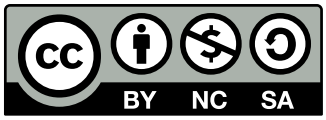
Aalto University
School of Electrical
Engineering

Master's programme in Automation and Electrical Engineering

Loss Estimator for Electric Motors Driven With Model Predictive Pulse Pattern Control

Johannes Lauriala

Master's Thesis
2024



Aalto University
School of Electrical
Engineering

Author Johannes Lauriala

Title Loss Estimator for Electric Motors Driven With Model Predictive Pulse Pattern Control

Degree programme Automation and Electrical Engineering

Major Electrical Power and Energy Engineering

Supervisor Prof. Marko Hinkkanen

Advisor DSc. Victor Mukherjee

Collaborative partner ABB Oy Large Motors and Generators

Date 31.12.2024 **Number of pages** 68 **Language** English

Abstract

Harmonic content of the supply voltage is an important design consideration for medium-voltage (MV) permanent magnet (PM) motors, because it can lead to increased losses in the motor, potentially causing issues with heating. Use of frequency converters for motor control commonly generates harmonics in the motor's supply voltage. Calculating harmonic losses during the design stage of a PM motor is often time-consuming and complicated. This thesis investigated an alternative approach for calculating harmonic losses for a MV PM motor supplied with a frequency converter utilising model predictive pulse pattern control (MP3C) as a control algorithm. The objective was to develop a fast and streamlined method for MP3C loss estimation. The thesis adopted a statistical machine learning (ML)-based approach for MP3C loss estimation. During the thesis, an estimator program was developed that utilises ML models trained using pre-calculated loss data. The estimator program was implemented using Python and LightGBM ML framework. The program generates estimates with a set of multiple ML models in conjunction with an interpolation algorithm. Training data for the ML models was calculated with finite element method (FEM) using a simulated MP3C voltage waveform. The general feasibility of a ML-based loss estimation program was demonstrated in the thesis. The implemented program serves as a functional prototype for further development of this concept. Accuracy of the estimations was not evaluated due to limited availability of the training data. The process for calculating the training data was identified as the key area where further development is needed, as the process requires a considerable amount of manual work. Continuing the automation efforts already undertaken in the thesis will likely increase the feasibility of the demonstrated approach.

Keywords MP3C, harmonic losses, machine learning, permanent magnet motor , lightgbm , python , finite element method

Aalto-yliopisto
Sähkötekniikan
korkeakoulu

Tekijä Johannes Lauriala

Työn nimi Häviöiden arviointi mallipredikti ivisellä pulssijonosäädöllä käytetyille sähkömoottoreille

Koulutusohjelma Automation and Electrical Engineering

Pääaine Electrical Power and Energy Engineering

Työn valvoja Prof. Marko Hinkkanen

Työn ohjaaja TkT. Victor Mukherjee

Yhteistyötaho ABB Oy Large Motors and Generators

Päivämäärä 31.12.2024 **Sivumäärä** 68 **Kieli** englanti

Tiivistelmä
Sähkömoottorin syöttöjännitteen sisältämä yliaaltosisältö aiheuttaa moottorissa harmonisia häviöitä. Häviöt aiheuttavat moottorissa lämpenemistä, joka saattaa puolestaan johtaa ongelmiin suorituskyvyssä ja toimintavarmuudessa. Taajuusmuuttajan käyttö moottorin ohjaamiseen on yksi yleisistä yliaaltosisällön lähteistä. Harmonisten häviöiden laskeminen on usein oleellinen tieto moottoria suunniteltaessa. Tyypillisesti se on kuitenkin hidasta ja monimutkaista. Tässä diplomityössä tutkittiin vaihtoehtoista menetelmää harmonisten häviöiden arviointiin kestomagneettimoottorille, jota syötetään mallipredikti ivisellä pulssijonosäädöllä ohjatulla taajuusmuuttajalla. Tavoitteena oli kehittää nopea ja helppokäyttöinen menetelmä häviöiden arviointiin. Työssä kehitettiin ohjelma häviöiden arvioimiseen. Ohjelman toiminta perustuu koneoppimismalleihin, jotka opetetaan ennalta lasketulla häviöaineistolla. Arvio häviöistä halutussa toimintapisteessä muodostetaan usean koneoppimismallin arvioiden pohjalta interpoloimalla. Opetusaineisto muodostettiin laskemalla harmonisia häviöitä elementtimenetelmällä. Laskuissa hyödynnettiin simuloimalla tuotettua mallipredikti ivisen pulssijonosäädön jänniteaaltomuotoa. Työ toteutettiin Python-ohjelmointikielellä hyödyntäen LightGBM-koneoppimiskirjastoa. Työn tuloksena voidaan todeta, että koneoppimismalleihin perustuva häviöiden arvionti vaikuttaa menetelmänä käyttökelpoiselta. Työssä tuotettiin toimiva prototyyppi häviöiden arvointiohjelmasta, jonka tarkkuutta ei kuitenkaan päästy rajallisesta opetusaineistosta johtuen mittaamaan. Työssä käytetty opetusaineiston keräysmenetelmä osoittautui aikaavieväksi. Joitakin osia aineiston keräysprosessista automatisoitiin työn aikana kehitetyillä apuohjelmilla, mutta aineiston kerääminen vaatii edelleen paljon manuaalista työtä. Aineiston keräysprosessin täysi automatisointi parantaisi työssä esitellyn menetelmän sovellettavuutta.

Avainsanat harmoniset häviöt, koneoppiminen, kestomagneettimoottori, mallipredikti ivinen pulssijonosäätö , taajuusmuuttaja , elementtimenetelmä

Preface

I want to thank Prof. Marko Hinkkanen from Aalto University and Dr. Victor Mukherjee from ABB for providing excellent supervision and guidance over the course of this thesis. Additionally, I want to thank Mr. Juha-Pekka Kivioja, Mr. Aleksi Koskinen and Mr. Wille Halmesmäki from ABB for providing me with an interesting thesis topic, as well as invaluable support and insight into the topic without which the thesis wouldn't have been possible. Last but certainly not the least, I want to thank my family for the support I have received throughout the process of writing the thesis.

Otaniemi, 31 December 2024

Johannes Lauriala

Contents

Symbols, operators and abbreviations

Symbols

A	area
$\mathbf{B}$	magnetic flux density
$\mathbf{E}$	electric field strength
$\mathcal{F}$	magnetomotive force
f	frequency
$\mathbf{H}$	magnetic field strength
i	current
$\mathcal{L}$	inductance
l	length
P	power
R	resistance
$\mathcal{R}$	reluctance
T	temperature
t	time
v	voltage
μ	permeability
ρ	resistivity
τ	torque
Φ	magnetic flux
Ψ	flux linkage
ω	angular frequency

Operators

$\frac{\mathrm{d}}{\mathrm{d}x}$	derivative with respect to variable x
$\oint_p \mathbf{F} \cdot dx$	closed-loop integral of vector function $\mathbf{F}$ along path p with respect to variable x
$\int_A \mathbf{F} \cdot d\mathbf{S}$	surface integral of vector function $\mathbf{F}$ with respect to surface $\mathbf{S}$

Abbreviations

AC	alternating current
CB-PWM	carrier-based pulse width modulation
DC	direct current
DTC	direct torque control
DOL	direct on line
FEM	finite element method
ML	machine learning
MPC	model predictive control
MV	medium voltage
MP3C	model predictive pulse pattern control
OPP	optimised pulse pattern
SVM	space vector modulation
THD	total harmonic distortion
TDD	total demand distortion
PM	permanent magnet
PMSM	permanent magnet synchronous motor
VSD	variable speed drive
VSI	voltage source inverter

1 Introduction

1.1 Background and motivation

Electrical motors are ubiquitous in the modern world. They are essential in applications that vary from domestic appliances to electric propulsion of large container ships. Because the applications of electrical motors cover an extremely wide envelope of operational regimes in terms of required power, speed and torque, an equally diverse range of electrical motors exist to serve these regimes. Regardless of motor type or use case, efficiency of the motor is typically considered important. Not only does it have a direct effect on the cost of running a motor, low efficiency also leads to excess heat being generated in the motor. Heating will adversely affect the motor, possibly decreasing efficiency further and decreasing the motor's lifespan. Thus, optimising efficiency has numerous benefits and is typically an important consideration during the design process.

For industrial applications large electric motors of medium voltage (MV) are often utilised. MV as used in this thesis refers to voltages over 1kV and under 15kV. The power output of these motors may reach to several megawatts. With increasing power the importance of maximising efficiency also greatly increases. For large motors, the capital cost of buying the motor is often negligible when compared to its running electricity costs. According to ABB (2023), the purchase price for 110kW 1500rpm motor forms only 2% of its total cost of ownership, with 97% being the electricity cost and 1% being maintenance. Therefore, it is clear that it is often beneficial to emphasise efficiency at the design stage of a motor.

Permanent magnet (PM) alternating current (AC) motors are an increasingly popular choice for large industrial MV motors. Asynchronous motors, an another popular motor type, are generally cheaper to purchase. However, comparable PM motors are typically more efficient. This is often preferable due to aforementioned economical reasons.

In industrial applications where MV AC motors are used, a variable speed range is often required. Because AC motors lack an inherent way to control speed, numerous approaches exist to accommodate this need. While it is possible to achieve speed control through mechanical means such as gear boxes, this approach greatly reduces efficiency and adds mechanical complexity. Thus, electrical motor control is often preferred. This is typically achieved with frequency converters. Frequency converters work by supplying the motor with a voltage of variable frequency and magnitude. Because the speed of an AC motor is dependent on the supply voltage's frequency, this enables the motor's speed to be varied at will. A combination of a frequency converter and a motor can be referred to as a variable speed drive (VSD).

Voltage source inverters (VSIs) are a common type of converter topology for large drives. A VSI converter operates by first rectifying the AC voltage provided by the electricity grid and storing resulting DC voltage in capacitors in the intermediate circuit. Then, the DC is converted into variable-frequency AC by using switching

devices such as thyristors or insulated gate bipolar transistors (IGBTs). The switching devices are used to generate a voltage resembling a sine wave of desired frequency, which is then fed to the motor's terminals. The resulting synthesised voltage is never truly sinusoidal, because the switching devices do not operate in the linear region. Instead, they are always either fully closed or fully open.

As Sen (2013) notes, the output voltage of a converter typically has a high harmonic content above the fundamental frequency, especially when the number of pulses per half-cycle is low. This is typically true for MV converters (ABB, 2021). This harmonic content of the supply voltage of a motor can present challenges for the driven motor. Namely, harmonic content present in the supply voltage of a motor can increase the losses induced in the motor (Pyrhönen et al., 2014). Increased losses can lead to temperature rise, which in turn can decrease the lifetime of the motor. As an example, an increase in temperature of 10 K can halve the lifetime of the motor insulation (Pyrhönen et al., 2014).

While the output voltage of a VSI can be filtered to reduce harmonics, filtering of industrial MV drive systems is challenging due to power and voltage levels involved. Thus, the output of VSI MV drives is often unfiltered. Another way to reduce the harmonic losses of the motor is changing the switching parameters of the VSI. As noted by Geyer (2017), a fundamental tradeoff exists between harmonic content of VSI output voltage and the switching frequency: increased switching frequency reduces the harmonic content of motor supply voltage, at the expense of increasing switching losses of the converter. Model predictive pulse pattern control (MP3C) is a drive control scheme that aims to mitigate this tradeoff by combining two techniques, model predictive control (MPC) and optimised pulse patterns (OPP) (Geyer et al., 2012). MP3C is especially useful at medium voltages because of the low switching speeds involved.

This thesis examines a MV drive system consisting of a PM motor driven by a three-level VSI converter. The VSI uses integrated gate-commutated thyristors (IGCT) as the switching devices. IGBTs exhibit large switching losses at high frequencies, which results in a typical switching frequency of IGBT-equipped converters to be about 500Hz (Hitachi Energy, 2024). To mitigate the harmonics caused by the low switching speed, the converter utilises MP3C.

While MP3C aims to optimise the switching frequency to minimise the total losses in the drive system, additional harmonic losses will inevitably be generated in the motor when compared to purely sinusoidal supply. During the design activities of a motor included in the drive system, it is important to be able to estimate the magnitude of additional losses generated by the MP3C supply. In addition to losses at nominal speed and power, estimates of losses at several partial speed an power points are often needed.

1.2 Objective

One commonly used way to calculate harmonic losses caused by a non-sinusoidal voltage is utilising finite element method (FEM) electrical calculations. In FEM, the magnetic fields inside the motor are evaluated based on a model of magnetic properties of the motor. This method is accurate but requires several specialised programs and a significant amount of manual work. In addition, the time required to run the calculations using the current method is non-trivial, typical calculation time for one motor at one operating point being in the order of 20-30 minutes.

These factors present significant barriers for utilising loss estimation in activities where it might otherwise be beneficial. As an example, during the offer phase, suitable motor design is determined based on customer requirements and other constraints such as adherence to standards. During this process, additional harmonic losses caused by converter supply can also be important factors. However, the current method for calculating MP3C losses is not directly accessible during the offer phase. Instead, cooperation is required with the specialists who have the necessary software and expertise to perform the calculation.

Additionally, the amount of time required to perform the calculations using the current method makes it infeasible for many activities. During the offer and design phases, a need might exist to estimate the losses for a large number of pre-existing designs at once in order to select the best design as a starting point for the new design. Similarly, a design engineer might want to run multiple calculations to see how a certain parameter affects the losses.

Therefore, the objective for this thesis is to develop a streamlined method for estimating the additional losses brought on by the MP3C control method for medium-voltage (>1 kV) permanent magnet motors. The thesis seeks to develop a loss estimation method that can be used for purposes where automated, fast loss estimation is required. More specifically, the requirements for the new method are:

- speed (estimation should take seconds instead of 20-30 minutes)
- ease of use (estimation should not require the use and knowledge of several separate programs or extensive manual configuration work)
- accuracy (estimation should be as close to the one provided by the current method as possible)

1.3 Structure

This thesis consists of five chapters. The first chapter serves as an introduction to the topic, problem, and approach of the thesis. In the second chapter, relevant theory of permanent magnet motors and model predictive control are presented. Then, in the third chapter, the methodology, development process and structure of a novel loss estimator program is described. In the fourth chapter, the implemented loss estimator program is given an overview in terms of capabilities and user interface, and

its performance is discussed. To conclude, in the fifth and final chapter, the results of the thesis are summarised and prospects for future development based on them are briefly discussed.

2 Theory

This chapter introduces the theory that is relevant to the topic of the thesis. The chapter begins with an overview of permanent magnet synchronous motors (PMSMs). Then, an overview is given on variable speed drives (VSDs), and the converter type examined in the thesis, voltage source inverter (VSI), is introduced. Next, losses in an electromechanical system are discussed with a special emphasis on harmonic losses. Following this, the per-unit system is briefly introduced and defined, followed by discussion about the various control methods used in converters. Finally, model predictive pulse pattern control is introduced.

2.1 Permanent magnet synchronous motors

2.1.1 Overview

PMSMs are electrical motors that employ permanent magnets for creating the rotor field. As implied by the name, they operate synchronously. Synchronous operation means that during the normal operation, the rotor of the machine rotates at a synchronous speed that is determined by the supply frequency and the number of pole pairs of the motor in a following manner (Hendershot Jr and Miller, 2010):

$$n_s = \frac{f \cdot 60}{2P} \tag{1}$$

where f is the fundamental frequency of the supply voltage and $2P$ is the number of pole pairs in the motor.

A PMSM can produce more transient and stable torque and is more efficient than an induction motor of the same size Slemon (1992). Similarly, compared to a separately excited synchronous motors, which feature electrically excited magnets in the rotor, PMSMs are less complex and compact for the comparable power.

Figure 1 depicts a simplified cross section of a PMSM. The stator features a slotted core with windings of the motor's three phases routed through the slots. Depending on the number of the pole pairs, the stator features one or more windings per phase. Each slot in the stator houses one half of a winding of one phase running in opposite directions. The permanent magnets are attached to the rotor. Depending on the type of PM motor, they may be attached on the surface or embedded in the rotor.

As depicted in Figure 1, the three phases a, b and c each have their own axes oriented along the field produced by them. The rotor has two axes commonly defined for the purposes of analysis: the direct axis (d) and the quadrature axis (q). The direct axis aligned with the direction of the rotor's field, whereas the quadrature axis is oriented orthogonally to the direct axis. The angle β in Figure 1 denotes the angle between the stator's a-axis and the rotor's d-axis.

It should be noted that the cross section presented in Figure 1 is a simplification, intended to only demonstrate the principal parts of a three-phase permanent magnet

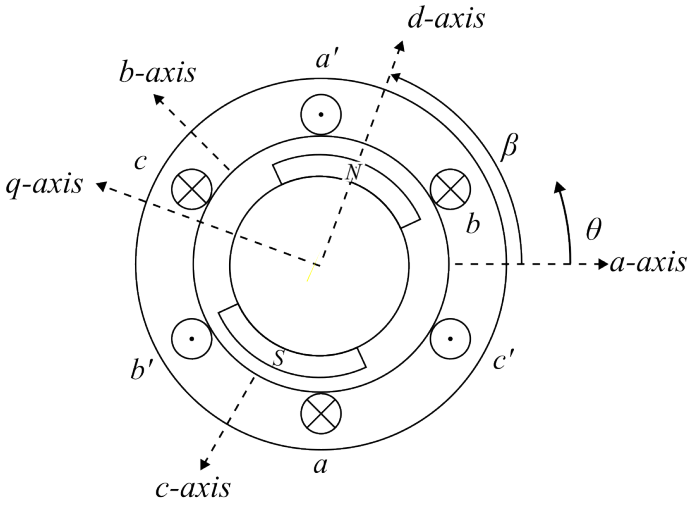


Figure 1: Simplified cross section of a 2-pole cylindrical PM motor with its axes a,b,c,d and q labeled. $\cdot$ and $\times$ symbols denote the opposite winding directions inside the slots, with $\times$ signifying direction away from the observer. The three phases are denoted as a, b and c.

motor. The picture is thus not representative of an actual typical PM cross section and omits features such as the winding distribution inside the slots and realistic geometry. Where applicable, these features will be explained below.

2.1.2 Three-phase supply

Alternating current (AC) electric motors utilise sinusoidal alternating voltage for generating torque. Large industrial motors, such as the type examined in this thesis, utilise an AC supply with three phases. Three-phase AC supply consists of three sinusoidal voltages with each phase having a 120 degree phase difference with respect to the other two. In an ideal system, where the amplitudes of the three voltages are identical, this can be characterised as phase voltage equations followingly:

$$v_a = \sqrt{2}V_P \sin \omega t \tag{2}$$

$$v_b = \sqrt{2}V_P \sin (\omega t - 120^\circ) \tag{3}$$

$$v_c = \sqrt{2}V_P \sin (\omega t + 120^\circ) \tag{4}$$

where v_a, v_b, and v_c are the phase voltages, V_P is the root mean square (RMS) value of the supply voltage, ω is the angular frequency of the supply voltage and t is the time.

When the three voltages v_a, v_b and v_c are fed to the windings of a three-phase motor, such as one depicted in Figure 1, the fields generated by the three winding will generate a rotating net field in the airgap of the motor. The field will rotate at the synchronous speed, defined in (1).

2.1.3 Magnetic circuit

PMSMs generate torque via electromagnetic interaction between the rotor and the stator. Flow of magnetic flux inside the motor can be modelled largely analogously to the flow of electrons in an electrical circuit with following quantities being comparable:

- Electromotive force E and magnetomotive force $\mathcal{F}$
- Electrical resistance R and magnetic reluctance $\mathcal{R}$
- Electrical current $I(=\frac{E}{R})$ and magnetic flux $\Phi(=\frac{\mathcal{F}}{\Phi})$

One exception to this analogous behaviour, as noted by Sen (2013), is reluctance. In magnetic circuits where ferromagnetic materials are used, the magnetisation characteristic will be non-linear, which means that the relation between reluctance $\mathcal{R}$ and resultant flux Φ is non-linear in these cases, which would not be the case in an equivalent electrical circuit.

The magnetic circuit of a rotating electrical machine consists of three principal elements: the stator, the air gap, and the rotor. The magnetic cores of stator and rotor consist of ferromagnetic materials, typically steel. The core material will have a high relative permeability, whereas the air in the air gap will have relative permeability of ≈ 1. With the reluctance of the material being inversely proportional to the relative permeability, this means that the air gap will typically present the point of highest reluctance in the magnetic circuit of PMSM.

Flux linkage is one of the fundamental quantities used to describe electromagnetic circuits. It is defined by Sen (2013) as follows:

$$\Psi = N\Phi \tag{5}$$

where Ψ denotes the flux linkage, N denotes the number of turns in the coil and Φ denotes the flux through the coil.

The inductance of a coil can further be defined by its flux linkage and current:

$$\mathcal{L} = \frac{\Psi}{i} \tag{6}$$

where $\mathcal{L}$ denotes the coil inductance and i denotes the current in the coil.

2.1.4 Equivalent circuit

The electrical characteristics of a motor can be described with an equivalent circuit presenting the motor as an electrical circuit. As Slemon (1992) describes, the torque is generated (in the case of motor) or converted (in the case of generator) by the magnetomotive force (mmf) that is the combination of the stator's and rotor's mmf. In a PM machine, the stator's mmf is generated by the stator windings, and the rotor's

mmf originates from the permanent magnets. An electrical equivalent circuit of a PMSM is shown in Figure 2.

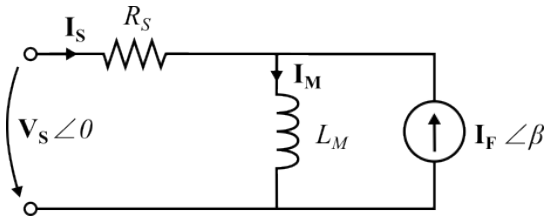


Figure 2: Electrical equivalent circuit of a syncrhronous motor.

2.2 Variable speed drives

Variable speed drives (VSDs), as they are referred to in this thesis, are systems consisting of a converter and an electric motor driving a mechanical load. The advantage of this configuration, as opposed to directly supplying the motor from the grid, is that the converter enables the speed control of the motor, which is an essential feature in many applications.

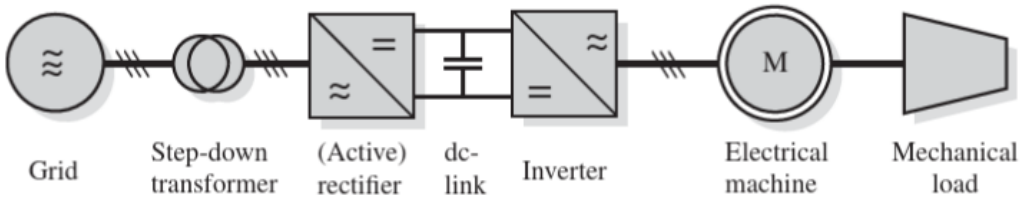


Figure 3: DC link VSD (Geyer, 2017, p. 4)

The converter can be further divided into three sections: the rectifier, the direct current (DC) link and the inverter. Converter achieves speed control by first rectifying the incoming AC supply and using the resulting DC voltage to charge capacitors in the DC-link. This rectified voltage is then utilised by the inverter section to produce a motor supply voltage of desired frequency. The inverter section synthesises the motor voltage waveform by opening and closing semiconductor switches connecting the DC-link voltage to the motor in sequence. The parts of a VSD with a DC-link converter are shown in Figure 3.

It should be noted that the DC-link converter topology described above is not the only way to achieve the speed control of an electrical motor and the term "variable speed drive" may be used to refer to various possible combinations of an electrical machine and a speed-controlling equipment depending on context. However, DC-link converters are both widely utilised in medium-voltage applications and the type of converter this thesis examines. Therefore, any mention of VSDs in this thesis explicitly refers to this type of a system.

2.3 Overview of losses

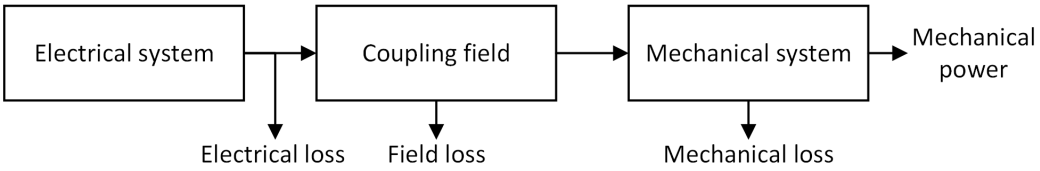


Figure 4: Electromechanical energy conversion.

Figure 4 describes the general process of electromechanical conversion from the electrical domain into the mechanical, as outlined by Sen (2013). As is shown, the process of converting between the electrical and mechanical domains involves inherent, classifiable losses. Thus, the output power (mechanical or electrical) will always be less than the input power (electrical or mechanical), and the difference will manifest as losses. These losses will be dissipated as heat in the system.

Heat management is an important part of motor design. As noted by Pyrhönen et al. (2014), the temperature rise of a motor in use is the final factor in determining the maximum continuous power it can be rated at. The running temperature of a motor has significant implications for the lifetime of its components, particularly insulation and bearings. For the insulation system, a temperature rise of 10 K can be expected to reduce lifetime by up to 50% (Pyrhönen et al., 2014).

In addition to the heat-related aging mechanisms described above, permanent magnet motors have an additional heat-related limitation. Because permanent magnets used in the motors undergo demagnetisation when they exceed their curie temperature, care must be taken to ensure the magnet temperature stays below the curie point of the magnet material. As Hendershot Jr and Miller (2010) note, monitoring magnet temperatures is challenging, and even a partial demagnetisation will significantly change the operating characteristics of a motor For these reasons, care must be taken in the design of a motor to make such events as unlikely as possible.

As noted by Pyrhönen et al. (2014), planning the heat management of an electrical motor is more difficult and complicated than its electromagnetic design and dimensioning. Accurate assessment of motor's heating necessitates modelling the heat flow inside the motor at the design stage. This process is already challenging for DOL motors, and additional losses precipitated by non-sinusoidal supply voltage of a VSD present an additional complicating factor.

2.3.1 Resistive losses

Resistive losses are a result of the winding resistance. They are a function of the resistance and current followingly:

$$P_{Cu} = i^2 R \tag{7}$$

where P_{Cu} is the resistive loss of a conductor, i is the current of the conductor and R is the DC resistance of the conductor. Conductor resistance is a function of conductor material resistivity, length and cross-sectional area in a following manner:

$$R = \rho \frac{l}{A} \tag{8}$$

where ρ is the resistivity of conductor material, l is the length of the conductor and A is the cross-sectional area of the conductor. The resistivity of copper, as listed by Kazimierczuk (2014) is 1.724×10^{-8} Ωm at temperature of 20 °C. However, resistivity changes as a function of temperature. The change is approximately linear, which means it can be characterised with a material-specific temperature coefficient. Kazimierczuk (2014) lists the temperature coefficient α for copper as 0.00393 $\frac{1}{^\circ C}$. The change of resistivity ρ_T over temperature change T is:

$$\rho_T = \rho_{T_0}[1 + \alpha(T - T_0)] \tag{9}$$

where T_0 is the initial temperature.

Equation (8) shows that resistive losses depend on resistance and current. Thus, content present in the supply voltage does not directly affect the resistive losses, given that current and resistance stay constant. However, as shown above in equation (9), temperature does affect the resistivity of the copper windings. Non-sinusoidal supply voltage can still affect resistive losses if it precipitates losses through other mechanisms that affect the winding temperatures.

Additionally, Equation (7) only accounts for resistive losses in cases where the current density is constant in the conductor. In reality, this may not always be the case. Skin effect and proximity effect discussed below can cause varying current densities in the windings of a PMSM and thus must be taken into account when discussing resistive losses.

2.3.2 Skin effect and proximity effect

Skin effect and proximity effect are frequency-dependent phenomena that can affect the current distribution in conductors such as the windings of a PMSM. An ideal conductor of uniform resistance has an uniform current density along its cross-sectional area. However, if the conductor is subjected to magnetic fields, they can affect the distribution of currents in the windings.

Skin effect is a phenomenon where the current density is concentrated on the outer layers of a conductor. When a conductor carries AC, eddy currents are generated in the conductor due to changing magnetic field generated by the AC. As per Lenz's law, the eddy currents induced by the magnetic field act to cancel out the main current near the center of the conductor and add to the current at the edges, leading to net

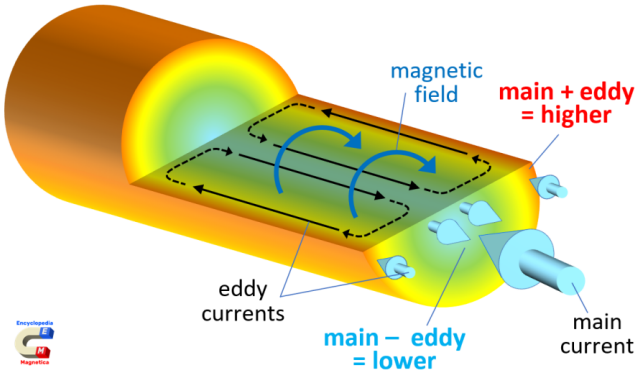


Figure 5: Illustration of skin effect. (Zurek, 2023*a*).

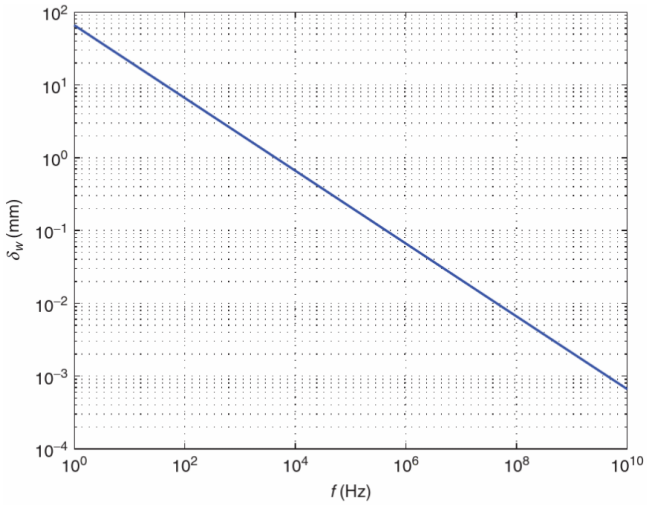


Figure 6: Skin depth for copper δ_{Cu} as a function of frequency (Kazimierczuk, 2014, p.169).

current being concentrated in the surface of the conductor. This is illustrated in Figure 5. Because generating eddy currents requires AC, skin effect does not occur at DC.

The depth of the current carrying layer at the surface of a conductor exhibiting skin effect is called the skin depth. Kazimierczuk (2014) defines the skin depth as the depth at which the electromagnetic field is reduced to $1/e$ of its original value. The skin effect is primarily a function frequency, as shown in Figure 6. The skin depth is also affected by the temperature of the conductor. The relation between the temperature and the skin depth is shown in Figure 7. As can be observed, temperature does not have considerable effect on skin depth across the typical range of operating temperatures for PMSM windings (20-80 °C). As Kazimierczuk (2014) notes, the skin effect can only be neglected in cases where the skin depth is considerably greater than the diameter of the conductor.

Proximity effect is also caused by induced eddy currents. However, whereas the eddy currents that generate the skin effect are the result of conductor's internal magnetic

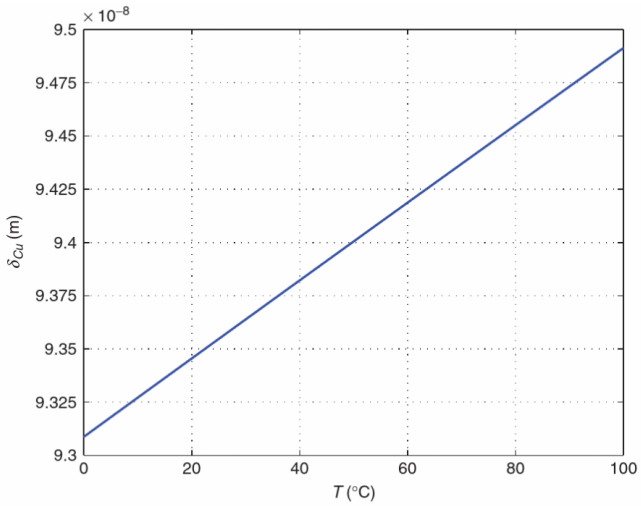


Figure 7: Skin depth for copper δ_{Cu} as a function of temperature (Kazimierczuk, 2014, p.168).

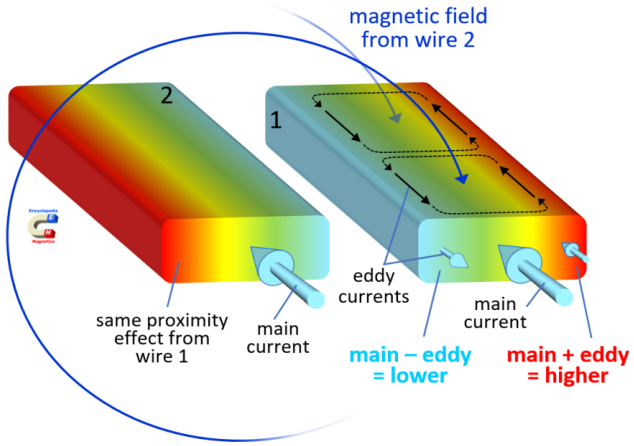


Figure 8: Illustration of proximity effect. (Zurek, 2023*b*).

field, proximity effect is caused by a field that is external to the cable. In a typical example, such as shown in Figure 8, the source of the external field is another parallel conductor. Similarly to skin effect, proximity effect only occurs with AC. The mechanism of action is also similar to skin effect. As shown in Figure 8, an adjacent conductor carrying AC generates a magnetic field, which induces eddy currents. These eddy currents then act to amplify the main current at one side of the conductor, and decrease it one the other side.

Because skin effect and proximity effect can cause increased current densities, they can also lead to increased resistive losses in areas where the current is concentrated. This, consequently, can increase the temperature of the winding, which, in turn, leads to increased resistive losses due to decreased conductivity of the winding. Therefore, it is important to consider the effects they have on additional losses caused by VSD control.

2.3.3 Hysteresis losses

When an external magnetic field intensity **H** is applied to a magnetic medium, such as the iron core of an electric machine, flux density **B** is generated in the medium. This process is called the magnetisation of the medium. If the relationship between **H** and **B** is linear, it can be defined as:

$$\mathbf{B} = \mu\mathbf{H} \tag{10}$$

where μ is the permeability of the medium, which is material-dependent.

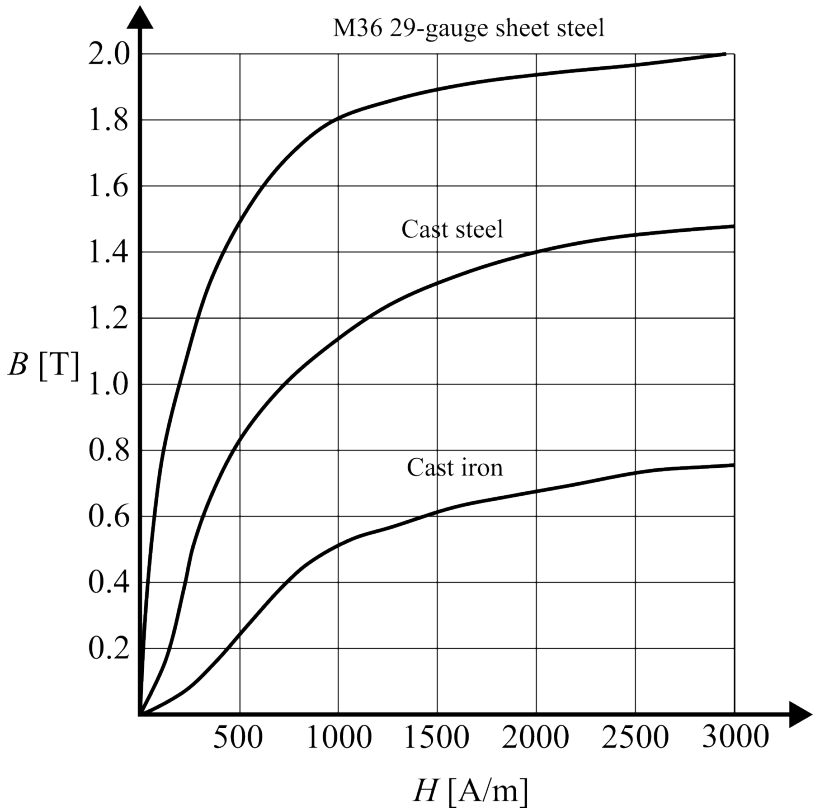


Figure 9: Magnetisation curves for cast iron, cast steel and M36 29-gauge steel sheet. (Slemon, 1992, p.17)

As noted by Slemon (1992), the behaviour depicted in (10) can be used to describe the $B - H$ relationship of most materials with an error of less than 0.001%. The previous is not true, however, for ferromagnetic materials, such as iron, nickel and cobalt. For ferromagnetic materials, the $B - H$ relationship is markedly non-linear. Magnetisation curves, as shown in Figure 9, are used to describe how the magnetic flux density B responds to incident magnetic field H.

This non-linearity is due to the structure of ferromagnetic materials. As Slemon (1992) describes, ferromagnetic materials are able to generate a magnetic field because of their atomic and molecular structure that differs from non-ferromagnetic materials.

Fundamentally, a magnetic field generated due to ferromagnetism is the product of individual electrons' magnetic moments. Ferromagnetic elements have their electron orbitals located asymmetrically such that the atom generates an external net magnetic moment. Additionally, ferromagnetic materials have a structure that prevents adjacent atom's magnetic moments from cancelling each other out, resulting in material that is able to generate a net magnetic field.

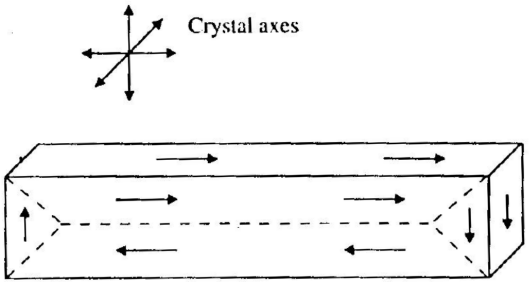


Figure 10: Magnetic domains inside a crystal of iron. (Slemon, 1992, p. 12)

However, ferromagnetic materials do not necessarily generate a strong external magnetic field. The typical structure of a ferromagnetic material consists of magnetically aligned areas, called magnetic domains. As Slemon (1992) explains, ferromagnetic materials often have a structure consisting of crystals, as depicted in Figure 10. Each crystal can contain multiple magnetic domains. The magnetic domains inside a crystal can have different orientations, but it is typical for them to find orientations at the time of material formation that require the least amount of energy, leading to domain orientations where the crystal's external magnetic field is minimised, as shown in Figure 10.

Magnetisation of a ferromagnetic material involves utilising an external magnetic field to align the magnetic domains contained in the material. As a result of magnetisation, an object consisting of electromagnetic material will have its magnetic domains aligned, and as a result, the object is able to generate an external magnetic field.

Magnetic materials can be divided by their magnetisation characteristics into soft and hard magnetic materials (Slemon, 1992). Magnetically soft materials have an internal structure that makes it easy for the magnetic domains to re-align. This makes them susceptible for being magnetised by an external field. Conversely, the magnetic domains of magnetically hard materials, also known as permanent magnet materials, require significant flux density to align, which leads to their ability to retain magnetisation.

Hysteresis is a phenomenon that occurs in ferromagnetic materials as a result of magnetisation. As a result of magnetic domain alignment, (10) does not hold for ferromagnetic materials. Instead, when the incident magnetic field H is removed, some amount of remanent flux B_r remains in the medium. Similarly, if subjected to the magnetic field in an opposing direction, $-H$, magnetic flux of $-B_r$ remains when the magnetic field is removed.

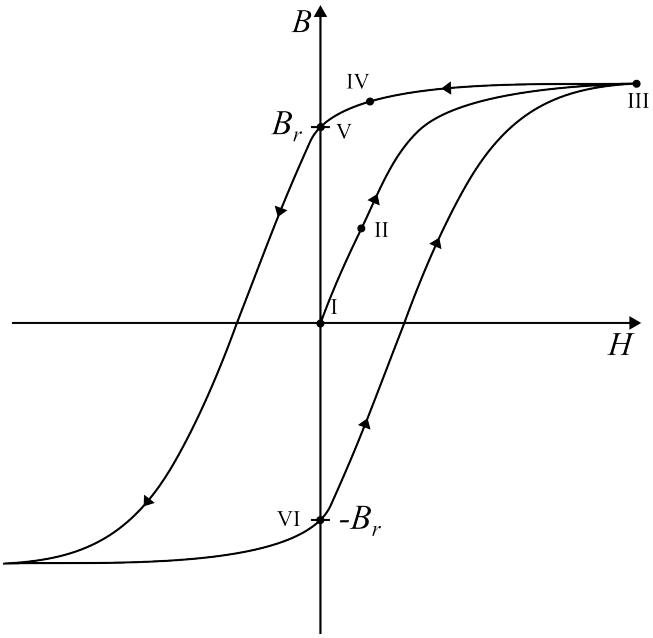


Figure 11: An example of a hysteresis loop.

The hysteresis loop, pictured in Figure 11, describes the hysteresis behaviour of a material in a changing magnetic field. When the field is first applied, both H and B are zero (I). When H increases, B also does so in a near-linear manner (II) until the material approaches saturation (III). When B in medium is not able to increase despite of incident H increasing, it is said to be saturated. When H starts to decreasing, B will also decrease, but the slope of decrease is significantly gentler than the initial increase (IV). As a result, when H reaches zero (V), residual flux B_r still exists within the medium.

When the magnetic field switches direction, the behaviour is similar. After the medium has been saturated in the reverse direction, and H again falls to zero, flux of $-B_r$ remains (VI). When H again turns positive, B stars to increase from $-B_r$. Finally, when the core again saturates in the positive direction, the hysteresis loop closes at (III).

As Slemon (1992) mentions, the size of the hysteresis loop is dependent on amplitude of H. A family of hysteresis loops for different value of H are shown in Figure 12. The shape of the hysteresis loop depends on the magnetisation characteristics of a material. As also seen in Figure 12, the tips of different hysteresis loops in the same family fall along the magnetisation curve of the material in question.

Hysteresis losses occur as a result of magnetisation cycles in a ferromagnetic material. As previously explained, the process of magnetisation involves aligning the magnetic domains with an external field. This process requires energy and inevitably incurs some losses. When the material is subjected to changing magnetic field, the losses caused by the domain alignment will manifest as hysteresis losses.

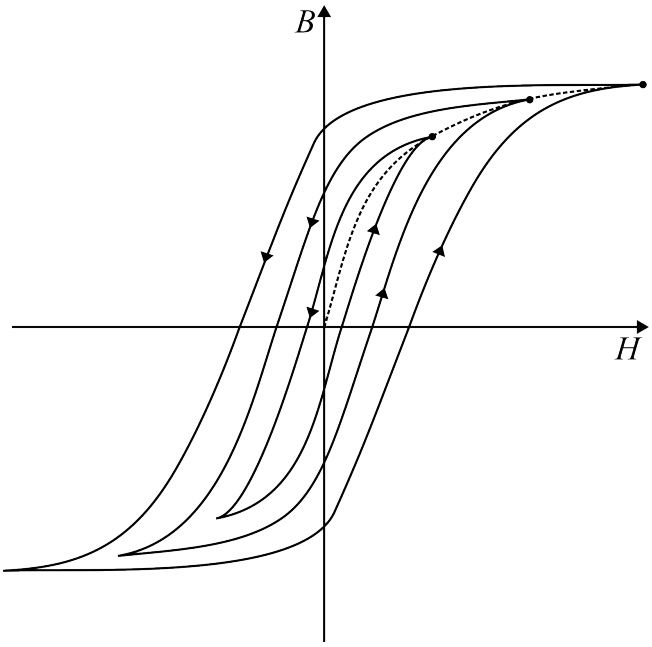


Figure 12: Family of hysteresis loops with different amplitudes of H. The resultant magnetisation curve is plotted with a dashed line.

As explained by Slemon (1992), hysteresis losses can be understood analytically by examining the process of traversing the hysteresis losses. The energy input from a changing flux per unit volume is

$$\Delta w = \int_{B_a}^{B_b} H \cdot dB \tag{11}$$

where w is energy, B_a is the flux at the beginning, B_b is the flux at the end and H is the incident magnetic field.

When traversing the hysteresis loop pictured in Figure 11, and starting from a, the energy input from a to c as per (11) is the area enclosed by $abcda$. Conversely, when H decreases from its peak value at c to zero at d, energy is released. The amount of released energy is the area enclosed by $cdec$. The same principle applies when traversing the loop from zero H to the negative peak value of H. Now, the area enclosed, and thus the amount of energy input, is the enclosed area $efghe$. Again, when H returns to zero, some amount of energy, enclosed by $ghag$, is released.

As Slemon (1992) points out, the amount of energy at the start of a hysteresis loop described in Figure must be the same as the energy at the end of the loop, assuming the loop is closed. This leads to the conclusion that the amount of energy per unit volume lost during one complete cycle of a hysteresis loop is, in the case of Figure 11:

$$\Delta w = abcda - cdec + efghe - ghag = abcefga \tag{12}$$

which is the area of the hysteresis loop.

2.3.4 Eddy current losses

Eddy current losses are caused by currents induced in the core material by the changing magnetic flux. Faraday's law of induction in an integral form can be used to describe the electromotive force caused by time-variant magnetic field.

$$\oint_{\partial \mathbf{S}} \mathbf{E} \cdot d\mathbf{l} = -\frac{\mathrm{d}}{\mathrm{d}t} \int_{\mathbf{S}} \mathbf{B} \cdot d\mathbf{A} \tag{13}$$

where $\mathbf{S}$ is surface with contour $\partial\mathbf{S}$, $\mathbf{E}$ is the electric field, $d\mathbf{l}$ is a vector element of surface contour $\partial\mathbf{S}$, $\mathbf{B}$ is the magnetic flux density and $d\mathbf{A}$ is a vector element of surface $\mathbf{S}$.

In the form presented in (13), the Faraday's law of induction states that when a time-variant magnetic flux $\mathbf{B}$ passes through surface $\mathbf{S}$, which is enclosed by contour $\delta\mathbf{S}$, and electric field $\mathbf{E}$ is induced along the contour. If the contour is conductive, a current will be induced along it. Subsequently, if the contour has resistance higher than zero, power proportional to current and resistance of the contour will be generated. If the changing magnetic flux passes through an uniform, conductive medium, circular currents known as eddy currents will be generated.

In an electric machine, the majority of eddy currents and associated losses are generated in the core, which experiences high flux densities. Additionally, in permanent magnet motors, the permanent magnets exhibit notable eddy current losses as well. Because the core is typically constructed of ferromagnetic materials, which are conductive, measures need to be taken to reduce the generation of eddy currents.

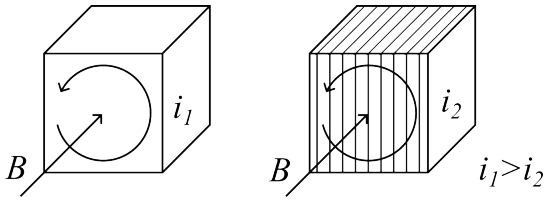


Figure 13: Eddy current formation in non-laminated and laminated core material. For equal magnetic flux B, resultant eddy current i_1 generated in non-laminated material is higher than eddy current i_2 generated in laminated material.

Sen (2013) lists two methods for reducing eddy current losses: increasing resistivity and using laminated cores. For increasing resistivity, adding silicon to the iron is mentioned. Addition of silicon increases the resistivity of iron significantly, decreasing eddy currents and losses. Laminated cores consist of thin sheets of core material that are electrically isolated from each other. The lamination is oriented oppose the induced eddy currents, as shown in Figure 13

2.3.5 Mechanical losses

Mechanical losses in an electric motor are caused by the relative rotational motion between the stator and the rotor. The rotation generates losses in different parts of the motor as a result of either friction or windage. Friction losses are generated as a result of two surfaces rubbing against each other, while windage losses are caused by the air movement generated by a moving object, such as the rotor of an electric motor.

Mechanical losses are largely dependent on the mechanical properties and speed of motor. Therefore, type of supply does not directly affect them. However, the type of supply can indirectly affect the mechanical losses. As an example, VSD supply makes it possible to operate at wide range of different operating points, which also has implications for the mechanical design of a motor. Because the focus of this thesis is on electromagnetic losses, further discussion of VSD control on mechanical losses is omitted.

2.4 Harmonic losses

Harmonic content in a signal consists of components with frequencies that are integral multiples of the signal's fundamental frequency f_N. In the context of PM motors, harmonics can be present, for example, in the stator current, stator voltage or torque. Harmonics in an electric motor can originate from many sources, some of which IEEE Working Group on Power System Harmonics (1983) lists as the tooth ripple, nonsinusoidal airgap flux and the flux changes caused by transient loads.

In addition to these, a frequency converter typically imposes harmonics on the motor's supply voltage. Because the frequency converter in a drive system produces its output voltage by switching, it contains, by nature, harmonics above the fundamental frequency. When a motor is supplied using a converter, the harmonic content present in the supply subsequently also induces stator current containing harmonics. These current harmonics can then precipitate additional losses in the motor. Because this thesis is primarily concerned with converter-borne additional losses, the following discussion is limited to the losses caused by converter-caused supply harmonics.

Harmonic content is typically quantified in terms of total distortion. Total distortion is calculated as a sum of harmonic compoments. This thesis uses total harmonic distortion (THD) and total demand distortion (TDD) as a measure of current distortion. Masoum (2015) defines current THD and TDD followingly:

$$THD_i = \frac{\sqrt{\sum_{h=2}^{\infty} (i^{(h)})^2}}{i^{(1)}} \tag{14}$$

$$TDD_i = \frac{\sqrt{\sum_{h=2}^{50} (i^{(h)})^2}}{i_N} \tag{15}$$

where THD_i is the current THD, TDD_i the current TDD, $i^{(1)}$ is the fundamental current component, i_N is the nominal stator rms current and $i^{(h)}$ is the amplitude of h:th current harmonic component. TDD is used in the thesis in conjunction with more common total harmonic distortion (THD) because it poses some advantages in the analysis of power systems. Namely, as Geyer (2017) explains, current TDD is defined in terms of nominal current instead of current dc component, which means that as current nears zero, TDD remains roughly constant while THD approaches infinity.

Harmonic losses due to current distortion occur in the windings and the core of a motor. In the windings, additional harmonic components cause losses due to the skin effect illustrated in Figure 5 and the proximity effect shown in Figure 8. As Masoum (2015) notes, harmonic losses P_h at harmonic frequency $h \cdot f_N$ require both harmonic voltage v_h and harmonic current i_h to be present. The total effect on losses due to harmonic skin effect and proximity effect losses is the sum of losses at different harmonic frequencies.

In addition to windings, harmonic losses can also occur in the core of a PMSM. Fluctuating magnetic field intensity caused by the harmonic frequency content of stator current results in harmonic hysteresis losses, which are often called iron losses. As explained in the previous subchapter, traversing the hysteresis loop involves an inherent loss. This is also true for the harmonic fluctuations. As explained by Masoum (2015), harmonic iron losses occur even under sinusoidal supply voltage, because the B-H relationship in the core material is nonlinear which, in turn, introduces current harmonics. Harmonics present in the supply voltage originating from the frequency converter will further increase these harmonic iron losses.

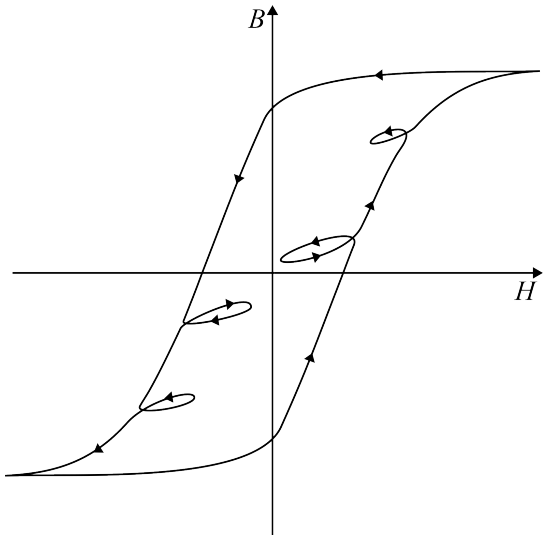


Figure 14: Minor hysteresis loops caused by rapidly fluctuating H.

As Slemon (1992) explains, harmonic iron losses be visualised as a number of smaller hysteresis loops along the path of the largest hysteresis loop that represents the fundamental frequency. The size of these smaller loops is dependent on the magnitude

of the harmonic content, and each additional loop incurs additional hysteresis losses. Formation of smaller hysteresis along the path of the main hysteresis loop is illustrated in Figure 14. The switching frequency and topology of the converter used affects the harmonic content. The magnitude of harmonics can be kept low if a high switching frequency is used. Similarly, the level of voltage levels the converter is able to produce affects magnitude of voltage transients in the output and thus the magnitude of harmonics in the output voltage.

In drive design, the selection of switching frequency involves a trade-off. Typically, the motor would benefit from higher switching frequencies and a greater number of output voltage levels in increased efficiency and decrease in design limits such as an insulation voltage rating. However, an increase in switching frequency typically increases switching losses in the converter. Increasing the number of output voltage levels increases complexity, and subsequently, cost.

2.5 Overview of common converter control methods

2.5.1 Scalar control

Scalar control is a simple control method based on controlling the supply voltage's magnitude and frequency. Under scalar control, the supply voltage's magnitude and frequency are controlled simultaneously to keep their proportion constant. For this reason, scalar control is also known as the v/f control. The goal is to achieve stator flux that is approximately constant (Finch and Giaouris, 2008).

The simplicity of scalar control limits its usefulness. The simple principle of maintaining constant v/f proportion assumes steady-state operation (Finch and Giaouris, 2008). Owing to this, the method is not well-suited to handle transients in load torque or speed. Furthermore, if operated open-loop, the scalar control does not offer accurate speed control.

While it is typical to perform scalar control in an open loop without feedback, closed-loop implementations that offer better accuracy and transient performance also exist (Chan and Shi, 2011). As Geyer (2017) notes, it is possible to also improve the performance using OPPs in conjunction with the scalar control, although this method requires a controller with a high bandwidth and can thus be impractical to implement.

2.5.2 Field oriented control

Field oriented control (FOC) aims to address the shortcomings of scalar control by controlling the driven motor more directly. Whereas scalar control does not offer control over the phase of the converter's output voltage with respect to the rotor, FOC is able to generate voltage where both magnitude and phase are referenced to the motor's rotor field. This type of control is also known as vector control, because it is based on synthesising voltage vectors at desired magnitude and angle frequency. FOC can control the motor more accurately than scalar control, resulting in superior dynamic performance.

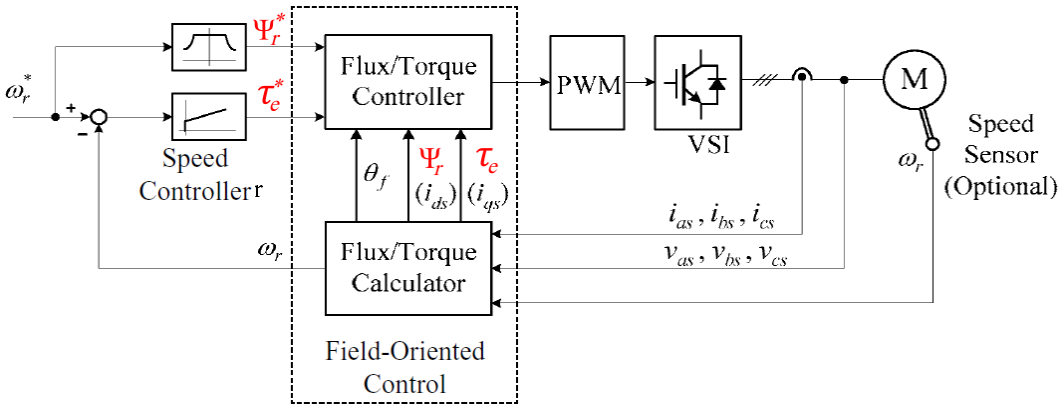


Figure 15: Simplified block diagram of a FOC control system (Wu, 2006). The figure has been edited to change the symbol for flux from λ to Ψ and the symbol for torque from T to τ to make it consistent with the convention used in the thesis. The edited symbols are highlighted in red.

Figure 15 displays an overview of a FOC controller. The FOC controller takes flux reference Ψ_r^*, torque reference τ_e^*, as well as the motor's phase currents and voltages as an input. The flux/torque calculator utilises phase voltages and currents and optionally a speed sensor to obtain the motor's flux and torque. They are then compared to the reference values by the flux/torque controller then generates a control signal based on them.This control signal is then used to generate the waveforms for driving the converter's switching devices. In the case of example in Figure 15, the output of FOC flux/torque controller is entered into a PWM controller, which then generates PWM waveforms for driving the IGBTs in the converter.

The FOC controller requires coordinate transformations in order to operate. This is due to the fact that the flux/torque converter depicted in Figure 15 operates in the synchronous reference frame of the rotor. This simplifies implementation of the actual controller because sinusoidal signals are converted to DC signals in the synchronous reference frame. Similarly, the output of the controller needs to be transferred back to the abc reference frame that represents the voltages of the motor's individual phases. These coordinate transformations present additional computational overhead on top of the calculations performed by the controllers.

As Wu (2006) notes, flux/torque calculator is the most important component of a FOC controller. Because FOC utilises a closed control loop where motor's actual flux and torque values are compared to their references, it is imperative that the values provided by the flux/torque calculator are correct.

2.5.3 Direct torque control

Direct torque control (DTC) is a vector control method that features performance enhancements over FOC. DTC is based on hysteresis controllers, which makes it possible to simplify the control loop over FOC.

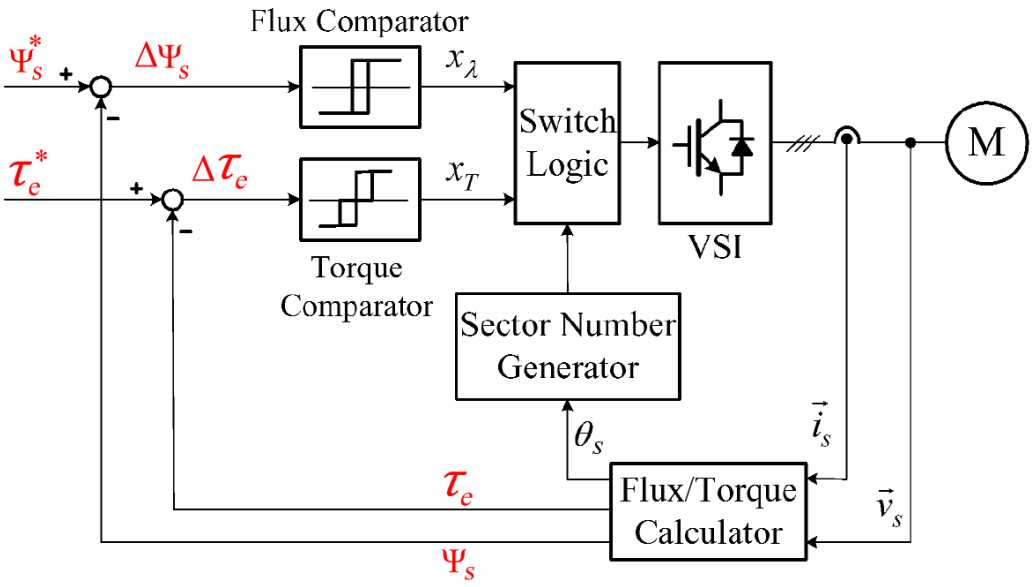


Figure 16: Simplified DTC control system (Wu, 2006). The figure has been edited to change the symbol for flux from λ to Ψ and the symbol for torque from T to τ to make it consistent with the convention used in the thesis. The edited symbols are highlighted in red.

Similarly to FOC controllers, a DTC controller, depicted in Figure 16, features flux and torque controllers and a flux/torque calculator. However, whereas in FOC the controllers operate in synchronous reference frame, which requires coordinate transformations, the hysteresis controllers used in DTC operate directly in the stator's reference frame. This means coordinate transformations are not required. The hysteresis controllers operate by trying to keep the flux and torque within a defined hysteresis band; they only produce an output other than zero when the error is outside the band. Due to this, another simplification is possible. The output of the hysteresis controllers can be directly utilised to operate the converter's switching devices without an intermediary PWM controller.

As noted by Casadei et al. (2002), DTC features numerous advantages over FOC. DTC requires less computations because coordinate transformations and PWM generation are not required. DTC is also less sensitive to inaccurate motor parameters than FOC. Whereas FOC depends on stator current, rotor resistance and rotor inductance for flux calculation, DTC only requires stator resistance (Begh and Herzog, 2018). However, in DTC, switching signals are produced by the hysteresis controllers that activate when flux or torque stray outside the hysteresis band. This makes the switching behaviour of DTC unpredictable when compared to PWM, where switching happens deterministically dictated by the modulation frequency and duty cycle. As noted by Geyer (2017), DTC tends to have a high current and torque TDD. Additionally, Casadei et al. (2002) mentions poor performance at very low speeds as a downside of DTC.

2.6 Model predictive pulse pattern control

Minimising switching losses in the converter and harmonic distortions in the driven equipment are antithetical objectives in power electronic control of electric motors. High switching frequencies increase the switching losses of the semiconductor switches in the converter, because the frequency of switching operations increases accordingly. In turn, low switching frequencies increase the harmonic losses in the driven equipment because they increase the TDD of the supply current. Figure 17 illustrates the inverse proportionality between total demand distortion (TDD) of converter output current and switching losses.

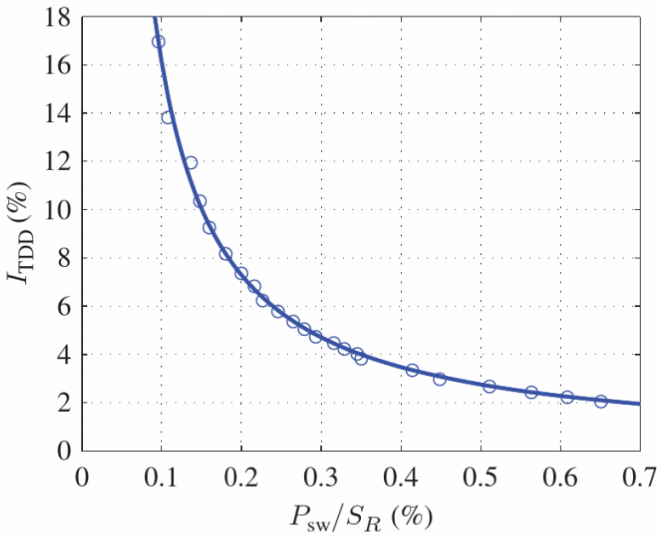


Figure 17: Total demand distortion of converter output current I_{TDD} as a function of converter switching losses P_{sw} normalized by converter apparent power S_R (Geyer, 2017, p.121).

Furthermore, control of high-power MV electric motor drives is a challenging task where fast impulse response is desirable in addition to low current distortions. As discussed previously, DTC can be used to achieve control with fast impulse responses, albeit with high amount of current and torque TDD present. Conversely, according to Geyer (2017), a steady-state operation with low harmonic distortions is possible by utilising scalar v/f control in conjunction with optimised pulse patterns (OPPs). Model predictive pulse pattern control (MP3C) is a control method that aims to offer both high dynamic performance and low harmonic distortions while minimising switching losses. To achieve this, it uses model predictive control (MPC) in conjunction with OPPs.

2.6.1 Model predictive control

Model predictive control (MPC) is a well-documented control method that has seen practical applications in the process industry for several decades Geyer (2017). It

has seen use in industries such as chemistry, aerospace and food processing. In power electronics, however, MPC has only been utilised in limited applications until lately. A prevalent reason for this, as Geyer (2017) notes, is that in power electronics, real-time control is required. Solving the control problem in real time with sufficient performance is challenging, and has only become feasible fairly recently.

While numerous differing approaches to implementing MPC exist, Geyer (2017) lists five defining attributes that are common to all MPC implementations:

1. internal dynamic model
2. constraints
3. cost function
4. optimisation stage
5. receding horizon policy

Internal dynamic model of the controlled system makes it possible to predict the future states of the system. More specifically, the internal dynamic model describes how the system's state vector $\mathbf{x}$ evolves in time. As described by Geyer (2017), the state vector contains the components of the system that are required for control. With electric motors, such components could be quantities such as stator voltage, stator current and motor speed. As Geyer (2017) describes, the state-space behaviour of power electronic systems when modelled this way is typically linear.

Constraints arise from limits in the controlled system. As an example, a five-level inverter is only able to produce five different voltage levels instead of a continuous distribution of voltages. As Geyer (2017) points out, this means, that even if the relationships contained in the dynamic model are linear, the constraints make the system as a whole nonlinear. In addition to constraints arising from physical reality, they may also be added to ensure the system's operating parameters stay in safe limits.

Cost function describes the cost of attaining the control objective. As described by Geyer (2017), the cost function generates a scalar cost value that is based on a sequence of future system states, outputs and variables. MPC aims to minimise the value of the cost function in order to find the most suitable set of control parameters. The cost function is chosen based on the properties of the system being modeled. In drive systems where the aim is often to reduce motor supply current distortions. That is also the case with MP3C, where the cost function is typically based on minimising the motor current distortion (Geyer, 2017).

Optimisation stage involves finding minimal solutions to the cost function that satisfy the constraints in the system. The opimisation stage yields an optimal sequence of the system's manipulated variables $U_{opt}(k)$ (Geyer, 2017). There, k denotes the future time step. Two approaches exist for solving the optimisation problem during this stage: online solving and offline solving (Geyer, 2017). Online solving means that the optimisation problems are solved in real time. In offline solving, the optimisation problems are instead solved beforehand for all possible states of the system. The

results can then be stored and accessed during the runtime. Geyer (2017) refers to the practice of solving the optimisation problem in advance and the retrieving them from a look-up table during the runtime as explicit MPC. As noted by him, due to the computational cost of solving all of the possible states in advance, explicit MPC is only viable for low-dimensional systems with few parameters. Therefore he notes that explicit MPC is not suitable for electric drive applications.

Receding horizon policy describes how long into the future the sequence $U_{opt}(k)$ is formed and how the elements of $U_{opt}(k)$ are utilised. In essence, the receding horizon policy dictates that at time step k, only the first element of $U_{opt}(k)$ is applied into the system, while the optimisation problem is solved for future steps up until time step $k + N_p - 1$ (Geyer, 2017). This yields a prediction horizon that has the length of N_p. Correspondingly, during the next time step, element $U_{opt}(k + 1)$ is applied into the system and prediction at time step $k + N_p$ is added to the end of $U_{opt}(k)$.

Geyer (2017) lists numerous advantages for MPC. Control of non-linear systems is made easier by the fact that the nonlinear behaviour can be included in the dynamic model of the system. MPC is able to accept hard constraints on the system's inputs, outputs and states. In addition, use of cost function makes is possible to account for a multitude of control objectives. Conversely, implementation of MPC also poses challenges as listed by Geyer (2017). These include computational difficulties in solving the optimisation problem for all possible states, optimisation challenges in solving the control problem in real time, and finding and investigating new ways to formulate and solve MPC problems in different applications.

2.6.2 Optimised pulse patterns

Optimised pulse patterns are pulse width modulation (PWM) switching patterns that have been optimised to minimise the value of the cost function of the drive system. As Geyer (2017) explains, OPPs typically refer to optimising the switching pattern with respect to current distortions. The calculation of OPPs is typically done with an assumption of quarter wave symmetry of the reference waveform $u(\theta)$ (Geyer, 2017). For a waveform with period of 2π, quarter wave symmetry requires that

$$u(\theta) = -u(\pi + \theta) \tag{16}$$

$$u(\theta) = u(\pi - \theta) \tag{17}$$

Under the assumption of quarter wave symmetry, pulse number d is defined by Geyer (2017) as an integer that denotes the number of switching transitions in a single phase waveform within each quarter wave. Geyer (2017) defines the angles at which the transitions happen during the first quarter of the fundamental wave as primary switching angles $\alpha_1 ... \alpha_d$.

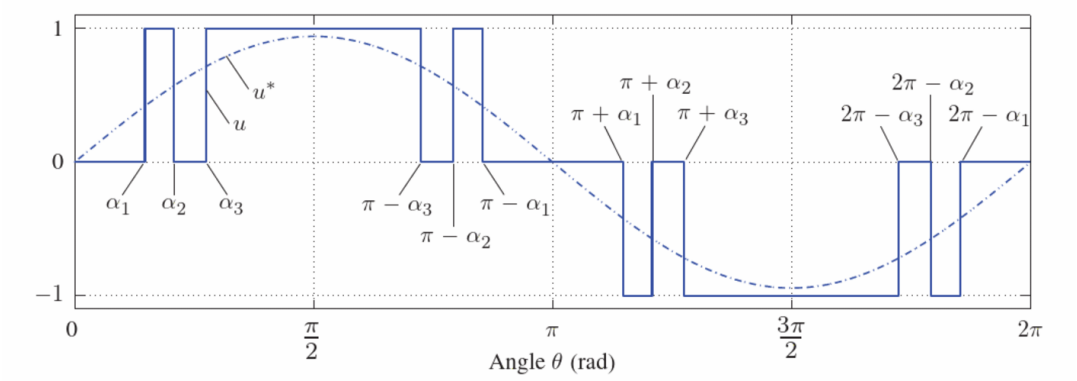


Figure 18: A pulse pattern with quarter-wave symmetry with d=3 with primary switching angles at α_1, α_2 and α_3. (Geyer, 2017, p.105)

Figure 18 depicts a pulse pattern with d=3. As can be seen, the waveform is demonstrates quarter wave symmetry according to equations (16) and (17).

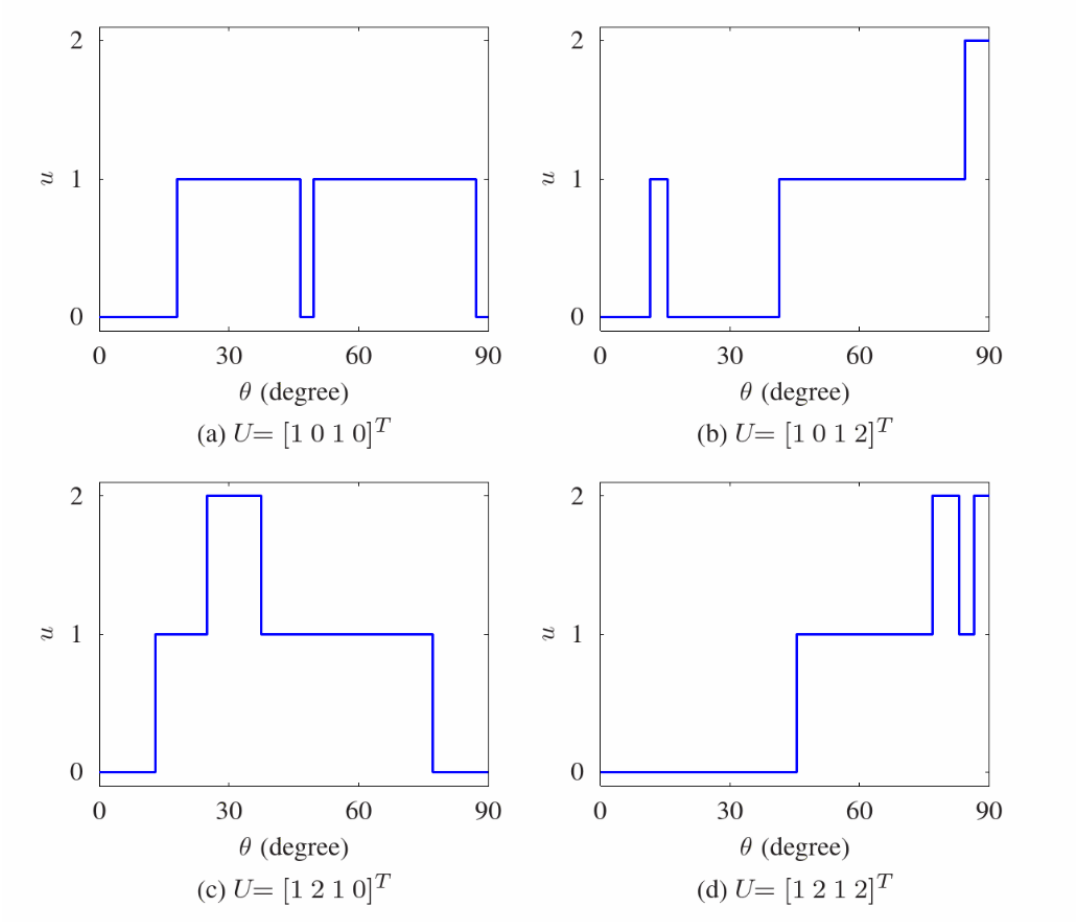


Figure 19: Switching sequences for pulse number d = 4 (Geyer, 2017, p. 113).

The method for calculating OPPs is dependent on the application In the case of

five-level MV converters such as the one examined in this thesis, they are calculated by minimising a cost function for each pulse in the pattern that describes the TDD of the converter current and is dependent on the primary switching angle α_i and switching transition Δ_i (Geyer, 2017). The current TDD is part of the cost function in addition to the switching angle due to number of voltage levels present. As Geyer (2017) notes, a five-level converter has a set of five possible single-phase switch positions: $[-2, -1, 0, 1, 2]$. Because OPPs are calculated with quarter-wave symmetry imposed, only the first half of the positive half-wave of the fundamental is considered. This leaves three switch positions, $[0, 1, 2]$ that have to be considered for OPP calculation. As a result, at voltage level 1, switching transition can either happen upwards to voltage level 2 or downwards to 0, which adds an degree of freedom to the cost function. Figure 19 demonstrates this by depicting possible switching sequences for pulse number $d = 4$.

As listed by Geyer (2017), the nature of OPPs gives rise to multiple interesting properties. Firstly, because OPPs are defined by an integer pulse number d which is in turn related to the number of pulses per quarter wave, OPPs are always synchronized with the fundamental wave. As a result, OPPs do not feature harmonic components below the fundamental frequency. However, unlike regular pulse width modulation where the modulation cycle has a fixed length, the length of the modulation cycle with OPPs is variable, which makes it not possible to directly sample the current to obtain the fundamental component. This is due to the ripple currents from the OPP interfering with the reading. As a result, the use of OPPs have typically been restricted to slow control methods, such as scalar control, where the ripple currents due to OPP are low due to limited bandwidth. Geyer et al. (2012) note that this problem is particularly prominent with MV converters that have slow switching frequencies, such as the converter examined in this thesis.

2.6.3 Properties of MP3C

Model predictive pulse pattern control combines MPC and OPPs to achieve control system that has both fast transient response and low current distortions. MP3C is based on offline-computed OPPs, but unlike other methods utilising offline-computed OPPs, it is able to model the system state (in this case, by following the trajectory of the motor's flux vector) and use this model to modify the OPPs at runtime without needing to completely recalculate them (Geyer et al., 2012). As a result, MP3C is able to provide fast transient response while not requiring complex OPP calculations during runtime, which significantly lowers requirements on the controller. The ability to modify the OPPs at run time addresses the issue of unregular modulation cycle length discussed previously, because the ripple current caused by OPPs can be accounted for when sampling the current (Geyer, 2017).

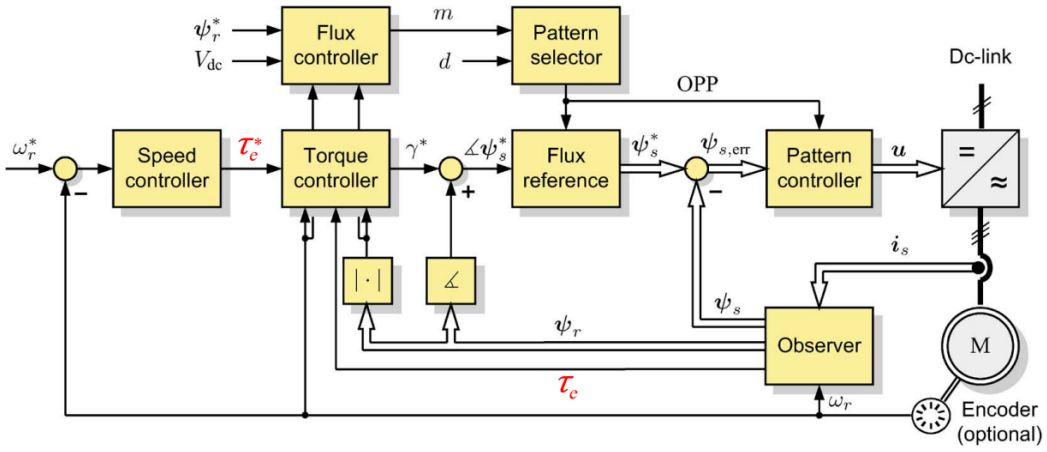


Figure 20: Block diagram of MP3C controller (Geyer et al., 2012). The figure has been edited to change the symbol of torque to from T to τ to make it consistent with the convention used in the thesis. Changed symbols are highlighted in red.

A block diagram of a MP3C controller is shown in Figure 20. The system consists of two control loops. A speed controller forms the outer loop, whereas the inner loop consists of a torque controller, flux controller and a pattern controller. Additionally, the inner loop features a pattern selector, which selects the OPP to be used, and an observer providing the torque controller with torque reference.

Control scheme	Control setting	f_{sw} [Hz]	$I_{s,\mathrm{THD}}$ [%]	$\tau_{e,\mathrm{THD}}$ [%]	$I_{s,\mathrm{THD}}$ [%]	$\tau_{e,\mathrm{THD}}$ [%]
CB	$f_c = 250$ Hz	150	16.1	11.0	100	100
SVM	$f_c = 250$ Hz	150	15.5	9.83	96.8	89.6
MP^{3}C	$d = 3$	150	7.36	6.62	45.9	60.3
CB	$f_c = 450$ Hz	250	7.94	5.79	100	100
SVM	$f_c = 450$ Hz	250	7.71	5.35	97.1	92.4
MP^{3}C	$d = 5$	250	4.13	3.41	52.0	58.9
CB	$f_c = 750$ Hz	400	4.68	3.41	100	100
SVM	$f_c = 750$ Hz	400	4.52	3.06	96.6	89.7
MP^{3}C	$d = 8$	400	3.63	2.88	77.6	84.5

Figure 21: Performance of MP3C at steady-state when controlling an induction motor compared to carrier-based PWM (CB-PWM) and space vector modulation (SVM) (Geyer et al., 2012). The figure has been edited to change the symbol of torque from T to τ to make it consistent with the convention used in the thesis. Changed symbols are highlighted in red.

Geyer et al. (2012) evaluated the performance of MP3C in terms of steady-state performance, dynamic performance and tolerance to motor parameter variation. As

seen in Figure 21, the current and torque distortion is significantly reduced at steady state when compared to carrier-based PWM and space vector modulation control schemes. In terms of dynamic performance, Geyer et al. (2012) found it to be sufficient for the most common use cases. MP3C was also found to be robust in terms of motor parameter variation by Geyer et al. (2012), with the steady-state torque and flux change under 0.5% when the stator and rotor resistances were changed from 75% to 125% of their initial value in different combinations. Furthermore, they also found that parameter variation has little effect on the dynamic performance of MP3C.

For the purposes of this thesis, the most important consideration of MP3C supply is the harmonics present in the motor supply current, as it is the primary driver for MP3C-borne losses in the motor. Because the OPPs used are calculated off-line, the voltage generated with MP3C is more deterministic than with DTC. This potentially makes it easier to train a machine learning model to estimate the MP3C losses. The frequency converter examined in this thesis is, in principle, able to use different set of OPPs depending on the desired application. This could have implications in terms of the validity of the estimates made by the estimator developed in this thesis. It's likely that each new set of OPPs needs to be handled separately. For the training data generated in the thesis, it was manually verified that the drive simulator software used to generate the MP3C voltage waveforms used same OPP for each entry in the dataset.

3 Methods

This chapter introduces the research methods and approach taken in the thesis. This includes description of the implemented estimator program and its features, along with justifications between the design decisions made. In addition, methodology for collection of the training data used is outlined.

3.1 Overview

The estimation program implemented in this thesis utilises a collection of machine learning (ML) regression models combined with selection logic to determine which model to use, and an interpolation logic to provide the final result. In essence, the approach is to use several separate regression models that are each trained with pre-calculated loss results from a set of source motors. Each separate regression model is then used to generate an estimate of motor losses at different speeds and load torque values. This yields a collection of loss estimates that can then be used to generate an estimate of losses at arbitrary speed and torque points. The speed and torque points at which the ML models are trained at are referred to as the estimation points from this point forward.

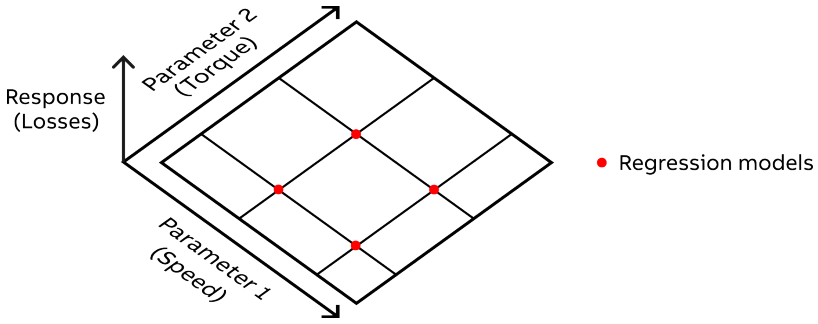


Figure 22: The concept of operating space. Regression models located in the operating space have been marked in red.

This thesis refers to the surface along which the estimation points are distributed as the operating space. The surface is defined by two orthogonal parameter axes. As mentioned, the parameters used in the thesis are the frequency f defined as a fraction of a motor's nominal frequency f_N, and torque τ defined similarly as a fraction of nominal torque τ_N. Because the power output of a PMSM is a product of torque and speed, this representation provides a way to conveniently define the range of expected operating conditions using two decimal numbers. Additionally, the operating space entails a third axis that is orthogonal to the parameter axes. This result axis represents the responses given by the regression model at that point in the operating space. In the case of this thesis, the result axis describes the estimate of additional losses given by the regression model. The concept of operating space and the orientation of its axes is illustrated in Figure 22.

Functionally, the operating space is a two-dimensional grid where the coordinates are

defined by f and τ. The estimation points are located on a grid in the operating space in positions defined by f and τ, with each estimation point containing the loss estimates at that point. Additionally, the operating space also contains a query point. The query point represents the torque and speed at which the loss estimate is generated and can be defined arbitrarily. Similarly to the estimation point, the query point contains the loss estimations. The loss estimations contained by individual estimation and query points are comprised of loss components. Each loss component is represented by a scalar value, which represents losses in one part of the motor. To generate an estimate for each loss component, each estimation point contains a ML model for each component.

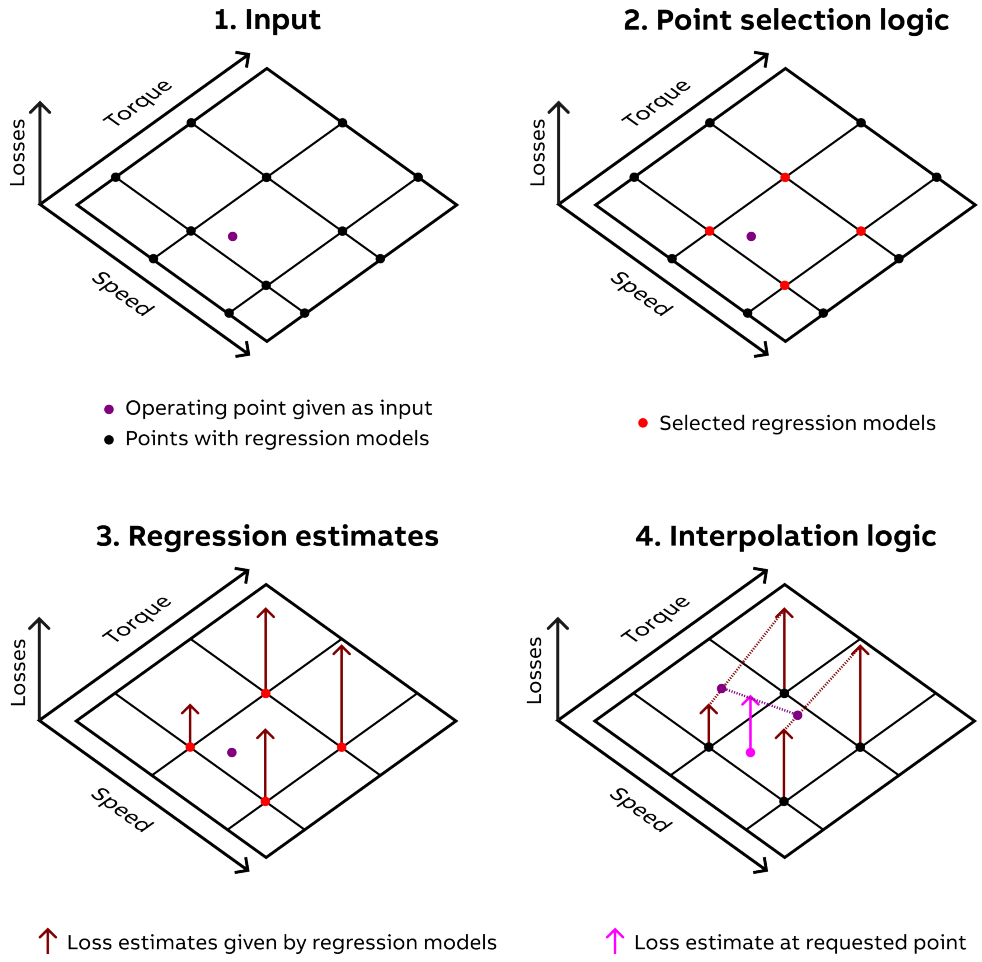


Figure 23: Illustration of estimating losses at arbitrary operating point.

The query point can be defined arbitrarily, whereas the estimation points have fixed locations on a grid. In order to generate estimates for locations in between the estimation points, the estimator utilises a combination of selection logic and interpolation logic. The selection logic is used to determine the estimation points that surround the query point. Because the estimation points are located on a grid, the selection logic always return four points that enclose the query point. The interpolation logic is then used to interpolate each loss component based on the loss components of the surrounding estimation points. The steps of estimating losses are depicted in Figure 23.

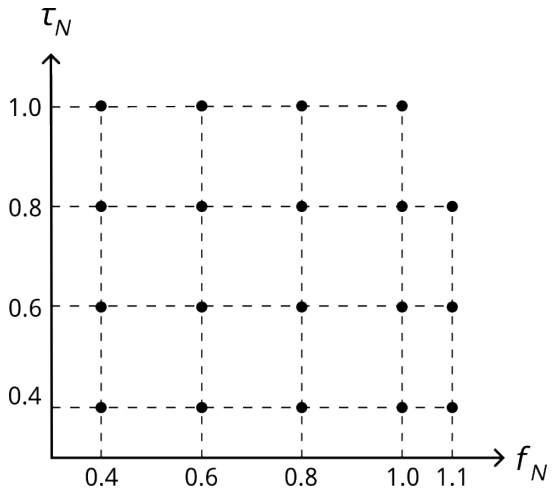


Figure 24: The operating space as implemented in the thesis. The dots denote estimation points.

The *operating space* as implemented in the thesis consists of 19 *estimation points*. It is bound between 0.4 and 1.1 relative frequency f and similarly between 0.4 and 1.1 relative torque τ, where coordinates $(1.0f_N, 1.0\tau_N)$ denote the motor's nominal point in terms of frequency and torque. The configuration of the estimator's operating space is shown in Figure 24. The number of points was a compromise between accuracy and the amount of training data required. Because each point requires an individual set of training data, the number of points was constrained by the speed of training data generation process. Because the data generation process was time-intensive, the number of points was kept fairly low. Similarly, the limits for f and τ were based on a combination of constraints and practical considerations that are explained in detail below.

The frequ lower limit of $0.4f_N$ was defined by the fact that the software controlling the drive control software switches over from MP3C to DTC at a fixed frequency. For most of the motors in the training data set, this frequency falls between 0.3 and 0.4 relative f. This means that the training data of MP3C losses could not be generated for all motors at frequencies below $0.4f_N$. The upper bound for f was selected as $1.1f_N$ based on similar constraints. The drive simulator software experienced difficulties with simulating some of the motors runnning at the frequency of $1.2f_N$. Therefore, a decision was made to define the upper bound of f at $1.1f_N$ instead of $1.2f_N$, that would have been consistent with the estimation point spacing of $0.2f$ below f_N.

The lower bound of τ was defined based on practical considerations. Because collecting the training data was time-intensive, including more points increases the workload. In general, PMSM losses can be assumed to be proportional to power output, and consequently the load torque. One of the primary uses for information about motor losses is estimating how much heat is generated in the motor. For this purpose, the returns for estimating losses gather diminishing returns at low torque. Because the primary goal for this thesis was to evaluate the feasibility of a specific loss estimation

method, including very low load torques in the training data was thought to provide little additional value.

The upper bound of τ was defined based on the nominal torque τ_N below the nominal frequency. As shown in Figure 24, the operating space does not contain an estimation point at $(1.1f_N, 1.1\tau_N)$. This is due to the constraints of the overspeed regime. Because the power output of the motor is the product of speed and torque, the motor delivers nominal power when operating at the nominal frequency and delivering the nominal torque. Consequently, when operating at overspeed, the torque equating to nominal power output is lower than the nominal torque. Therefore, operating at point $(1.1f_N, 1.1\tau_N)$ would exceed the nominal power output of the motor and thus it is not included in the operating space. This limits the upper bound of τ in the overspeed region to $0.8\tau_N$

Ability to integrate with the existing software tools and workflows was an important consideration in the process of defining requirements and developing the estimation method. This influenced the design of the interfaces for the estimator program. Namely, the program built to be configured and run from a command line. This enables the program to be either manually run by an operator or function as a part of a more complex system where the estimator program is run by an another program. Inputs and outputs are files located in a working directory of the estimator program. When the program is launched, the estimator reads the input files and generates output files.

The file-based input/output scheme further increases flexibility in terms of integration with other programs. An another program can write the estimator's input files and read its output files without additional configuration. Additionally, the working directory is configurable, which enables the estimator program to function as a part of several systems at once. As an examble of this, if two separate programs want to interface with the estimator program, they can both run it with their specific working directories as an argument. Inputs and outputs are handled in two separate working directories without any additional configuration needed.

3.2 Program environment

The *estimator program* implemented in the thesis integrates with, and depends upon two existing in-house programs. The relationship between the different programs is depicted in Figure 25. The *motor design software* is used to design PM motors. It contains calculation modules for calculating PM losses, among many other operational parameters. The estimator program implemented relies on the motor design software to retrieve the motor design data in order to generate an estimate. Motor loss data used to train the regression models is also generated with the motor design software.

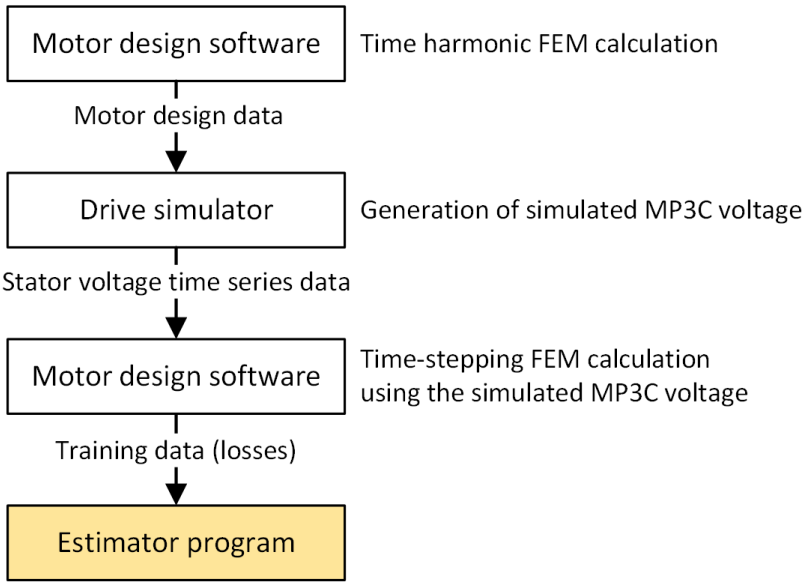


Figure 25: Relationships between the estimator program and existing in-house software.

The drive simulator software is used to simulate a drive system consisting of a MV motor and a frequency converter utilising MP3C. It is given the motor's design data as an input. It then is able to run a simulation of the drive system where speed and torque can be changed at will while converter and motor parameters can be monitored. The drive simulator is used in the thesis to generate MP3C stator voltage time series data for each motor at each operating point. These voltage waveforms are then used as an input for a FEM calculation run in the motor design software. This FEM calculation then produces the loss data used for training.

The motor design software is not able to use the voltage time series data output from the drive simulator directly. Instead, a converter program is required to modify the data. This involves changing the way the data is formatting, as well as modifying the beginning of the time series data to ensure the stability of the FEM calculation done by the motor design software. The converter program is omitted in Figure 25 omitted for brevity.

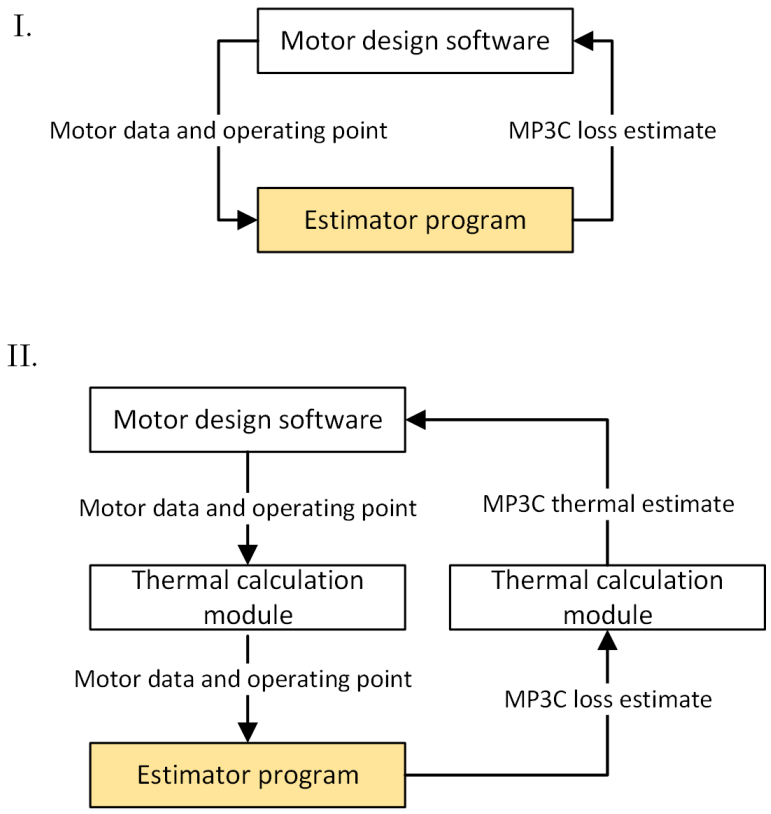


Figure 26: Different potential use cases for the estimator program. (I) integrated into the motor design software and (II) integrated as a part of thermal calculation module

The estimator program was designed to provide input for programs used in the process of designing and configuring PM motors. The eventual use case is intended to be fast loss estimation during engineering and sales activities. As an example, a sales configurator software could run the estimator program to quickly provide the user with an loss estimate. Thermal calculation is another potential use case. In order to generate thermal estimates, the motor design software uses a thermal calculation module which needs the losses at different parts of the motor as an input. The thermal calculation module could call the estimator program in order to provide estimates of thermal impacts of MP3C control. Figure 26 illustrates different potential use cases for the estimator program.

3.3 Program architecture

The system built over the course of this thesis consists of multiple parts serving different purposes. The intention was to produce a working system that could potentially be used as a basis for further development and eventual integration into the design software toolchain. Therefore, the estimator program described here includes, in addition to the core estimation logic, significant amount of code involved with practical aspects of

handling inputs, outputs and interfacing with users and other programs. Therefore, the final estimator program is fairly complex.

The estimator program was built from the ground up to be modular in structure. In practice, this modularity was achieved by implementing each functional block as a separate Python class. This logical segmentation of the program serves two primary purposes. Firstly, it improves readability and maintainability of the codebase. Secondly, the program was built with the consideration that the potential end goal is integration as a part of motor design software toolchain.

Because the program is modular, changes to a single module can be made without affecting other parts of the system as long as the module interfaces do not change. This means that modules can also be refactored in other programming languages without affecting the other modules. Python was selected as a language because of ease of use and extensive module library. However, it was clear from the start that the potential final program could not use Python for performance and compatibility reasons. Instead, the final implementation would be written in lower-level language such as Rust or C. The modular approach allows for modules to be refactored in more performant languages one by one. The refactored modules can be evaluated as a part of the program with minimal changes to the other modules. This has the potential to make the implementation process significantly faster and easier. In addition, one by one refactoring allows for easily evaluating the performance impact of changes made to one module.

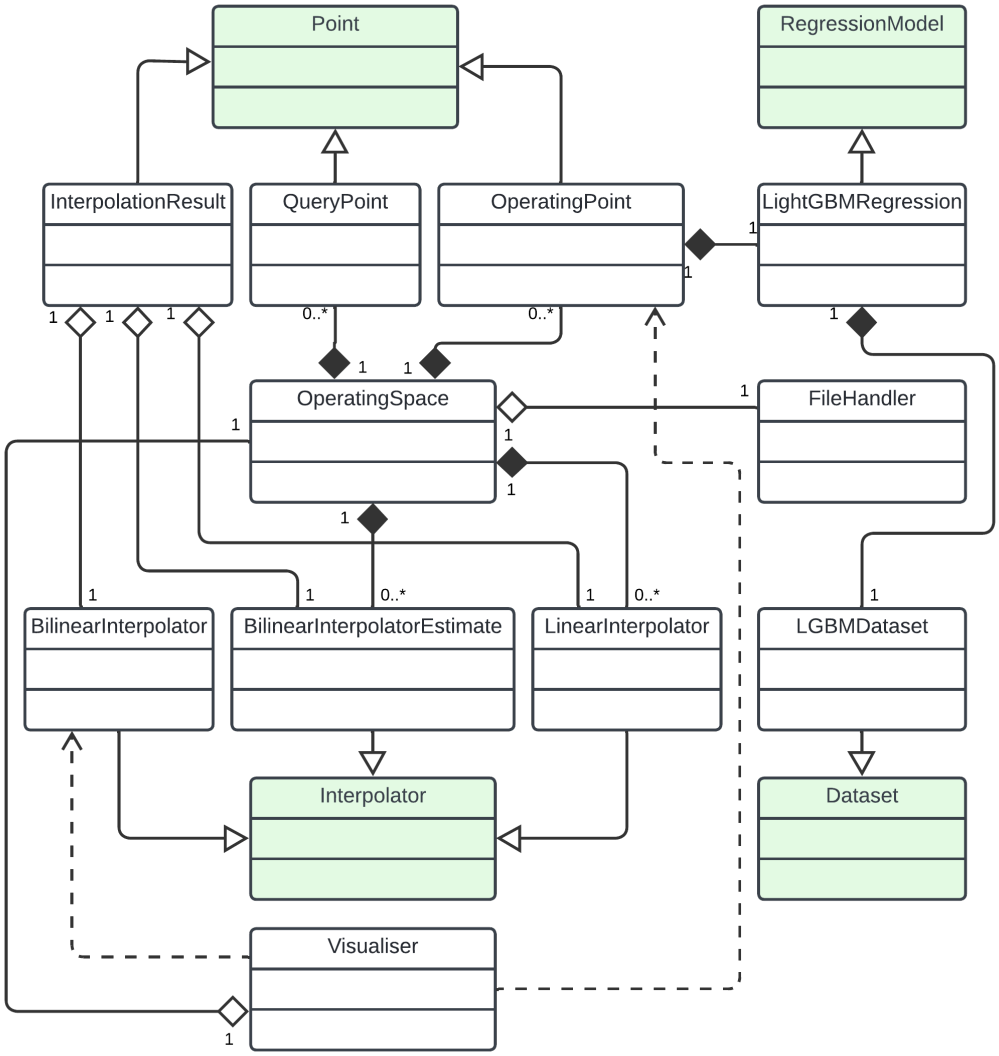


Figure 27: Class diagram of the estimator program. Class attributes not shown for simplicity. Abstract base classes shown in green.

A simplified unified modelling manguage (UML) class diagram of the estimator program's structure is shown in Figure 27. The UML diagram visualises the different relationships between the classes. The OperatingSpace class envelopes most of the logic needed for estimating the losses either in itself or through its component classes. During execution of the program, exactly one instance of OpeatingSpace is created and most of the operations needed to generate an estimate are performed using this instance's methods.

Different operating points are represented by instances of OperatingPoint class. This class contains the operation point-specific dataset, as well as logic required to initialise the point-specific estimation model and generate the estimation results by. The training

data is provided by and the results are collected by an instance of OperatingSpace. More specifically, OperatingSpace initialises an array storing the OperatingPoint instances. Each point is provided with data loaded from a point-specific data file. Training of the point-specific models is then performed by iterating through the list of OperatingPoint instances and calling each point's training method. Similarly, estimation results for each points are generated by iteratively calling each point's estimation method. OperatingPoint class utilises a LightGBMRegression class to train and save the point-specific ML model and get estimates.

LightGBMRegression class implements the ML regression algorithm that performs the actual estimation. Each instance of OperatingPoint has a collection of instances of the LightGBMRegression class. Each of these instances represents one loss component to be predicted. The LightGBMRegression class is responsible for training models and generating estimates, as well as the input and output operations involved with models. LightGBMRegression utilises LightGBMDataset class to store and manipulate the training data. LightGBMDataset contains a method for splitting the training data loadaded from a file into a training dataset and a validation dataset. The LightGBMDataset then stores the two datasets for later use by an instance of LightGBMRegression.

Some classes have abstract base classes (ABCs) which are shown in Figure 27 in green. The purpose of a ABC is to define the common attributes and methods for similar classes. The child classes share the attributes and methods defined in the base class. This makes implementing similar classes easier by removing the need to define identical attributes and methods multiple times. Additionally, the readability and understandability of the source code benefits from defining common traits in the ABC.

As an example, the Point ABC has attributes x and y, which means that all child classes of Point inherit these attributes. Therefore, when making changes to the program, the fact that all children of Point have these attributes can be relied upon. As another example, the RegressionModel ABC defines the attributes and methods required for a ML regression model. Therefore, implementing another algorithm in addition to LightGBMRegression is simply a matter of writing another child class for RegressionModel and changing the OperatingPoint class to call the new class instead of LightGBMRegression.

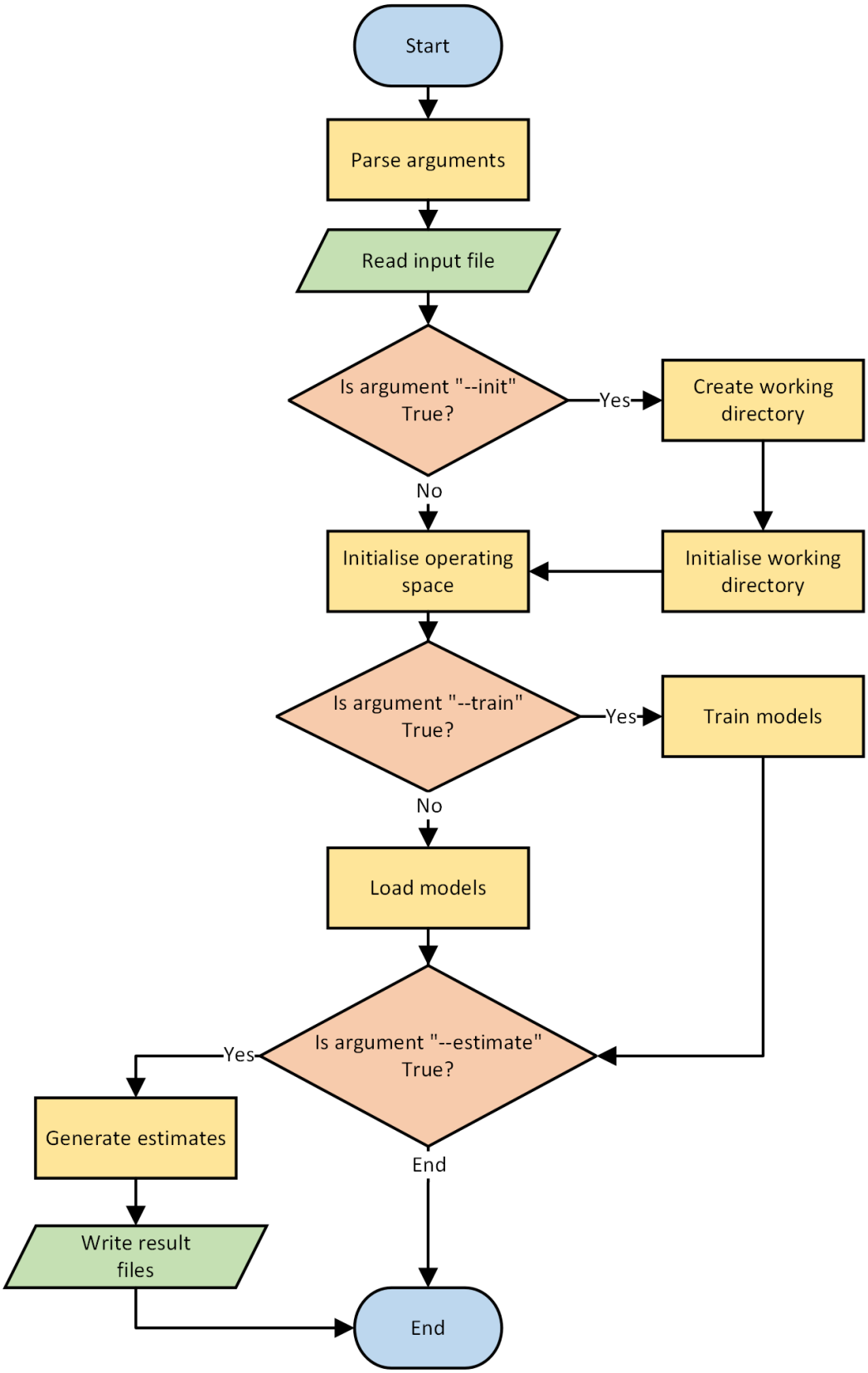


Figure 28: A flowchart of estimator program operation.

A flowchart of the estimator program operation is shown in Figure 28. The estimation program begins by parsing user-provided arguments. These include the filename of the input data, information about the operating point to generate an estimate for, and optionally, a temperature used to temperature-compensate the results. Then, the input file is read to load the input data. If the argument '–init' is passed as an argument, the program then initialises a new working directory, along with default parameter and definition files. A parameter file is used to define the data items to use as features, the data items to estimate, and LightGBM model parameters. The definition file contains listing of the training data and saved model filenames for each of the estimation points.

If '–init' is not passed, the program attempts to read existing parameter and definition files. If this succeeds, an operating space is created based on the configurations in these files. If the argument '–train' is passed, the program then proceeds to train the models at each point using the data from the dataset files. Following this, if argument '–estimate' is given, the estimator program generates an estimate for each of the estimation points. Then, the results are written in the results files and the program ends.

3.4 Regression model

The ML framework used to implement the estimation points is LightGBM. It is based on decision trees, which are a commonly-used ML algorithm. LightGBM, specifically, is an improvement on decision trees aiming to improve efficiency and scalability with a method called gradient boosting (Ke et al., 2017). Decision trees are well-suited for the purposes of this thesis for several reasons. Their advantages include ability to handle outliers, non-linear relationships and missing values in the training data (Attard, 2024).

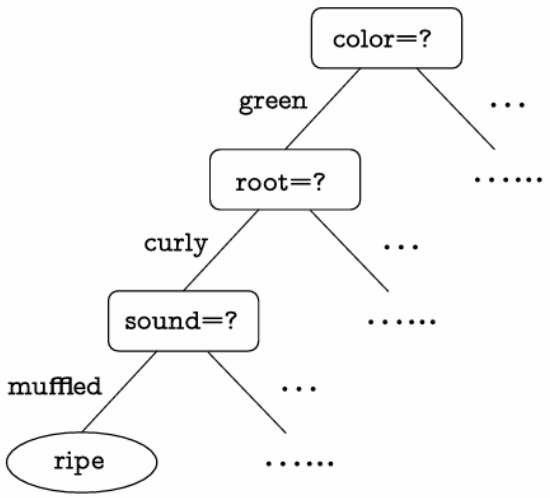


Figure 29: Concept of a decision tree for sorting watermelons (Zhou, 2021).

The basic principle of a decision tree algorithm is shown in Figure 29. Specifically, the decision tree shown describes steps for sorting watermelons in terms of colour, root shape, and sound in order to determine whether it is ripe. Decision trees in their basic form are only suitable for making classifications based on discreet values, as shown in Figure 29. In this thesis, however, the loss values in the training data are continuous and thus this approach does not work. Zhou (2021) explains that decision trees can be adapted to handle continuous values by introducing points at which the continuous values are split in order to form the branches of the decision tree.

Boosting, as defined by Zhou (2021), is a method that combines multiple ML algorithms in order to improve the response of a model. Gradient boosting is a ML method that combines several sequentially trained decision trees to arrive at a result (Ke et al., 2017). While numerous gradient boosting decision tree (GDBT) implementations exist, LightGBM was chosen because it has previously been used with good results in similar in-house engineering progmams, and because it is well-documented and has Python support. Additionally, LightGBM provides an application programming interface (API) for C, which, as a low-level language, is a more suitable candidate for production software. This is beneficial from future development point of view. If the system presented in this thesis is adopted for production, LightGBM's C API provides a direct pathway for low-level, high-performance implementation.

As previously discussed, the other parts of the estimator program are agnostic in terms of the ML model used. The program was built from the ground up to allow for refactoring of the ML module without requiring changes to the other modules. This is of special importance because performance of the LightGBM model could not be properly verified due to the challenges with generating training data. The modular structure makes process for implementing another ML model straightforward, should it be necessary in the future.

3.5 Point data generation

Two different approaches for estimating the losses were considered. The first approach involved using the motor's direct on line (DOL) loss data as a base, and then training the regression models on additional losses caused by MP3C. In essence, with this approach the estimator deals with differences between DOL and MP3C results. In the second approach, the regression models are simply trained with MP3C loss values instead of differences.

The benefit of the first approach is that since the regression models are trained on the differences between DOL and MP3C, they are effectively trained on the only difference that matters. Because the objective is to know the differences in losses between DOL and MP3C operation, the full magnitude of losses is not important. While no quantitative examination was performed to this effect, the assumption was that this method would result in more accurate predictions since the possible errors and inaccuracies would only apply to the difference between DOL and MP3C, not the entire magnitude of losses.

The primary benefit of the second approach over the first is its simplicity. For this approach, no DOL losses are required as an input data. Instead, the models are trained to estimate the complete loss components under MP3C. Thus, as opposed to the first approach, the second approach requires no pre-calculated loss value and just the motor's design data is sufficient as an input. This approach is potentially more inaccurate because the regression model generates the MP3C loss estimate without using calculated DOL losses as a base.

The choice between the two approaches was dictated by technical limitations. The first approach requires pre-calculated DOL losses. The motor design software handles the calculation of DOL losses separately for each loss component. For example, resistive losses in the stator widning and the iron losses in the stator are calculated as separate element. Two different approaches are used in the design software for calculating these components. For components where it is applicable, a time harmonic FEM calculation is used. In time harmonic FEM, losses are calculated at a steady state with the magnetostatic method. Time harmonic FEM calculations are quick, taking at most seconds per calculation. Some loss components, however require more time intensive time-stepping FEM calculation where the behaviour of the motor is simulated through discrete time steps. This is significantly more time-intensive. In order to calculate accurate losses at an operating point, the simulated motor's speed and torque must stabilize. This necessitated running long time-stepping FEM calculations in the order of 10-20 supply voltage periods long, taking several hours for one motor at one operating point.

Time harmonic calculation of losses in the motor design software is only possible for some of the stator loss components. Calculation of any rotor loss components requires time-stepping FEM. The goal was to develop a solution that would allow for quick assessment of losses. Therefore, the approach where DOL losses are used as a baseline was not deemed feasible, and the second approach where the estimator generates the loss estimates without DOL baseline was chosen. The estimator program was still built to facilitate the DOL baseline method. This was done for two reasons. Firstly, this allows the estimator program to be used for other purposes than PM loss calculation where the baseline data might be more easily available. Secondly, in-company research to the topic of fast PM rotor value calculation has been done. It is possible that the results from this research are incorporated into the motor design software in the future. This would make using DOL losses more feasible.

3.6 Selection logic

Selection logic is used to determine which estimation points are adjacent to the query point. Because point selection is performed as a prerequisite for interpolation, the selection logic and interpolation logic are inherently intertwined. As such, the selection of interpolation logic unavoidably sets the requirements for the selection logic. In the case of the estimator program, bilinear interpolation was chosen as the interpolation method.

Bilinear interpolation, as implemented in the program, requires exactly four estimation points in order to calculate a result. A notable practical implication of this is that the boundaries of the operating space are effectively defined by its outermost points in any direction, because the query point must be surrounded in all sides by the estimation points. The boundaries are, however, inclusive. Therefore, the interpolation logic is able to interpolate coordinates where either τ or f are located at their respective boundaries.

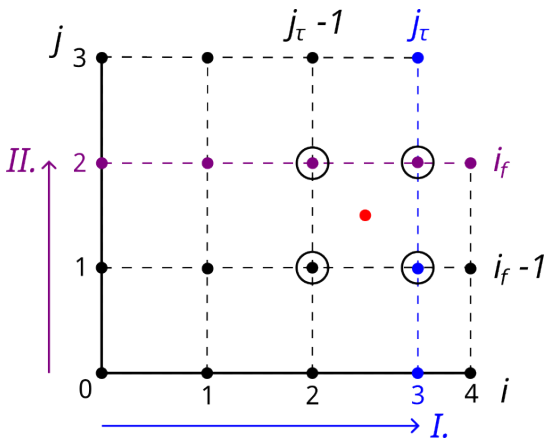


Figure 30: Point selection logic illustrated. Query point in red, the first traversal operation (I.) in blue and the second traversal (II.) in magenta. Selected estimation points have been circled.

The basic logic for the selection logic is illustrated in Figure 30. The estimation points are stored in an array sorted by τ and f. This enables the selection logic to iterate through the array and compare the coordinates of each estimation point to those of the query point. First, the array is traversed column-wise to locate the first column index i_f where the column f-coordinate is greater than or equal to the f-coordinate of the query point. Then, a similar traversal operation is performed row-wise to find the first row j_τ where τ is greater than or equal to the query point's τ. The selection logic then returns the estimation points at grid indexes (i_f, j_τ), $(i_f - 1, j_\tau)$, $(i_f - 1, j_\tau - 1)$ and $(i_f, j_\tau - 1)$.

If either traversal operation is completed without a match, the selection logic determines the query point is out of bounds and raises an exception. Similarly, an exception is raised if any of the four point is not found in the array, which similarly signifies that the requested point is out of bounds. The exception is captured by the code that calls the selection logic, which then notifies the user that the requested query point is invalid.

3.7 Interpolation logic

Interpolation between estimation points is performed using simple bilinear interpolation. The estimation points adjacent to the query point are first determined using the selection logic. The selection logic, if successful, always returns four estimation

points. If the query point is located at the exact location of one of the estimation points, the interpolation logic still utilises four adjacent points, but the relative weights of the other three points are zero and the interpolation result is equal to the value of the estimation point at which the query point is located.

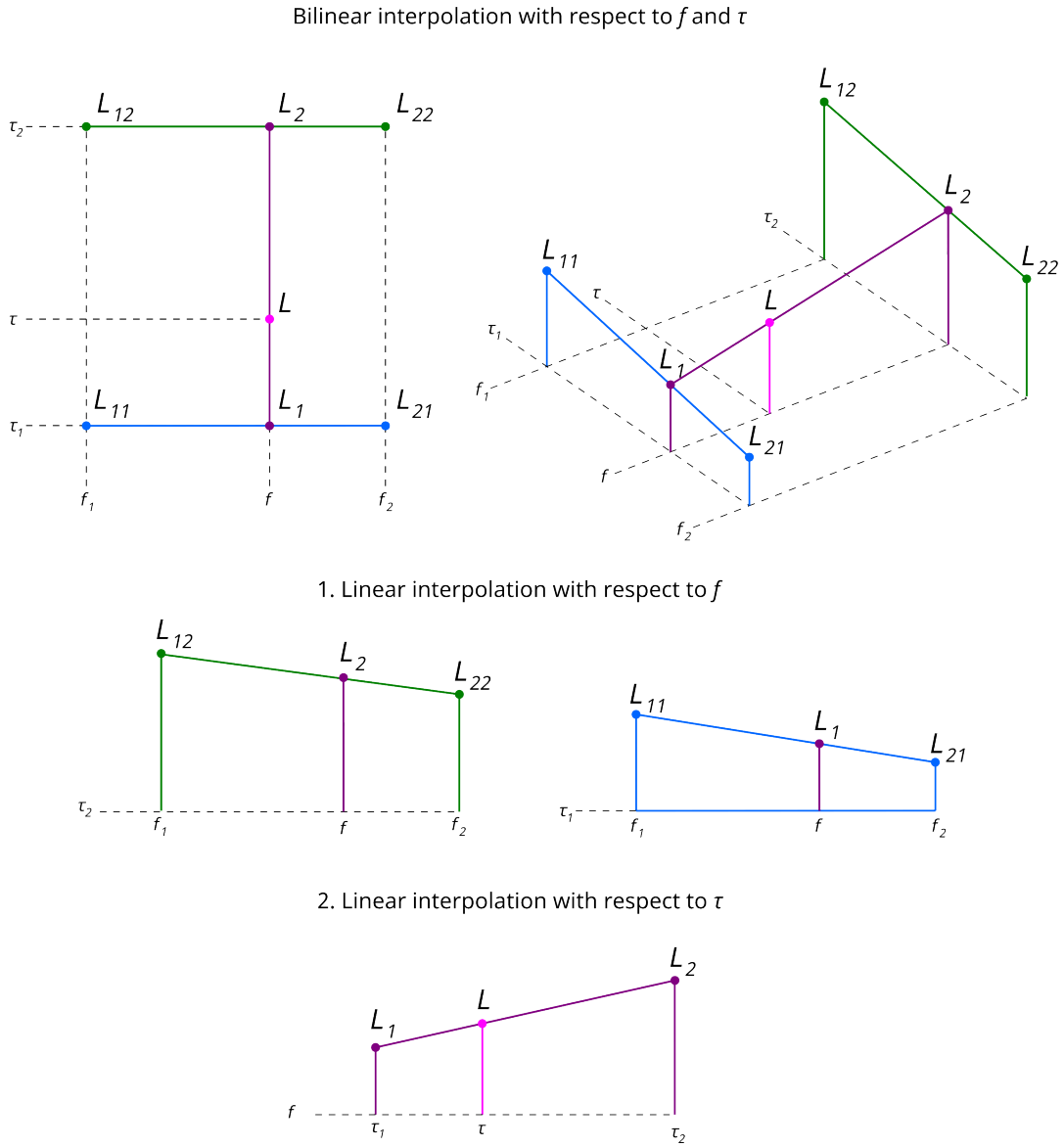


Figure 31: Decomposition of bilinear interpolation into three linear interpolation operations.

Bilinear interpolation is performed with respect to both f and τ coordinates. Effectively, bilinear interpolation is the equivalent to performing two linear interpolation operations and then combining the results on them with a third linear interpolation operation. The principle is illustrated in Figure 31, where L_{11}, L_{12}, L_{21} and L_{22} represent the losses at estimation points surrounding the query point L. The coordinates of the

estimation points are coordinates (f_1,τ_1), (f_1,τ_2), ($f2$,τ_1) and (f_2,τ_2), respectively. L_1 and L_2 represent the restults of first two linear interpolation steps. L_1 is the result of interpolating between L_{11}, L_{21}. Because these two points share a common τ coordinate of τ_1, the linear interpolation is performed with respect to their f coordinates. Similarly, linear interpolation is performed between the f coordinates of L_{12}, L_{22}, yielding L_2. Finally, a linear interpolation operation is performed between L_1 and L_2 with respect to their y coordinates. This yields result L at coordinates (f,τ) that has been bilinearly interpolated between the four estimation points. The set of equations for bilinear interpolation used by the estimator is:

$$L_1 = \frac{f_2 - f}{f_2 - f_1} L_{12} + \frac{f - f_1}{f_2 - f_1} L_{22} \tag{18}$$

$$L_2 = \frac{f_2 - f}{f_2 - f_1} L_{11} + \frac{f - f_1}{f_2 - f_1} L_{12} \tag{19}$$

$$L = y_1 \frac{\tau_2 - \tau}{\tau_2 - \tau_1} L_1 + \frac{\tau - \tau_1}{\tau_2 - \tau_1} L_2 \tag{20}$$

The estimated losses are separated into multiple components and each estimation point generates an estimate for each of the components. As a result, the process to obtain the interpolation result as described in equations (18), (19) and (20) needs to be performed separately for all of them. The total losses are then calculated as a sum of the interpolation results.

The bilinear interpolation approach generates an estimate that is based on the nearest four points that surround the query point. In principle, it would also be possible to consider other points in the operating space when generating the interpolation results. Such approaches would enable the interpolator to capture trends that happen across the entire operating space, instead of just relying the closest estimation points. As an example, spline interpolation

3.8 Temperature compensation

Because temperature affects losses, the estimator includes a method for temperature compensation. As a result of temperature compensation, losses are adjusted up or down based on temperature value given to the estimator program as an argument. The logic for temperature compensation in a winding is based on an assumption that the loss values contained in the training data set have all been calculated at the same temperature. Then, based on the difference between this training temperature and the temperature passed as an argument, resistance, and current, the difference in losses

can be estimated followingly:

$$R_0 = \frac{P_0}{I^2} \tag{21}$$

$$R = R_0(1 + \alpha_{Cu}(T - T_0)) \tag{22}$$

$$P = i^2 R \tag{23}$$

$$P_{diff} = P - P_0 \tag{24}$$

where R_0 is non-corrected winding resistance, P_0 is non-corrected winding power, i is the winding current, R is temperature-corrected winding resistance, α_{Cu} is the temperature coefficient of copper, T is compensation temperature, T_0 is the dataset's calculation temperature and P_{diff} is the difference in losses at temperature T.

It should be noted that this approach can only be applied for stator's resisitive losses, where the stator current required for the estimate can be estimated using the regression model. Similar estimation is not trivial to implement for other loss components. For rotor's resisitive losses, estimating currents is more challenging because the currents generated in the stator are more complex than the stator current and aren't readily available in the training data. Similarly, temperature compensating rotor's and stator's iron losses also requires a more complicated approach. The estimator implemented in this thesis was primarily intended as a proof-of-concept. Additionally, the validity of results was majorly affected by the small amount of available training data. Therefore, the small accuracy improvement afforded by implementing temperature compensation for all loss components was not deemed to be in the scope of the thesis.

3.9 Data collection

Simulation results were used as a training data for the ML models. Training data for losses at different operating points and for different motor designs was generated by utilising design data from the examined motors and a drive simulation software.

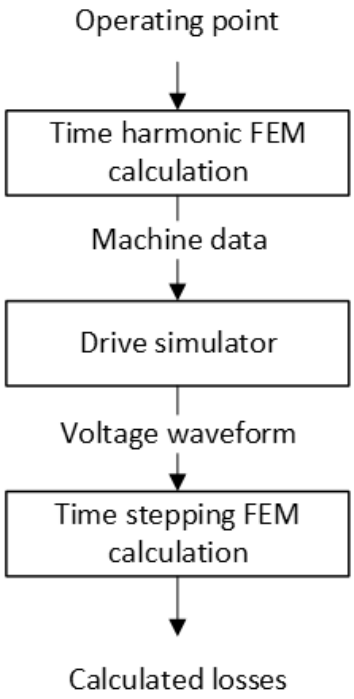


Figure 32: Loss calculation process.

General workflow of data collection is outlined in Figure 32. Overall, the process entails three steps. Firstly, time harmonic FEM calculation is performed in order to obtain motor data. The time harmonic FEM calculation takes as an input the motor's design data, as well as the operating point (speed and torque). This step is relatively fast, taking approximately 10-20 seconds to calculate.

Next, the drive simulator is used to generate voltage waveforms. Drive simulator is a separate program that runs real-time simulation of the drive system including the motor and the converter. The drive simulator requires the motor's principal design data, such as nominal power, speed and voltage. Additionally, motor parameteres such as direct and quadrature axis inductances, efficiency and stator resistance are required. These additional parameters are provided by the time harmonic FEM calculation. As an output, the drive simulator generates phase voltages for each of the motor's phases under MP3C control. The drive simulator runs in real time, which means that the voltage waveforms are relatively quick to generate, taking, in average, 2-5 minutes.

Finally, a second FEM calculation is run. This second calculation uses the same motor design data as the first. However, instead of sinusoidal voltage, the second calculation is run using the MP3C phase voltages generated by the drive simulator. While calculating the motor parameters was possible with relatively fast time harmonic FEM, non-sinusoidal voltage waveforms (such as the MP3C voltage waveform) require time-stepping FEM which is significantly more time-consuming. A time stepping FEM calculation with the parameters required to calculate MP3C losses takes approximately

20 to 30 minutes for one motor at one operating point. The duration of time-stepping calculations was compounded by the fact that the non-sinusoidal supply voltage necessitated running long simulations of 20 supply voltage periods or more in order for the simulation to stabilise as measured by the torque and speed ripple. As a comparison, a simulation with sinusoidal supply voltage typically stabilises in 10 periods or less.

As a result of the time-stepping FEM calculation, stator and rotor loss components are obtained. The loss components studied in this thesis were:

- stator's iron losses
- stator's resistive losses
- rotor's iron losses
- rotor's resistive losses

The losses were divided into elements because it was expected that the components would respond differently to the harmonic frequencies contained by the non-sinusoidal MP3C supply voltage. Thus, in order for the ML models to capture the relationships between the motor parameters, operating point data and losses, the loss components needed to be estimated separately. Additionally, the components were selected specifically to be compatible with other in-house calculation tools including thermal calculation models. This makes it possible for the results given by the estimator to be directly utilised by these tools as an input.

In reality, the data collection process involved more steps than are depicted in Figure 32, but detailed description of these steps has been omitted because they are inconsequential for the results presented in this thesis. However, it should be mentioned that in addition to the time taken by the steps outlined in Figure 32, the data collection process included several steps that required significant manual labour. As an example, the drive simulator does not directly accept the results from time harmonic FEM calculator. Instead, a Matlab script is required to convert the results into drive simulator configuration files. Likewise, the voltage waveforms outputted from the drive simulator need conversion before time-stepping FEM simulations can utilise them. During the course of the thesis, Python scripts were used to automate the process where possible, such as collection of results and conversion of the voltage waveform data. However, a significant amount of manual labour remains in the data collection workflow. This has potential implications for applicability of the method outlined.

During the analysis of the collected data, an anomaly was observed that gave reason to suspect the validity of selected approach to calculate the training data. Namely, the calculated losses for some motors were significantly higher in magnitude when operating at partial torque and speed than at the nominal torque. One possible reason for this could be that the voltage waveform is calculated separately from the actual loss calculation, as outlined in Figure 32. Therefore, no dynamic coupling exists between the frequency converter model and the PM model for which the losses are calculated.

Instead, the drive simulator uses its own PM model. While the PM model in the drive simulator has the same principal parameters such as nominal power, frequency and voltage, as the motor model in the motor design software, it is possible that the two motor models behave differently enough to affect the results. This might cause the losses acquired to not be representative of losses during actual use. This finding requires further investigation before this method can be applied in practice.

4 Results

This chapter presents the results of the thesis. This includes an overview of the developed estimator program, as well as an overview and discussion about the accuracy of the estimation results.

4.1 Overview

During the course of the thesis, a program for estimating MP3C losses was developed. The approach taken relies on pre-calculated MP3C loss results in order to generate an estimate of the losses at a operating point. The estimator requires as an input the motor's design data, which includes values such as the nominal voltage, nominal frequency, nominal torque and stator and rotor design dimensions. These values are readily available from an in-house design software that is used to design permanent magnet synchronous motors. In addition to the motor's design data, requested speed point and requested torque point are given as decimal numbers. The speed and torque are given as percentages of the motor's nominal values.

The estimator was implemented as a command line program. The program is launched and required arguments are given through the command line, or optionally through a script or another program that launches the estimator using desired arguments. This approach was taken both because it is easy to implement and flexible, and because similar approaches are already in use with other programs that interface with the in-house design software. The intention was to make it as easy as possible to integrate the estimator with other design tools by offering an interface that is similar with tools that are already in use.

In addition to interface, the inputs and outputs of the estimator were also specifically designed with emphasis on integration. In practice, this means that the motor data that is input into the program follows the same data item naming convention as the in-house design software that the values originate from. Similarly, outputs from the program use the same data item names in order to make the output easy to understand for a person familiar with the motor design tools.

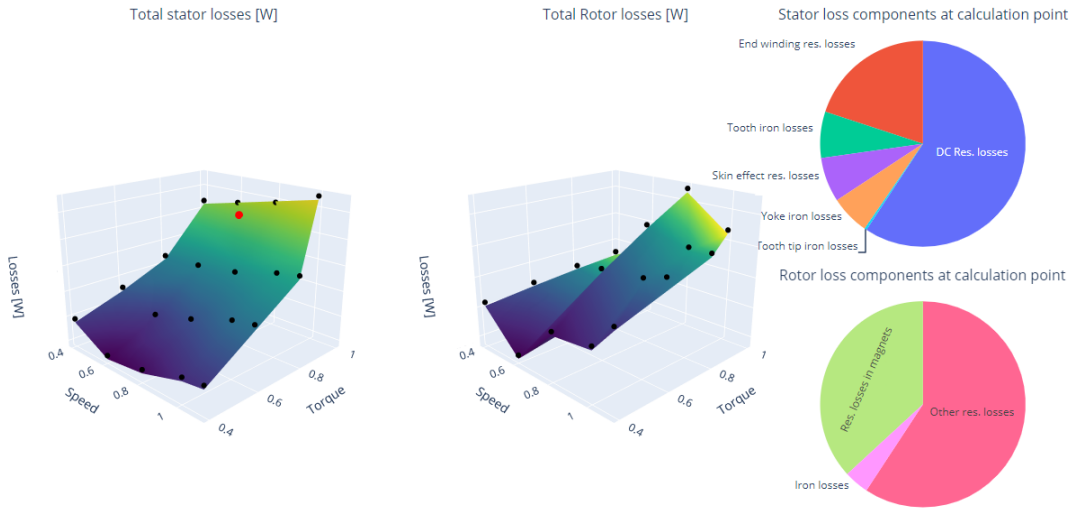


Figure 33: Overview of the result dashboard.

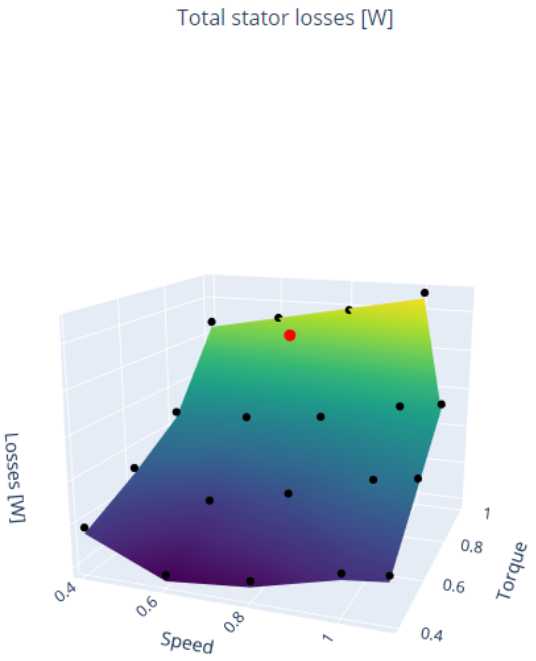


Figure 34: The stator loss surface. Black dots represent estimation points, while the red dot denotes the point at which losses were requested.

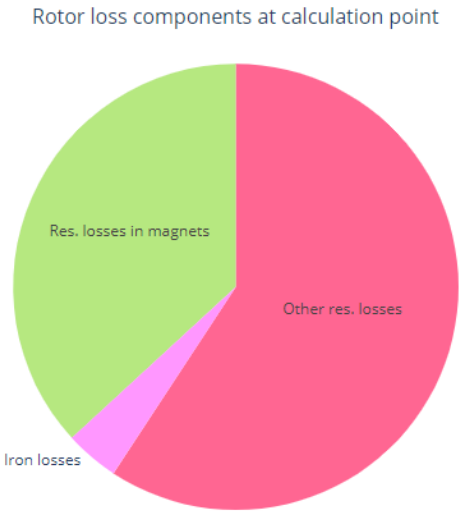


Figure 35: Pie chart displaying rotor loss components at the requested point.

The estimator program generates three output files. The three files are designed to serve the needs of different end-users and use cases. The output files are:

- visual dashboard
- printout
- result data file

The visual dashboard, shown in Figure 33 displays an overview of the results. The dashboard includes two surface plots that visualize the total losses across the entire operating space in addition to two pie charts that display the different loss components of stator and rotor. Figure 34 depicts and example of the loss surface, which plots the total losses as a function of speed and torque. The loss surface is generated from a group of estimation points that are shown as black dots. A red dot denotes the point at which losses were requested. Figure 35 shows and example of rotor loss components at the requested point (the point marked with red dot in Figure 34. The purpose of the visual dashboard is to give a quick overview of how the estimated losses behave at different speed and torque points, as well as visualise the relative magnitudes of different loss components.

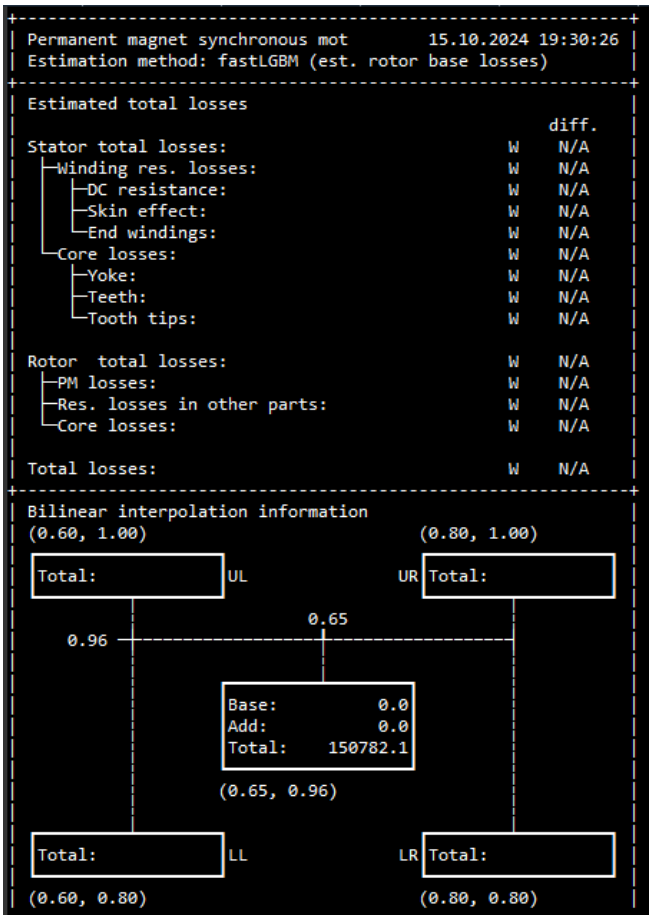


Figure 36: Example of a printout file. Loss magnitudes and references to internal product names have been removed.

Printout is a text file displaying the results in a human-readable form. An example of a printout is shown in Figure 36. The primary purpose of the printout is to provide an easy-to-read and easy-to-implement way of the results. The motor design software already employs this method to display results from various independently-run calculation modules. Thus, a text printout provides a compact and convenient integration pathway for displaying the estimator results. The printout displays the estimated loss components, as well as information about how the losses were interpolated for debug purposes.

In addition to visual dashboard and printout, which are meant to be human-readable, the estimator also generates result data file which simply lists each loss component's name and magnitude. This format is intended for use cases where the loss estimates are used as an input to another program instead of being read by a human. As an example, the result data file can be used to supply a thermal network model, which then, in turn, can generate an estimate of how much the increased losses affect the motor temperature.

4.2 Performance

Validating the performance of the estimator proved challenging because of the limited size of training data set. A typical approach to verifying the performance of a ML model is to split the dataset into two parts: training data and validation data. This way, the models can be trained using the training dataset, and the trained models can be tested by feeding them the validation dataset. The difference between actual values and model-predicted values then provides information about the accuracy of the model. Moreover, comparison between the estimations based on the training data and validation data may reveal problems with the models. Because the dataset only contains ten motors, splitting the dataset drastically reduces the already low amount of data available to train the models.

Because the LightGBM regression model used for the estimator program is well-established and shown to work in many applications, validating the general performance of LightGBM was not emphasised. Instead, steps were taken to evaluate whether LightGBM is suitable for PM loss estimation based on training data. As previously discussed, the number of selected PM motors was limited and training data was cumbersome and time-consuming to generate.

The functionality of the estimator software was tested using the PM training data. The primary reason for these tests was to verify the general functionality of the program, as well as to serve as a 'dummy input' for demonstrating use of the program. For several reasons, these results can not be considered to offer a realistic picture of the performance. As mentioned above, the training data consisted of only ten motors. These ten motors were then calculated at 19 different points, forming the dataset.

In terms of estimation speed, the estimator program meets the requirements for its intended use cases. The process for training and generating estimates for all of the 19 estimation points takes approximately 10-20 seconds with each point having a training dataset of 200 entries. The number of motor design parameters used as features was 23 and the number of estimated loss components was 8. Estimation using pre-trained models is even quicker, taking approximately 5 seconds. The tests were run on a laptop computer with i7-11850H processor and 32 gigabytes of RAM memory.

Initially, the intention was to optimise the program so that estimates are only retrieved from the four adjacent estimation points to the query point, as only they are needed for the estimate. This was omitted because the performance is sufficient even when generating an estimate for all of the points. Additionally, this allows for generating the surface plots shown in Figure 34. If speed increases are required, the program can be refactored to only calculate the required points with minimal effort. This could facilitate, for example, generating estimates for a large mass of motors. For generating estimates for one motor at a time, the current estimation speed is sufficient.

5 Conclusions

This chapter briefly summarises the results of this thesis. Possibilities and prospects for future development in the topic are also discussed.

In this thesis, a program was developed for estimating losses for a medium voltage permanent magnet motor supplied with a frequency converter utilising MP3C. The aim of the thesis was to develop a faster and more streamlined method for estimating MP3C. The current method requires the use of several different programs and takes a significant amount of time, making it unsuitable for many motor design activities.

The primary finding of the thesis was that a loss estimation approach based on machine learning regression models is feasible in principle. However, two significant challenges were identified relating to this approach. Firstly, a limited number of motor designs of the selected type were available. This limited the size of the training data set. Subsequently, the estimation performance was not evaluated for the selected motor type. Secondly, the process for generating the MP3C loss data using the existing methods proved resource-intensive, both in terms of calculation time and manual work required.

Despite efforts undertaken during the thesis work to streamline the process of collecting training data by automation scripts, it remains time-consuming and requires a significant amount of manual work. Even if all manual work is removed through automation, FEM time-stepping calculations still require considerable calculation time. As an example, a set of 274 calculations took 50 hours on a dedicated calculation server. These factors mean that for the loss estimation program proposed here to be feasible, the loss data calculation process should be automated as comprehensively as possible. In addition, an anomaly where the calculated losses were unrealistically high at some operating points was identified. This was suspected to be a result of the process used to generate the loss data. Namely, the lack of dynamic coupling between the drive simulator software which generates the MP3C voltage waveform could cause erroneous results. Further investigation into the phenomena is needed in order to identify the cause.

The estimation system as implemented in this thesis contains all of the required functions to estimate losses based on training data. However, the program was designed from the start to serve as a prototype and test bed for further development. Therefore, it is likely that a complete rewrite of the estimator is needed before the program can be integrated as a part of design software toolchain. This is primarily due to the program being implemented with Python. Python is a language that is well-suited for building prototypes such as the estimator implemented here, because it features simple syntax and a wide selection of machine learning libraries. However, Python is an interpreted language which introduces potential performance challenges. While the performance of the estimator proved sufficient when testing with a small dataset, performance issues could arise when using larger datasets.

In addition to performance differences between Python and compiled low-level languages such as Rust and C, compatibility could also be an issue with Python

program. Each computer running the estimator program needs to have Python and all of the estimator program's dependencies installed. This makes it cumbersome to run on different computers, and introduces possibility of issues if Python or dependencies change over time. While it is possible to create an executable out of Python code by packing the source code, all of the dependencies along and a Python virtual machine together into an executable, the resulting program is very large in size and performs even worse than native Python.

In conclusion, the approach taken to estimate losses using a machine learning model appears feasible, given that sufficient amount of training data is available and the credibility of the training data can be verified. During the thesis, a limited amount of training data was available and thus comprehensive evaluation of the estimation performance was not performed. Additionally, even if more data were available, the method for collecting the training data was found to be time-intensive. In order for a machine learning estimator such as the program presented in this thesis to be feasible, the process for generating the training data for the models needs to be automated further. The estimator program implemented during the thesis is, however, fully functional and can serve as a test bed for future development on the topic. The design of the estimator program is modular and widely configurable. These design choices were made specifically to facilitate further development.

References

ABB (2021), 'New MP3C control method maximizes efficiency and dynamics'. Accessed 31.12.2024.
URL: *https://new.abb.com/news/detail/74747/new-mp3c-control-method-maximizes-efficiency-and-dynamics*

ABB (2023), 'Reaching IE5 efficiency with magnet-free motors'. Accessed 31.12.2024.
URL: *https://resources.news.e.abb.com/attachments/published/112525/lt-LT/3E76C144B942/ABB_EE_WhitePaper_IE5_SynRM_271223.pdf*

Attard, M. (2024), '8 Key Advantages and Disadvantages of Decision Trees'. Accessed 18.06.2024.
URL: *https://insidelearningmachines.com/advantages_and_disadvantages_of_decision_trees/*

Begh, M. and Herzog, H.-G. (2018), 'Comparison of Field Oriented Control and Direct Torque Control'. DOI: 10.13140/RG.2.2.21677.59360/1.

Casadei, D., Profumo, F., Serra, G. and Tani, A. (2002), 'FOC and DTC: two viable schemes for induction motors torque control', *IEEE Transactions on Power Electronics* **17**(5), 779–787.

Chan, T.-F. and Shi, K. (2011), *Applied Intelligent Control of Induction Motor Drives*, John Wiley & Sons (Asia), Singapore.

Finch, J. and Giaouris, D. (2008), 'Controlled AC electrical drives', *IEEE Transactions on Industrial Electronics* **55**(2), 481–491.

Geyer, T. (2017), *Model predictive control of high power converters and industrial drives*, first edition. , John Wiley & Sons, Incorporated, Chichester, West Sussex, United Kingdom.

Geyer, T., Oikonomou, N., Papafotiou, G. and Kieferndorf, F. D. (2012), 'Model Predictive Pulse Pattern Control', *IEEE Transactions on Industry Applications* **48**(2), 663–676.

Hendershot Jr, J. and Miller, T. (2010), *Design of Brushless Permanent-Magnet Machines*, Motor Design Books LLC.

Hitachi Energy (2024), 'Integrated gate-commutated thyristors (IGCT)'. Accessed 08.04.2024.
URL: *https://www.hitachienergy.com/products-and-solutions/semiconductors/integrated-gate-commutated-thyristors-igct*

IEEE Working Group on Power System Harmonics (1983), 'Power System Harmonics: An Overview', *IEEE Transactions on Power Apparatus and Systems* **PAS-102**(8), 2455–2460.

Kazimierczuk, M. K. (2014), *High-frequency magnetic components*, second edition. , Wiley Blackwell, Chichester, West Sussex.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q. and Liu, T.-Y. (2017), LightGBM: a highly efficient gradient boosting decision tree, *in* 'Proceedings of the 31st International Conference on Neural Information Processing Systems', NIPS'17, Curran Associates Inc., Red Hook, NY, USA, pp. 3149–3157.

Masoum, M. A. S. (2015), *Power quality in power systems and electrical machines*, second edition. , Academic Press, an imprint of Elsevier, London.

Pyrhönen, J., Jokinen, T. and Hrabovcova, V. (2014), *Design of rotating electrical machines*, second edition. , Wiley, Chichester, West Sussex, United Kingdom.

Sen, P. C. (2013), *Principles of electric machines and power electronics*, third edition. , John Wiley & Sons, Inc, Hoboken, New Jersey, United States.

Slemon, G. R. (1992), *Electric Machines and Drives*, Addison-Wesley Publishing Company, Inc.

Wu, B. B. (2006), *High-power converters and AC drives*, Wiley, Hoboken, N.J.

Zhou, Z.-H. (2021), *Machine learning*, Springer, Gateway East, Singapore.

Zurek, S. (2023*a*), 'Encyclopedia Magnetica'. Accessed 17.06.2024.
URL: *https://www.e-magnetica.pl/doku.php/file/skin_effect_illustration_magnetica_png*

Zurek, S. (2023*b*), 'Encyclopedia Magnetica'. Accessed 17.06.2024.
URL: *https://www.e-magnetica.pl/doku.php/file/proximity_effect_proximity_loss_magnetica_png*