested.

Furthermore, the EKF estimator is the only one that handles all the test cases and achieves satisfactory result. Although this comes at a price of larger errors in both resistance and inductance estimation. It is also very high for test case 7, which means that the EKF is not suitable for conditions in lower speed. Although, a major factor why this is not working is because the i_d current switches to negative values when the speed is lowered. The current at the peak of the switch reaches value of negative 60A. One way to reduce this error could be to filter the current signals, however it is not clear exactly how this should be done as the filtration could impact the other test cases as well. Moreover, a major factor that speaks for the EKF is that it handles test case 6 very well with sensorless control with minor errors in both resistance and inductance. Here as well, the choice of excitation signal gives different results for each

test case. Although, by simulation results, the chosen excitation signal gives the best overall results it is not proven to be the best one.

Summary

To summarise, In terms of how feasible the estimation algorithms are for this PMSM application, both the EKF and MRAS show promising results. They work in various different test cases and they converge rather well to the given parameters. However, for the MRAS estimator, it is not working well with the sensorless case on the test cases with parameter variations. This is a major drawback of the estimator. Although with position feedback, the results for the MRAS estimations were very accurate. As stated above, a better position estimator could improve the MRAS parameter estimator, but it is nonetheless not working in this case.

Moreover, to contrast the EKF and MRAS to the RLS, it has a hard time at reaching reasonable results for all the test cases. It also has wind-up problems that are hard to solve for the operating points that this motor experiences. It is also seen that the RLS has a much better estimation result when it comes to the regular control scheme and not the sensorless one. This shows that it is not applicable for this case.

Furthermore, the convergence time for the EKF and the MRAS are good, although they perform much better at the regular scheme than the sensorless case. However, in general the simulations show that the EKF has a much better convergence time than the MRAS, but this as seen in table 5.5, comes at the price of larger error percentage compared to the MRAS.

The third question that needed to be investigated is how hard they are to implement and what are the different estimators computation complexity. Since this thesis has not made a deep research in the computation of the estimators, they are classified on a very basic level. They are categorized in a simple three level categorization; low, medium and high. Since the RLS does not handle the test cases well, it should not be considered, however it has a medium computational complexity. The MRAS thrives on its simplicity and the simple design it is implemented with, therefore it has the least computational burden on the processing unit, thereby giving it the categorization of low. To contrast that to the EKF, it is the most computationally heavy estimation method of them all, meaning category high, but it is also the most robust one in terms of how well it handles the test cases. Thereby it should be taken into consideration when choosing estimation method.

For the purpose of the motor in this thesis, the EKF seems like the most suitable OPEA to be implemented, given the robustness of it and the capability of handling different test cases. However, it should be noted that there are many more factors that can affect the estimation process in practical implementation. This thesis only simulates the OPEA's and factors such as sampling frequency are not investigated, which affects the dynamics and bandwidth.

7 Conclusion & future work

In this chapter the work done in this thesis is concluded and wrapped. Some points for future work is also included that can improve this work.

Conclusion

The aim of this thesis is to investigate different parameter estimation algorithms and to simulate selected few of them. The investigation is based on three main points; What different type of OPEA's are available, how well and fast do the converge to the estimated values and how computationally complex are they. These questions are further explored in chapter 5 and 6.

Chapter 2 introduces the general theory behind the PMSM and the FOC control system. The difference between the sensorless control and the position feedback control is also explained. Moreover the mathematical model of the PMSM is explained as well as the different transforms used to get to dq coordinates.

Further, chapter 3 introduces different OPEA's and gives a general theory of how they work and how they are implemented along with the mathematical model of them. A brief introduction of other OPEA's than the simulated ones is also presented. An evaluation of which OPEA's to simulate is also given.

Chapter 4 explains the simulation process and how all the components are built in the Simulink environment. The parameters for every model and the modelling structure is also presented.

Results from the simulation and all the different test cases are presented in chapter 5 and they are further commented on and analyzed in chapter 6.

Based on the analysis presented in chapter 6, the OPEA that is deemed most suitable for this motor is the extended kalman filter. This is because it handles different operating points of the motor and is the only that is robust enough to work for all the test cases.

Future work

Future work, that builds upon this thesis, is to investigate the different OPEA's in a simulation environment where noise and disturbance is added. It is also interesting to investigate the OPEA's in an experimental setup and to test it on the actual motor. This thesis focuses on models and model based simulation. Real applications are prone to deviate from models and both noise and disturbance can be major factors that limit

the OPEA's.

Further, it should be investigated more thoroughly how to implement the OPEA's in a DSP and how computationally heavy the calculations are for each OPEA. It is beneficial to quantify the computation time in order to better understand the algorithms and if they can be tweaked for certain operational points on the motor.

Bibliography

[1] Mats Alaküla, Per Karlsson, and Hans Bängtsson. *Power Electronics. Devices, Converters, Control and application*. Lund University, 2019.

[2] Mats Alaküla, Olof Samuelsson, and Lars Gertmar. *Elenergiteknik*. Industriell Elektroteknik och Automation, Lunds Tekniska Högskola, 2011.

[3] T. Boileau, B. Nahid-Mobarakeh, and F. Meibody-Tabar. "On-Line Identification of PMSM Parameters: Model-Reference vs EKF". In: *2008 IEEE Industry Applications Society Annual Meeting* (2008), pp. 1–8.

[4] G. Feng, C. Lai, and N. Kar. "A Novel Current Injection-Based Online Parameter Estimation Method for PMSMs Considering Magnetic Saturation". In: *IEEE Transactions on Magnetics* 52 (2016), pp. 1–4.

[5] F. Giri. "AC electric motors control: advanced design techniques and applications". In: 2013.

[6] Naresh Gujjula and G. Laxminarayana. "Field-Oriented Control of PMSM Drive Based on SVPWM Using MATLAB". In: 2014.

[7] S. Gunnarsson. "Combining tracking and regularization in recursive least squares identification". In: *Proceedings of 35th IEEE Conference on Decision and Control* 3 (1996), 2551–2552 vol.3.

[8] Mohammad Hosein Holakooie, A. Taheri, and M. Sharifian. "MRAS Based Speed Estimator for Sensorless Vector Control of a Linear Induction Motor with Improved Adaptation Mechanisms". In: *Journal of Power Electronics* 15 (2015), pp. 1274–1285.

[9] S. Iderus. "The field oriented control of a permanent magnet synchrounous motor (PMSM) by using fuzzy logic". In: 2014.

[10] V. C. Ilioudis. "Sensorless Control of Permanent Magnet Synchronous Machine with Magnetic Saliency Tracking Based on Voltage Signal Injection". In: 2020.

[11] Y. Jia. "Recursive Least Squares Estimation". In: 2015.

[12] A. Khlaief, M. Boussak, and A. Chaari. "A MRAS-based stator resistance and speed estimation for sensorless vector controlled IPMSM drive". In: *Electric Power Systems Research* 108 (2014), pp. 1–15.

[13] Youngjoo Kim and H. Bang. "Introduction to Kalman Filter and Its Applications". In: 2018.

[14] Xinyue Li and R. Kennel. "Comparison of state-of-the-art estimators for electrical parameter identification of PMSM". In: *2019 IEEE International Symposium on Predictive Control of Electrical Drives and Power Electronics (PRECEDE)* (2019), pp. 1–6.

[15] M. Linderoth et al. "Initialization of the Kalman filter without assumptions on the initial state". In: *2011 IEEE International Conference on Robotics and Automation* (2011), pp. 4992–4997.

[16] Kan Liu et al. “Comparison of two novel MRAS based strategies for identifying parameters in permanent magnet synchronous motors”. In: *International Journal of Automation and Computing* 7 (2010), pp. 516–524.

[17] Qian Liu and K. Hameyer. “A fast online full parameter estimation of a PMSM with sinusoidal signal injection”. In: *2015 IEEE Energy Conversion Congress and Exposition (ECCE)* (2015), pp. 4091–4096.

[18] Xiaojun Liu et al. “Speed estimation and Parameters Identification Simultaneously of PMSM Based on MRAS”. In: *WSEAS Transactions on Systems and Control archive* 12 (2017).

[19] L. Gustaffson M. Olsson. *Robust on-line estimation*. 1999.

[20] Henrik Neugebauer. “Parameter identification of a permanent magnet synchronous motor”. In: 2012.

[21] Saiful Bahri Samsuri et al. “Computational cost analysis of extended Kalman filter in simultaneous localization and mapping (EKF-SLAM) problem for autonomous vehicle”. In: *ARPN journal of engineering and applied sciences* 10 (2015), pp. 7764–7768.

[22] Hyun-Woo Sim, June-Seok Lee, and Kyo-Beum Lee. “On-line Parameter Estimation of Interior Permanent Magnet Synchronous Motor using an Extended Kalman Filter”. In: *Journal of Electrical Engineering & Technology* 9 (2014), pp. 600–608.

[23] Martin A. Skoglund, G. Hendeby, and D. Axehill. “Extended Kalman filter modifications based on an optimization view point”. In: *2015 18th International Conference on Information Fusion (Fusion)* (2015), pp. 1856–1861.

[24] Ratanak So. “Extended Kalman Filter Simulink Model for Nonlinear System Modeling”. In: 2015.

[25] A. Vahidi, A. Stefanopoulou, and H. Peng. “Recursive least squares with forgetting for online estimation of vehicle mass and road grade: theory and experiments”. In: *Vehicle System Dynamics* 43 (2005), pp. 31–55.

[26] Tuan Pham Van et al. “Online Rotor and Stator Resistance Estimation Based on Artificial Neural Network Applied in Sensorless Induction Motor Drive”. In: *Energies* 13 (2020), p. 4946.

[27] I. Veselý, L. Vesely, and Z. Bradác. “MRAS identification of permanent magnet synchronous motor parameters”. In: *IFAC-PapersOnLine* 51 (2018), pp. 250–255.

[28] Isak Westin. “Sensorless Control of a PMSM”. In: 2016.

[29] B. Wittenmark, K. Årzén, and K. Åström. “Computer Control: An Overview”. In: 2002.

[30] Zakipour et al. “Simultaneous Application of EKF and RLS Methods for Induction Motor Speed and Parameters Estimation”. In: 1385. URL: `https://civilica.com/doc/19653`.

[31] Zedong Zheng, Yongdong Li, and M. Fadel. “Sensorless control of PMSM based on extended kalman filter”. In: *2007 European Conference on Power Electronics and Applications* (2007), pp. 1–8.

[32] Minglei Zhou, Jiang Long, and Chenchen Wang. “Real-Time Multiparameter Identification of a Salient-Pole PMSM Based on Two Steady States”. In: *Energies* 13 (2020), p. 6109.