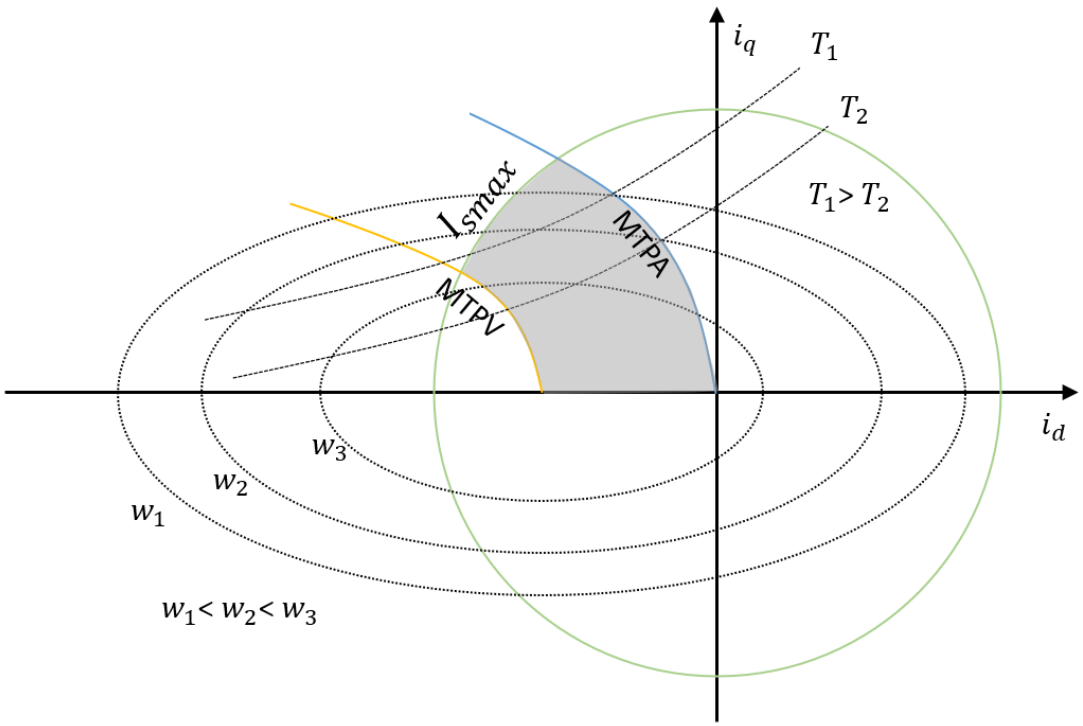
CHALMERS
UNIVERSITY OF TECHNOLOGY



Comparing numerical and machine learning algorithms for optimized operation points of an electrical machine

Master's thesis in Electrical Engineering

YIJIE REN

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2023
www.chalmers.se

Master's thesis 2023

Comparing numerical and machine learning algorithms for optimized operation points of an electrical machine

YIJIE REN

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden 2023

Comparing numerical and machine learning algorithms for optimized operation points of an electrical machine
YIJIE REN

Supervisor: Joachim Härsjö, Volvo Cars
Supervisor: Raik Orbay, Volvo Cars
Examiner: Martin Fabian, Chalmers University of Technology

Master's Thesis 2023
Department of Electrical Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Operating region of IPMSM

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2023

Comparing numerical and machine learning algorithms for optimized operation points of an electrical machine

YIJIE REN
Department of Electrical Engineering
Chalmers University of Technology

Abstract

This work compares the Lookup Table (LuT) based numerical method with neural network (NN) based, and reinforcement learning (RL) based methods, for finding the optimal operating point of an Interior Permanent Magnet Synchronous Machine. Commonly, numerical methods are used to search for Maximum Torque Per Ampere (MTPA) points, which, although relatively accurate, often require significant computation time and generate large amounts of output data to obtain precise operating points. In this thesis, a simple approach was employed to establish a three-dimensional LuT based on nonlinear data, which is used as a baseline for comparing machine learning models. By comparing multiple metrics, it was verified that the presented NN-based method can quickly, efficiently, and accurately fit the LuT data, making it suitable for data reduction and addressing the issue of large output data of LuT. The RL-based method offers a simple model that is not dependent on data and can essentially achieve MTPA control, providing new inspiration for finding operating points. Finally, based on the comparative results, the advantages and challenges of the proposed different models are presented.

Keywords: Maximum Torque Per Ampere (MTPA), Interior Permanent Magnet Synchronous Machine (IPMSM), neural network, reinforcement learning, SARSA

Acknowledgements

I would like to express my sincere gratitude to Department of Propulsion Innovation Lab at Volvo Car Corporation, Gothenburg, Sweden for providing me with a wonderful learning platform that facilitated the completion of my thesis.
I am deeply thankful to my examiner Full Prof. Martin Fabian for his continuous support and guidance. His assistance and encouragement have played a crucial role in shaping the direction of my thesis. I would also like to extend my appreciation to my industrial supervisor Dr. Raik Orbay and Dr. Joachim Härsjö for their invaluable advice and meticulous guidance in refining the finer details of my work.
I am grateful to my colleague Łukasz Sobieraj and friend Justin Cheung for their insightful discussions and inspirational ideas.
Lastly, but not least, I would like to acknowledge the support of my parents and my friend Shuying, whose unwavering encouragement and belief in my abilities have been a constant source of motivation.

Yijie Ren, Gothenburg, 05 2023

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

back EMF	Back Electromotive Force
BEV	Battery Electric Vehicle
CO_2e	Carbon Dioxide Equivalent
DDPG	Deep Deterministic Policy Gradient
IPMSM	Interior Permanent Magnet Synchronous Machine
kWh	kilowatt-hour
ML	Machine Learning
MLPRegressor	Multi-layer Perceptron Regressor
MSE	Mean Squared Error
MTPA	Maximum Torque Per Ampere
MTPV	Maximum Torque Per Voltage
NN	neural network
PHEV	Plug-in Hybrid Electric Vehicles
PMSM	Permanent Magnet Synchronous Machine
ReLU	Rectified Linear Unit
RL	Reinforcement Learning
SARSA	State-Action-Reward-State-Action
SPMSM	Surface-mounted Permanent Magnet Synchronous Machine
SVM	Support Vector Machine
TD learning	Temporal Difference learning
TDP	Thermal Design Power

Nomenclature

Below is the nomenclature of indices, parameters, and variables that have been used throughout this thesis.

Indices

d, q	Indices for direct-quadrature system
i, j	Indices for NN neurons in different layer
k	Index for NN layer
t	Index for the learning step

Parameters

p	Number of pole pairs
R_s	Stator resistance
ψ_m	Permanent magnetic flux linkage
L_d, L_q	d-/q-axis inductance
U_{dc}	DC-bus voltage
I_{smax}	Maximum stator current
w_{max}	Maximum speed
T_{max}	Maximum torque
ϵ	Learning rate in ML
α	Learning rate in RL
γ	Discount factor

Variables

f	Voltage, current, or flux linkage
u_d, u_q	Stator voltage in d-/q-axis
i_d, i_q	Stator current in d-/q-axis
ψ_d, ψ_q	Stator flux linkage in d-/q-axis
w_e	Electrical angular velocity
I	Initial point of fmincon
U_{smax}	Maximum voltage
U_{smin}	Minimum voltage
x	Independent features
y	Dependent features
h_i^{k+1}	Output of a neuron i in the layer $k+1$
b_{k+1}	Bias of the neurons in layer $k+1$
$w_{i,j}^k$	Weight that connects the neuron j in layer k to the neuron i in layer $k+1$
J	Loss
S	Current state
S'	Next state
A	Current action
A'	Next action
R	Reward
T_{ref}	Reference torque
w_{ref}	Reference speed
u_{ref}	Reference voltage

Contents

List of Figures

List of Tables

0

About Volvo Car Corporation

Since the introduction of the first VOLVO car in 1927, Volvo Car Corporation has evolved into a globally recognized brand with a rich heritage. Over the years, VOLVO has strived for innovation and maintained values that align with societal trends, making itself become one of the fastest-growing luxury brands. With its global headquarters in Gothenburg, Sweden, VOLVO has also established its Americas headquarters in Mahwah, New Jersey, and its APAC head office in Shanghai. The company operates multiple production facilities in countries such as Sweden, China, and Belgium, enabling VOLVO cars to be sold in over 100 countries worldwide.

Throughout its history, Volvo Car Corporation has earned a reputation for producing vehicles that prioritize occupant safety while maintaining performance and style. The company's unwavering commitment to safety has resulted in numerous groundbreaking innovations and pioneering technologies that have set industry standards.

In 1959, VOLVO engineer Nils Bohlin introduced the three-point safety belt during mass production of the PV 544 [1]. To benefit as many people as possible, VOLVO chose to forgo patent rights. To date, it is estimated that over a million lives have been saved as a result. Over the decades, VOLVO has continuously introduced new automotive safety technologies. The latest VOLVO EX90 exemplifies the highest safety standards in VOLVO's history. In addition to the fundamental safety features found in VOLVO vehicles, such as *colllision avoidance* and *lane keeping aid*, EX90 is equipped with advanced functionalities like the *driver understanding system* and *occupant sensing*. These intelligent systems aim to provide enhanced protection for occupants, aligning with VOLVO's vision of achieving zero collisions.

In the realm of design, Volvo Car Company embraces the concept of Scandinavian minimalism, seamlessly merging elegant aesthetics with functional practicality. This sets VOLVO apart from other luxury car manufacturers, allowing it to stand out from the crowd.

Figure 0.1: VOLVO EX90

Volvo Car Company's commitment to sustainability is another prominent aspect of its corporate philosophy. Recognizing the importance of reducing its environmental impact, the company has made significant strides in developing and implementing eco-friendly technologies. VOLVO aims to achieve full electrification by 2030. Today, every VOLVO car has an electrified version, and by 2030, the company plans to transform into a fully electric car company. In line with its commitment to sustainability, VOLVO has set ambitious targets approved by the Science Based Targets initiative. The company aims to become a climate-neutral organization by 2040, further demonstrating its dedication to addressing climate change and contributing to a greener future.
This thesis project aligns with VOLVO's commitment to achieving full electrification goals. The thesis focuses on comparing various methods for finding the optimal operating point of electric machines to achieve more efficient motor control.

1
Introduction

1.1 Background

In the past few decades, countries worldwide have been committed to achieving carbon neutrality due to the rising global emission levels and the imperative to meet sustainable development goals. Among them, Sweden has emerged as a leader in the production and utilization of alternative fuels and renewable energy. In the new Climate Act, Sweden has pledged to reduce carbon dioxide emissions by 63% (relative to 1990 levels) by 2030 and achieve carbon neutrality by 2045. The transportation sector plays a crucial role in influencing carbon emissions. Sweden aims to decrease domestic transportation emissions by 70% compared to 2010 levels by 2030. Through incentives such as national grants and tax subsidies, plug-in electric vehicles have experienced robust growth in the Swedish automotive market. In 2022, the market share of plug-in electric vehicles reached its all-time high at 74.6%, comprising 51.3% for battery electric vehicles (BEV) and 23.3% for plug-in hybrid electric vehicles (PHEV). By comparison, the respective proportions in 2021 were 36.4% and 24.3% [2]. Sweden has made significant progress in the electrification of automobiles.

The propulsion system is the core of electric vehicles. Permanent magnet synchronous machine (PMSM) has been widely used due to its advantages of high efficiency, high power density, compact structure, and fast dynamic response [3]. Therefore, exploring methods for achieving more efficient control of PMSM holds considerable research significance.

1.2 Motivation

This project has been performed at Propulsion Innovation Lab at Volvo Car Company. For VOLVO, the future is all about electric cars. Since the cars are using batteries, the amount of energy in the car is limited and it is therefore important to also include energy efficiency in the optimization of the electrical machine's operation points. A common way to optimize the operating points is to use a numerical method to find what is called maximum torque per ampere (MTPA) where the requested torque is achieved using the minimum amount of current while still considering several constraints. Although this method is accurate, the computational time required as well as the resulting output data quickly grows as more constraints are included.

1.3 Scope

In this thesis, we are interested in the MTPA control of an interior permanent magnet synchronous machine (IPMSM). The main objective is to compare numerical and machine learning (ML) methods to see which method is more suitable for MTPA control of IPMSM, if any of these approaches uses the least amount of computational and storage resources, and which method is more environmentally friendly. The numerical method here is the lookup table (LuT) based method. For the specific machine learning methods, this thesis compares the neural network algorithm using MLPRegressor and the SARSA-based reinforcement learning algorithm. The detailed content is presented in chapters 3 and 4.

1.4 Thesis outline

Chapter 2 mainly analyzes the mathematical model of IPMSM, defines the optimal operating boundary based on this model, and elaborates on the MTPA control and field-weakening control strategies. The methods for finding operating points are then presented, which provide the theoretical basis for the algorithm proposed in this thesis.
In Chapter 3, two numerical methods were proposed: the basic LuT-based method based on a linear model and the extended LuT-based method based on a nonlinear model.
Chapter 4 establishes and compares a neural network based supervised learning algorithm and a SARSA-based reinforcement learning algorithm for IPMSM control.
The Results and Conclusion sections are presented in Chapter 5, and Chapter 6, respectively.

2

Theoretical Background

This chapter presents the mathematical model for describing the IPMSM, including the corresponding voltage, flux linkage, and torque equations. Based on this, the MTPA and the field-weakening control strategy are introduced, and the optimal operating boundary is defined. Then, the methods to find operating points are presented, which provide the theoretical basis for the algorithm proposed in this article.

2.1 IPMSM model

PMSMs can be divided into two types according to their rotor structure: surface-mounted (SPMSM) and interior (IPMSM). IPMSM has a saliency effect, so their torque output capability is generally greater than that of SPMSM and therefore has a wider range of applications. According to this, the algorithm studied in this thesis is only for the IPMSM. The IPMSM has an unequal quadrature and direct axis inductance, so the mathematical model is more complex compared to SPMSM and the control algorithm is correspondingly complex.

For IPMSM, in order to simplify the study, the two-axis theoretical analysis method is often used. The three-phase current and voltage are transformed into current and voltage in a synchronous rotating d-q coordinate system by means of the Park transform [5]. The d-axis of the system is aligned with the rotor's magnetic field whereas the q-axis is orthogonal to the d-axis. After the Park transformation, the task of controlling three-phase AC power can be converted into controlling DC power.

The transformation equation is shown in 2.1. Where f_d and f_q represent the voltage, current, or flux linkage in the d-q axes. f_a, f_b, and f_c represent the voltage, current, or flux linkage in the three-phase stationary coordinate system. α represents the angle between the d-axis and the a-axis as seen in the Fig. 2.1.

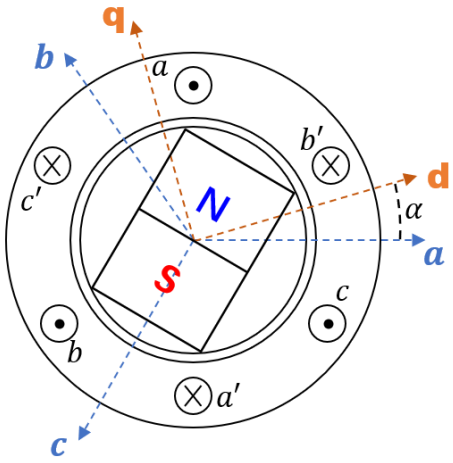


Figure 2.1: Schematic representation of an IPMSM

$$\begin{bmatrix} f_d \\ f_q \end{bmatrix} = \frac{2}{3} \begin{bmatrix} \cos\alpha & \cos(\alpha - \frac{2}{3}\pi) & \cos(\alpha + \frac{2}{3}\pi) \\ -\sin\alpha & -\sin(\alpha - \frac{2}{3}\pi) & -\sin(\alpha + \frac{2}{3}\pi) \end{bmatrix} \begin{bmatrix} f_a \\ f_b \\ f_c \end{bmatrix} \tag{2.1}$$

The stator voltage equation in the d-q coordinate system can be given as:

$$\begin{aligned} u_d &= R_s i_d + L_d \frac{\mathrm{d}i_d}{\mathrm{d}t} - w_e \psi_q \\ u_q &= R_s i_q + L_q \frac{\mathrm{d}i_q}{\mathrm{d}t} + w_e \psi_d \end{aligned} \tag{2.2}$$

where u_d, u_q are stator voltage, i_d, i_q are stator current, L_d, L_q are inductances, and ψ_d, ψ_q are stator flux linkage, each in the respective d and q axes. R_s is stator resistance, and w_e is electrical angular velocity, which is equal to the mechanical angular velocity multiplied by the number of pole pairs.
The flux linkages ψ_d, ψ_q are formulated as the following equations:

$$\begin{aligned} \psi_d &= L_d i_d + \psi_m \\ \psi_q &= L_q i_q \end{aligned} \tag{2.3}$$

where ψ_m is the permanent magnetic flux linkage.
The torque can be described as:

$$T = \frac{3p}{2}[\psi_m + (L_d - L_q) i_d] i_q \tag{2.4}$$

where T is torque, and p is the number of pole pairs.

2.2 Control strategy

When the machine speed is below the maximum speed that can be achieved at rated torque without exceeding the maximum permissible inverter voltage (base

speed) [4], use the MTPA strategy. Any point on the constant torque curve corresponds to a pair of i_d and i_q. Expression 2.4 reveals more clearly that there are countless stator current vectors synthesized by i_d and i_q that can generate the same torque magnitude. Among them, there exists a stator current vector whose magnitude is minimum while producing the desired torque. The combination of the corresponding i_d and i_q of this current vector is referred to as the MTPA point. A more general definition can be understood as the point where the maximum torque can be obtained under a given current magnitude. Utilizing the MTPA strategy can significantly enhance the efficiency of the machine.

When the speed rises beyond the base speed, the power supply voltage reaches its maximum value. Since the back electromotive force (back EMF) of a machine increases proportionally with the machine's speed, at this point, the back EMF exceeds the power supply voltage, thereby preventing further increases in machine speed. Field weakening enables higher speed by reducing the back EMF generated by the machine. The d-axis of the stator current i_d is in line with the axis of the rotor's magnetic flux. A negative d-axis current reduces the amount of the rotor's flux. Since back EMF is created by rotor flux, adding negative i_d reduces back EMF. This allows a larger portion of the bus voltage to be used for increasing the machine speed.

2.3 Optimal operation boundary

The optimal operating locus is the region bounded by the MTPA curve, maximum torque per voltage (MTPV) curve, and current limit circle as depicted in Fig. 2.2. The gray area in the figure represents the field-weakening region. Under specific reference torque and speed, the optimal operating point can be found from this optimal operation boundary or the field-weakening region. When the machine speed is below the base speed, the optimal operating point coincides with the MTPA curve. As the speed increases, the voltage limit ellipse gradually shrinks. When the speed is greater than the base speed, a part of the voltage limit ellipse is located in the gray area, and the machine enters the field-weakening region where the flux from the permanent magnet should be lowed by i_d. When the torque is small (T_2 in the Fig.), the intersection point of the constant torque curve and the constant speed curve (w_2 in the Fig.) is to the right of the MTPA curve. At this time, the optimal operating point is the intersection point of the constant torque curve and the MTPA curve, which is point A in the Fig.. When the torque is large (T_1 in the Fig.), the intersection point of the constant torque curve and the constant speed curve(w_2 in the Fig.) is in the gray area. At this time, the intersection point is the optimal operating point, which is point B in the Fig.. When both the speed and torque are relatively large (T_1 and w_3), there is no intersection point, which means that the torque cannot be achieved at this speed. The optimal operating point that can achieve the maximum torque at this speed is the intersection point of the constant speed curve and the MTPV curve, which is point C in the Fig..

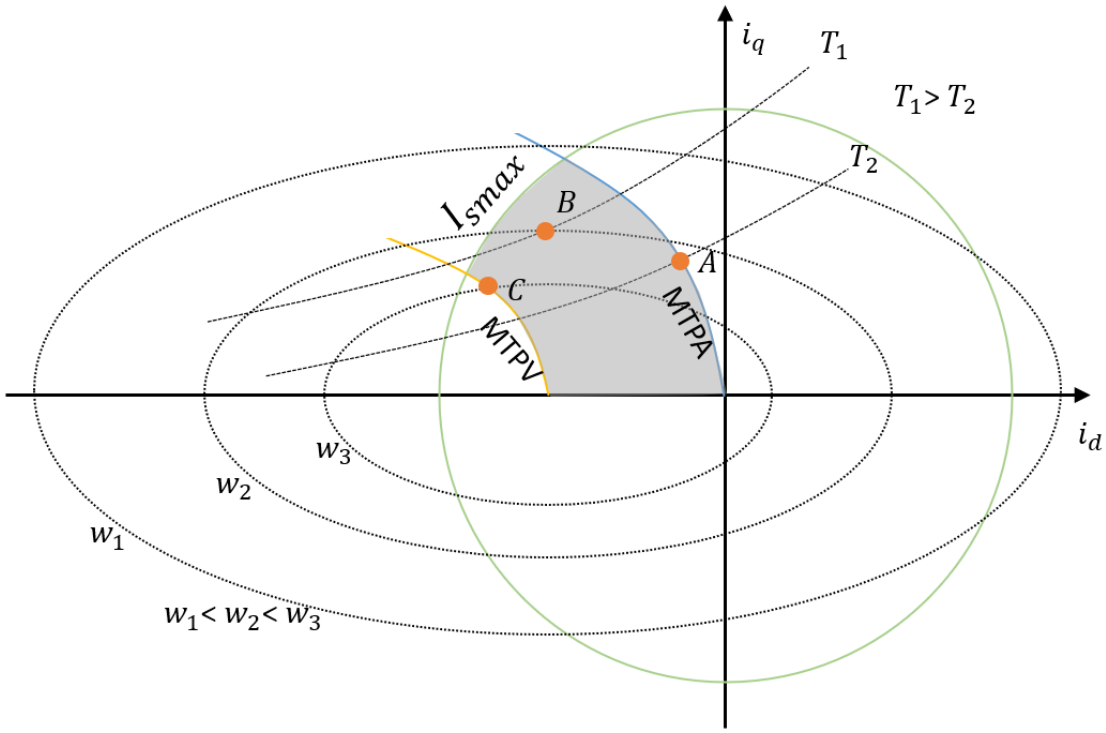


Figure 2.2: Operating region of IPMSM, where w_j corresponds to the speed, T_k corresponds to the torque, I_{smax} corresponds to the current limit.

2.3.1 Maximum torque per ampere

The MTPA strategy can be a current control method for the IPMSM, which is only applicable to states where the speed is below the base speed. The MTPA curve does not belong to the field-weakening region, but it is the boundary of the field-weakening region. The curve is composed of the intersection points of the constant torque curve and the current circle at different current levels. This means that for a given constant torque curve, the minimum current required to achieve that torque is located on the MTPA curve. From 2.4, the relation between i_d and i_q for the MTPA control is derived as [6] :

$$i_d = \frac{\psi_m}{2(L_q - L_d)} - \sqrt{\frac{\psi_m^2}{4(L_q - L_d)^2} + i_q^2}. \tag{2.5}$$

While it is feasible to use this expression in combination with the torque equation to ascertain the minimal value of i_d and i_q necessary for MTPA control in a given torque scenario, the intricacy of the formula and the considerable computational effort involved render it unfeasible for real-world applications. Commonly used alternative methods are described in detail in Section 2.4.

2.3.2 Current limit circle

The maximum amplitude of the current vector synthesized by i_d and i_q is determined by both the power source and the machine itself. The operating points of the motor

are all within the current limit circle with a radius of the maximum current. The formula for the current limit circle is as follows:

$$i_d^2 + i_q^2 = I_{smax}^2 \tag{2.6}$$

where I_{smax} is the maximum stator current capability of the system.

2.3.3 Voltage limit ellipse

The voltage limit ellipse is a series of ellipses obtained when the stator voltage vector reaches its maximum value. When the electrical machine runs at high speed under steady operation, the current transients can be neglected, and $R_s i_d \ll w_e \psi_q$, $R_s i_q \ll w_e \psi_d$ [11], the voltage balance equation can be described as follows:

$$\begin{aligned} u_d &= -w_e \psi_q \\ u_q &= w_e \psi_d \end{aligned} \tag{2.7}$$

Combining (2.3) with (2.7), obtains an expression for the voltage limit ellipse:

$$\begin{aligned} u_d^2 + u_q^2 &= U_{smax}^2 \\ (L_d i_d + \psi_m)^2 + (L_q i_q)^2 &= (\frac{U_{smax}}{w_e})^2 \end{aligned} \tag{2.8}$$

where $U_{smax} = U_{dc}/\sqrt{3}$, U_{smax} is the maximum voltage, U_{dc} is the DC-bus voltage. As the speed increases, the voltage limit ellipse gradually shrinks, and when it reaches the rated maximum speed, the voltage limit ellipse is at its minimum. These ellipses are also known as constant speed curves, with their center point at $(-\frac{\psi_m}{L_d}, 0)$.

2.3.4 Maximum torque per voltage

For each constant speed curve, the torque value of each steady-state operating point can be calculated. The point of tangency between the voltage limit ellipse and the constant torque curve tangent to it provides the maximum torque that can be output at a given speed with the maximum voltage. If the current operating point is situated on the MTPV curve, it signifies that the U_{dc} has been completely utilized. As the speed increases, the current operating point shifts only towards the center of the voltage limit ellipse along the MTPV curve, resulting in a decrease in the torque output of the machine. From [7], MTPV line is derived as:

$$i_d = -\frac{\psi_m}{L_d} + \frac{-L_d \psi_m + \sqrt{(L_q \psi_m)^2 + 4L_q^2 (L_d - L_q)^2 i_q^2}}{2L_d (L_d - L_q)} \tag{2.9}$$

2.4 Methods to find operating points

2.4.1 Numerical methods

The traditional method involves taking the derivative of the torque equation and solving for the stator current components in the *d-q* coordinate system when the

derivative is equal to zero. Given the torque command and the known motor model and parameters, the expression can be directly used to determine the value of the stator current component. However, this approach is computationally complex and since CPU power in an embedded system used in automotive is limited, it can be a significant burden on the system. Moreover, while this algorithm is accurate when the motor parameters are constant, in practical applications, the parameters can change due to factors such as temperature and magnetic saturation. In [8], the Lagrangian multiplier function is used to minimize the current in the torque equation. This method calculates the MTPA operating point using constant parameters such as the inductance and flux linkage. While the use of constant parameter IPMSM models is attractive due to their ease of implementation, the presence of magnetic saturation and cross-coupling effects can cause significant changes in motor parameters during operation, especially for the IPMSM. Under high-load conditions, simply using a linear IPMSM model can lead to higher copper losses and reduced control performance.

Another more commonly used numerical method is to establish a LuT for IPMSM. In [9], the authors present a convenient and straightforward approach for establishing a torque LuT for non-linear IPMSM. They obtain the operating point matrix $[i_d, i_q, \lambda_d, \lambda_q, T_{calc}]$ from IPM profiling tests and then use Matlab to calculate the optimal operating boundary that follows MTPA control and field-weakening control. It is worth noting that since the original data comes from experimental measurements, the motor magnetic circuit is non-linear. Moreover, the logic in the written Matlab code only locates the point on the target function curve closest to the origin (there may actually be multiple points), so the directly calculated MTPA and MTPV curves are not smooth. The authors also performed a 3-degree polynomial curve fitting to achieve a smoother optimal operating boundary. Although the method in [9] has been verified to be feasible, torque discrepancies may occur in the high-speed low-torque region, which is believed to be caused by both core loss and position sensor accuracy issue at high speed. And the LuT is two-dimensional, only supporting current lookup based on torque and speed without considering varying DC-bus voltage or more dimensions. When more dimensions are introduced, the amount of computational time and storage space required can increase significantly. Since the LuT-based method is based on the electrical machine mathematical model's MTPA control scheme, it has a strong dependence on the model and parameters. In practical applications, the stator resistance and permanent magnet flux will change with the motor temperature and the d-q axis inductance will change with the air gap magnetic flux saturation. Therefore, this nonlinear behavior needs to be considered in the process of finding optimal operating points.

2.4.2 Machine learning

ML-based IPMSM control typically does not require a comprehensive understanding of the motor model and parameters and can simulate the nonlinear characteristics of the system without the need for precise calculations and the huge storage space required by numerical methods. Neural network based algorithms for IPMSM control have been proven to be efficient due to their easy implementation and effective

control. In [16] is presented a simple neural network structure that allows for fast and accurate generation of reference currents and an efficient neural network training mechanism is proposed based on the neural network back-propagation (LMBP) algorithm. Compared to traditional neural network based control methods [17, 18], [16] overcomes the challenge of achieving a smooth transition between MTPA and field-weakening operation. The network model proposed in [16] covers both MTPA and field-weakening control, allowing for stable operation at high speeds and effective control of the IPMSM torque over the entire speed range. Offline training improves the robustness of the neural network under noise.
Despite the numerous advantages of the Neural network based model, it relies heavily on large training datasets to learn how to provide accurate reference currents due to the characteristics of supervised learning. To alleviate the issue of the dataset, [19] proposes a deep reinforcement learning based parameter-independent speed and current control method for IPMSM. This article uses the deep deterministic policy gradient (DDPG) algorithm, and the control methodology is based on continuous improvement through measurement feedback for an optimal control solution. The established model has strong adaptive capabilities and can be widely applied to other complex nonlinear systems. However, this reinforcement learning model is relatively complex and challenging to implement, and a long training time is required to achieve usable control performance.

3

Numerical Methods

There are many numerical methods for the current control of IPMSM, each with its own advantages and disadvantages. LuT-based methods obtain the optimal d-q-axis current reference values corresponding to different operating conditions and stores the data in the controller memory. During IPMSM operation, the optimal reference values are accurately indexed based on different operating conditions. The LuT-based methods avoid online calculations, reduce hardware burden, and have been widely used in engineering. In order to balance the complexity of algorithm implementation, response speed, computation, and robustness, this thesis selects the LuT-based method, which is commonly used in industry, as the comparative object.

3.1 Basic LuT-based method

While a nonlinear model provides more accurate results, it also increases the computational load and the difficulty of coding the control algorithm. Using a linear IPMSM model can simplify the calculation process and greatly reduce the computing load. This section is based on an IPMSM with constant parameters and uses the basic LuT-based method to find the optimal operating points. Although sacrificing some torque accuracy, this method quickly establishes a usable algorithm, paving the way for more complex algorithms for nonlinear IPMSM models in later sections.

3.1.1 Parameters of the linear IPMSM

Table 3.1: Parameters of the linear IPMSM

Parameter	Symbol	Value
Number of pole pairs	p	4
Stator resistance	R_s	0.05 Ω
Permanent magnetic flux linkage	ψ_m	0.1 Wb
d-axis inductance	L_d	0.379 mH
q-axis inductance	L_q	0.766 mH
DC-bus voltage	U_{dc}	400 V
Maximum stator current	I_{smax}	500 A
Maximum speed	w_{max}	12000 rpm
Maximum torque	T_{max}	400 Nm

The linear IPMSM parameters used in this thesis are shown in the table above. Linear means that the *d*-*q*-axis inductance and permanent magnetic flux linkage are constant values and are not affected by motor temperature or air gap magnetic flux saturation, and do not vary with current.

3.1.2 Finding optimal operation points

The use of the LuT-based method for MTPA control removes stress from the embedded computer used in the vehicle and is relatively easy to implement. In traditional methods, the optimal operating point is found using a scanning method, where two loops are used to scan the stator current I_s from 0 to I_{smax} and the current angle β (the angle between stator current vector I_s and d-axis) from 90° to 180°. The formula for torque T as a function of I_s and β is then used to compute the maximum torque point. Although this method is widely used in engineering, it cannot compute the optimal operating points in the MTPV and field-weakening regions, for which other methods need to be employed. Alternatively, more comprehensive and accurate algorithms, as mentioned in Chapter 2, [9], can be used but still require calculations to be performed in different regions and are therefore less convenient.
This thesis uses an algorithm that balances full-region control and computational efficiency, enabling the computation of operating points for both MTPA and field-weakening regions in one single run. The core idea of the algorithm is to use the *fmincon* function in MATLAB to perform computations for each value within two loops established from 0 to w_{max}, and 0 to T_{max}. *fmincon* can be used to find the minimum of a constrained nonlinear multivariable function. It needs an objective function and some constraints defined by equalities, inequalities, and bounds.

The pseudo-code is as follows:

```
% Generate w_ref and T_ref
w_ref = linspace(0,w_max,30); T_ref = linspace(0,T_max,30)

% Run main function
for i=1:length(w_ref)
    for j=1:length(T_ref)
        LuT_temp = LuT(w_ref(i), T_ref(j), R, Ld,...
                       Lq, psi_m, I_smax, p, u_max)
        id_ref(i,j) = LuT_temp(1)
        iq_ref(i,j) = LuT_temp(2)
    end
end

% Function - LuT
function LuT(w_ref, T_ref, R, Ld, Lq, psi_m, ...
             I_smax, p, V_max)
    A = []; b = []; Aeq = []; beq = []; lb = []; ub = []
    I0 = [0 0]
    I = fmincon(objfun_LuT, I0, A, b, Aeq, beq, lb, ub,...
                nonlcon_LuT)
    id_ref = I(1)
    iq_ref = I(2)
    LuT = [id_ref; iq_ref]
end function
```

Here the objective function (objfun_LuT in the pseudo-code) is defined as:

$$(T - T_{ref})^2 - (\frac{T_{ref}}{norm(I)})^2 \tag{3.1}$$

where I is the initial point of *fmincon*, used to determine the variables that *fmincon* accepts, specified as $[i_d\ i_q]$. T is from the torque equation 2.4, calculated by using the value of the above $[i_d\ i_q]$ and the machine parameters. T_{ref} is the given reference torque, it is the incremental value in the inner loop. The incremental step size is determined by the resolution needed for the LuT.

The first term $(T - T_{ref})^2$ in the objective function is used to ensure that the actual torque, T, obtained from the independent variable I in the *fmincon* function, converges to the given reference torque, T_{ref}. The second term, $(\frac{T_{ref}}{norm(I)})^2$, is used to ensure the control strategy. The two terms are connected by a negative sign. The *fmincon* function is used to find the minimum value, which means that the first term in the objective function is as small as possible at the minimum value, while the ratio in the second term is as large as possible. As the first term is squared, its minimum value is zero. The closer the first term is to zero, the closer the actual torque, T, is to the reference torque, T_{ref}. The larger the ratio in the second term, the closer it is to the optimal operating point.

The nonlinear constraints (nonlcon_LuT in the pseudo-code) consist of several inequalities:

$$\begin{aligned} u_d^2 + u_q^2 - U_{smax}^2 &\leq 0 \\ i_d^2 + i_q^2 - I_{smax}^2 &\leq 0 \\ i_d &\leq 0 \\ \frac{T}{T_{ref}} - 1 &\leq 0 \end{aligned} \tag{3.2}$$

These inequalities together form the optimal operation boundary and ensure the convergence of the objective function within the MTPA and field-weakening regions. The above algorithm appears to be correct for the parameters of this particular machine, but when attempting to run it with a different set of machine parameters, the results were as depicted in Fig. 3.1.

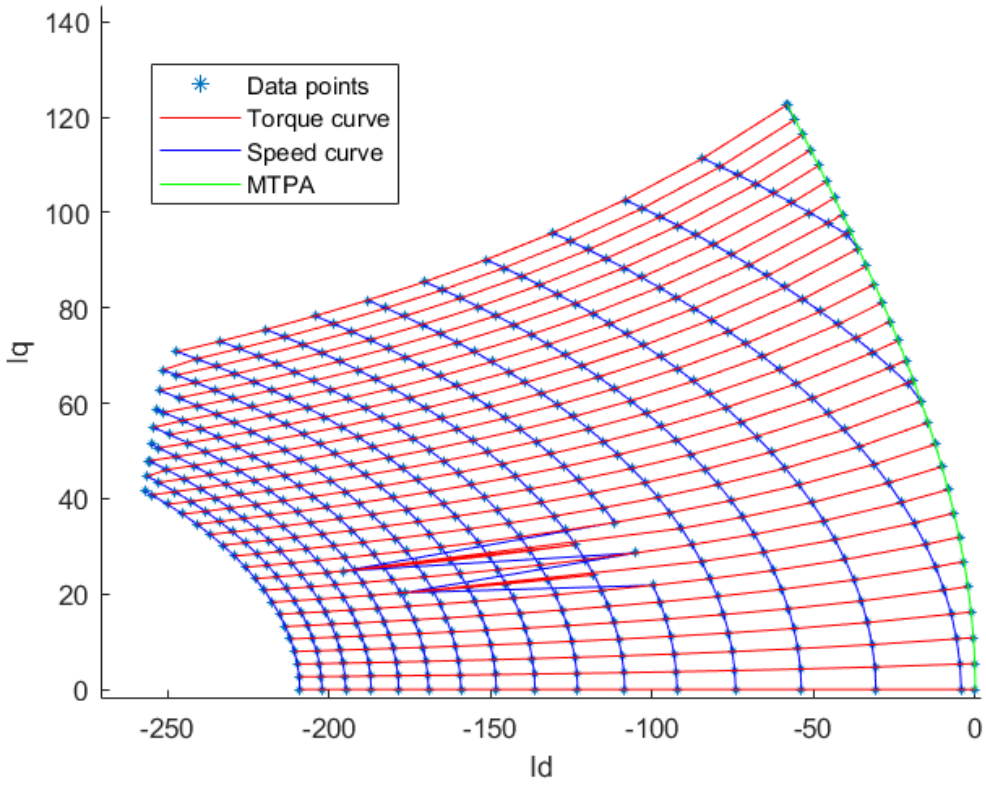


Figure 3.1: LuT with wrong data points

From the above figure, it can be seen that for this machine, the optimal operating points were not correctly found in some regions. By examining the calculated values of i_d and i_q, it can be observed that for some operating points, the current does not show a monotonically increasing trend with increasing torque reference at the same speed reference. Therefore, starting from this issue, the first attempt was to add two conditions in the nonlinear constraints part of *fmincon*, that is $|i_{d1}| \geq |i_{d0}|$, $|i_{q1}| \geq |i_{q0}|$, where $|i_{d0}|$ and $|i_{q0}|$ represent the previous values of $|i_{d1}|$ and $|i_{q1}|$. These

conditions are used to force the current to increase monotonically with increasing reference values of torque and speed.
However, even after adding these conditions, the algorithm still did not yield satisfactory results. Considering that the default setting of *fmincon* is to find local minimum values rather than the desired global minimum value, some additional information is required to help *fmincon* solve this problem, which cannot be solved by simply adding nonlinear constraints. In the previous algorithm, the initial point of *fmincon* was set to [0 0], which means that when calculating each optimal data point, the independent variable starts increasing from 0. This calculation method can inevitably cause *fmincon* to get stuck at the first local minimum it encounters, and it also greatly increases the unnecessary workload for each point search. The initial point can be based on the previous data point, which not only eliminates some non-essential computational workload but also prevents the algorithm from getting stuck in a local minimum.

The pseudo-code of the improved algorithm is as follows:

```
% Generate w_ref and T_ref
w_ref = linspace(0,w_max,30); T_ref = linspace(0,T_max,30)

% Run main function
for i=1:length(w_ref)
    for j=1:length(T_ref)
        if i ~= 1 && j~=1
            ref_i = i
            ref_j = j-1
        elseif i ~= 1 && j==1
            ref_i = i-1
            ref_j = j
        elseif i==1 && j~=1
            ref_i = i
            ref_j = j-1
        elseif i==1 && j==1
            ref_i = 1
            ref_j = 1
        end
        id_ref(1, 1) = 0
        iq_ref(1, 1) = 0
        I0 = [id_ref(ref_i , ref_j) iq_ref(ref_i , ref_j)]

        LuT_temp = LuT(w_ref(i), T_ref(j), R, Ld,...
                        Lq, psi_m, I_smax, p, u_max, I0)
        id_ref(i,j) = LuT_temp(1)
        iq_ref(i,j) = LuT_temp(2)
    end
end

% Function - LuT
function LuT(w_ref, T_ref, R, Ld, Lq, psi_m, ...
                I_smax, p, V_max, I0)
    A = []; b = []; Aeq = []; beq = []; lb = []; ub = []
    I = fmincon(objfun_LuT, I0, A, b, Aeq, beq, lb, ub,...
                nonlcon_LuT)
    id_ref = I(1)
    iq_ref = I(2)
    LuT = [id_ref; iq_ref]
end function
```

The data points of LuT obtained using the improved algorithm are satisfactory. The results of running the new algorithm model are as in Fig. 3.2.

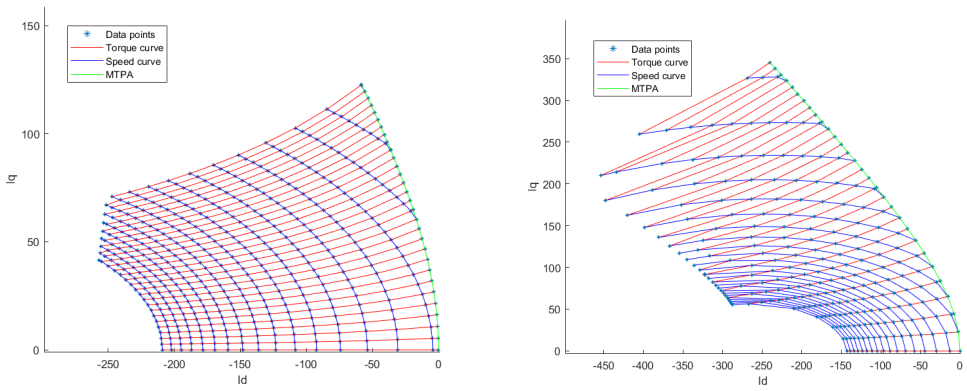


Figure 3.2: Data points of LuT (left graph used the parameters from another machine, right graph shows the parameters of the local machine)

As can be seen, the algorithm has calculated accurate data points for both sets of machine parameters shown in the figure. The constant speed curves and the constant torque curves are relatively smooth. If a more refined resolution is used (more data points are involved in the calculation), the curves will become even smoother. The improved algorithm has also significantly reduced the computation time for both sets of parameters, from about 150s initially to about 48s.

3.2 Extended LuT-based method

From the results of the previous section, it can be seen that using a LuT-based method with constant parameters is relatively simple, and the computational workload is kept within a small range, making the calculation process relatively fast. However, ignoring the parameter changes caused by magnetic saturation and cross-coupling effects in the actual application process will lead to a reduction in torque accuracy and make it difficult to cope with complex operating conditions.

Furthermore, in practical applications of IPMSM in electric vehicles, the bus voltage received by the machine may fluctuate with different operating conditions. Some studies have ignored the influence of temperature on resistance. In [21], the author proposes a torque-speed two-dimensional look-up table considering DC-bus voltage variation under the allowable maximum temperature by converting the speed formula proportionally. Although this approach requires less engineering effort, it sacrifices a portion of the control performance.

To achieve more accurate control, the basic LuT-based method proposed earlier is extended by considering the nonlinearity of the parameters and the effects of different voltage levels on the machine's performance.

3.2.1 Parameters of the nonlinear IPMSM

Table 3.2: Parameters of the nonlinear IPMSM

Parameter	Symbol	Value
Number of pole pairs	p	4
Stator resistance	R_s	0.05 Ω
DC linkage voltage	U_{dc}	0 - 400 V
Maximum stator current	I_{smax}	500 A
Maximum speed	w_{max}	12000 rpm
Maximum torque	T_{max}	400 Nm

The value of inductance and flux linkage of the IPMSM used here were obtained through the finite element method [20]. Due to the high computational resource requirement and time-consuming nature of finite element simulation, the data points obtained from simulation are usually sparse. In order to establish a more accurate LuT, it is necessary to fit the relationship between inductance, flux linkage, and current. The fitting result is shown in Fig. 3.3.

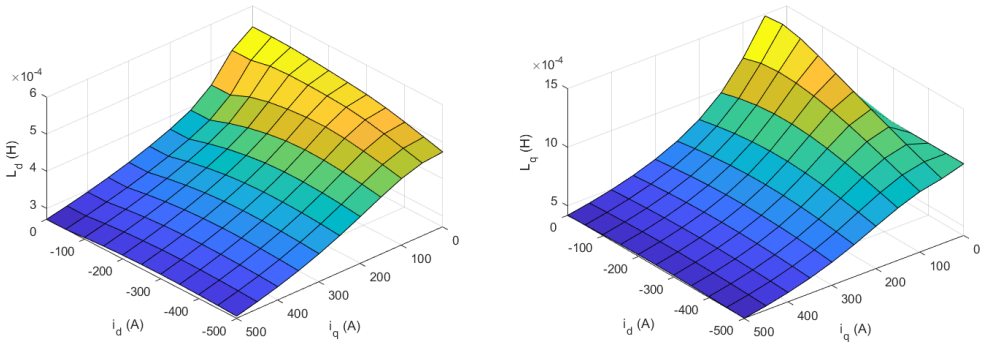


Figure 3.3: The left figure represents the relationship between L_d and current i_d&i_q, the right figure represents the relationship between L_q and current i_d&i_q

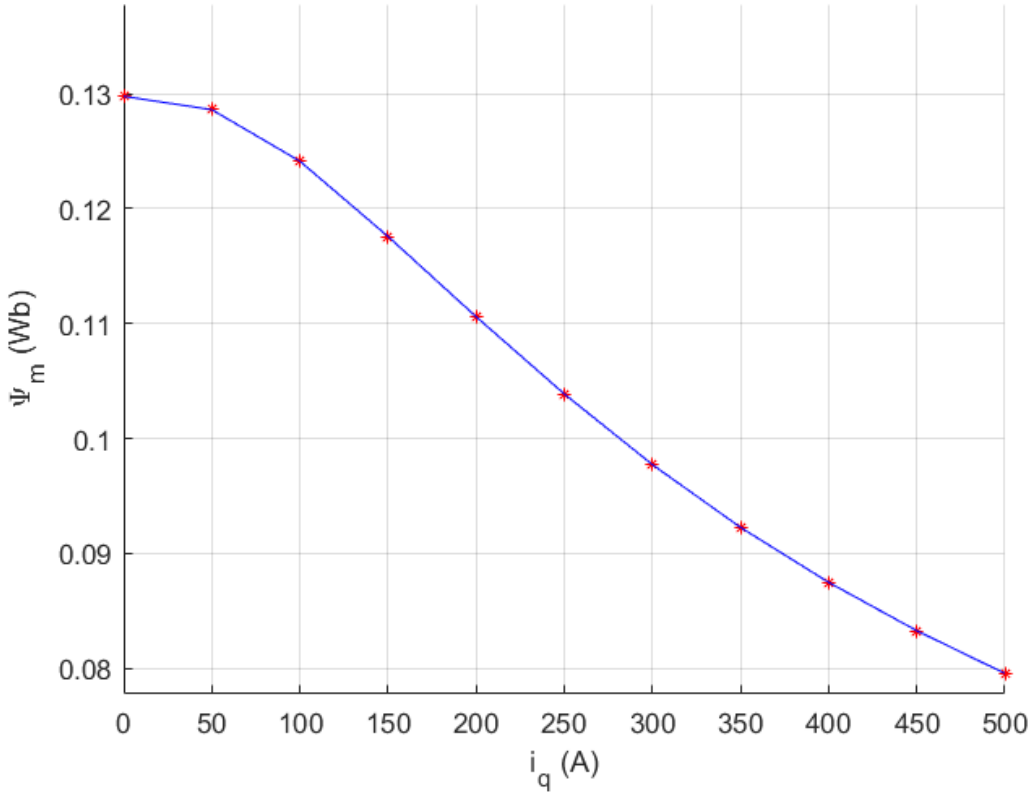


Figure 3.4: The relationship between ψ_m and i_q. Since ψ_m only depends on the i_q, this figure is a 2-D graph

3.2.2 Voltage dimension of the LuT

Based on the basic LuT-based method, adding a voltage requires another outer loop outside the existing two loops. The new loop increases from U_{smin} to U_{smax}. The corresponding changes within the algorithm are mainly focused on the nonlinear constraint part, where the value of U_{smax} in formula 3.2 is set as the increment of the newly added voltage dimension. It should be noted that the voltage involved in the calculation is the phase voltage. The relationship between phase voltage and line voltage is $U_{smax} = U_{dc}/\sqrt{3}$ [22].

3.2.3 Finding optimal operation points

The extended LuT-based method is designed for the nonlinear IPMSM model, and the algorithm requires a parameter LuT to obtain the corresponding parameters for different i_d and i_q values.

As in the previous section, the algorithm is illustrated more intuitively here with the help of pseudo-code:

```
% Generate w_ref and T_ref
u_ref = linspace(u_min,u_max,4); w_ref = linspace(0,w_max,30)
T_ref = linspace(0,T_max,30)

% Run main function
for v=1:length(u_ref)
    for i=1:length(w_ref)
        for j=1:length(T_ref)
            if i ~= 1 && j~=1
                ref_i = i
                ref_j = j-1
            elseif i ~= 1 && j==1
                ref_i = i-1
                ref_j = j
            elseif i==1 && j~=1
                ref_i = i
                ref_j = j-1
            elseif i==1 && j==1
                ref_i = 1
                ref_j = 1
            end
            id_ref(1,1,v) = 0
            iq_ref(1,1,v) = 0
            I0 = [id_ref(ref_i,ref_j,v) ...
                  iq_ref(ref_i,ref_j,v)]

            LuT_temp = LuT(w_ref, T_ref, R, I_smax, p,...
                           I0, Id_mat, Iq_mat, Ld_mat,...
                           Lq_mat, Iq_zero_Id, psim_mat)
            id_ref(i,j,v) = LuT_temp(1)
            iq_ref(i,j,v) = LuT_temp(2)
        end
    end

% Function - LuT
function LuT(w_ref, T_ref, R, I_smax, p,I0,Id_mat,...
             Iq_mat, Ld_mat, Lq_mat, Iq_zero_Id, psim_mat)
    A = []; b = []; Aeq = []; beq = []; lb = []; ub = []
    I = fmincon(objfun_LuT, I0, A, b, Aeq, beq, lb, ub,...
                nonlcon_LuT)
    id_ref = I(1)
    iq_ref = I(2)
    LuT = [id_ref; iq_ref]
end function
```

The objective function is defined the same as in the previous section, but the algorithm requires the use of a parameter LuT to interpolate and search for the corresponding parameters for different i_d and i_q values. From the results of the algorithm's execution, a similar problem to that in the basic LuT-based method was observed, even though the initial point has been adjusted here.
Considering adding the previously mentioned condition $|i_{d1}| \geq |i_{d0}|$ to the nonlinear constraints, the modified model provides relatively ideal results. The calculated data points are shown in Fig. 3.5.

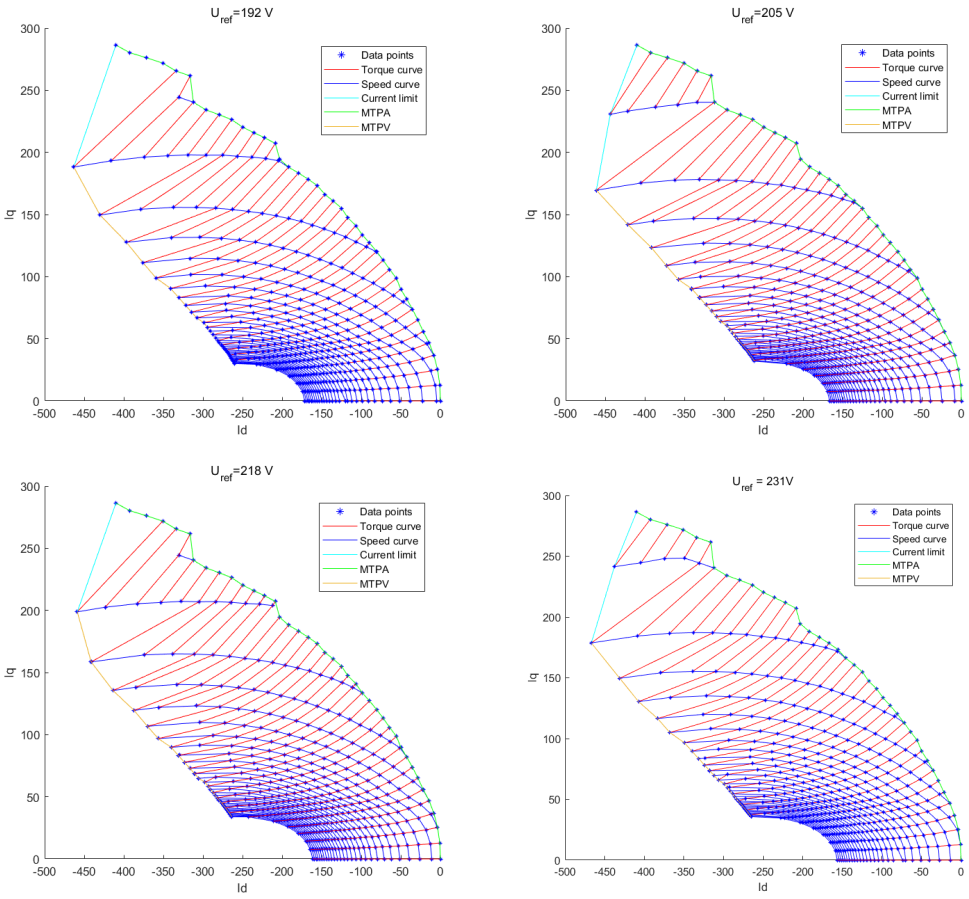


Figure 3.5: Data points of the extended LuT (voltage range from 192V to 231V)

To facilitate implementation, this set of data points has been calculated using four evenly chosen voltage values between $U_{smin} = 192\text{V}$ and $U_{smax} = 400/\sqrt{3}\text{V}$. In actual applications, the range and resolution of voltage changes can be adjusted as needed.
From Fig. 3.5, it can be seen that the data points calculated using the extended LuT-based method cannot form a smooth MTPA curve. This is mainly attributed to the fact that the original data comes from experimental measurements, and there are nonlinear factors in the flux path of the nonlinear machine [9]. Sometimes, curve fitting techniques may be used to achieve smoothness if needed.

4

Machine Learning

ML (Machine Learning) exhibits excellent feature representation capabilities when dealing with multi-input multi-output, and nonlinear data. Appropriately, the IPMSM is a highly complex system that is multivariate, nonlinear, and strongly coupled. This suggests that ML is highly suitable for IPMSM control. Another advantage of using an ML model is that it can be saved in a pickle format [25], which requires minimal on-disk storage space.
This chapter proposes two different ML models for IPMSM current control. One is a data-driven supervised learning model based on a neural network, which requires learning from the data of the extended LuT. The other is a reinforcement learning model based on the SARSA algorithm [34], which enables self-learning through feedback from its own actions and experiences, without the need for a large dataset for training. This chapter includes specific algorithms used by both models, the structure of the neural network, data preparation, the process of model training, parameter tuning, and the analysis of the results after training.

4.1 Supervised learning

Supervised learning is a subclass of ML and is an active research field in ML. It is often applied in various fields such as retail, finance, medicine, and transportation [24]. The implementation process of supervised learning algorithms is similar to the process in which humans gain their own experiences by learning past knowledge and applying their summarized knowledge to similar new problems. This algorithm requires learning from annotated training sets, which have correctly matched input-output pairs and represent experiences in certain real-world applications. The model analyzes the relationship between different inputs and outputs to gradually improve the fitting to this relationship. It measures its accuracy through a loss function and updates it until the error is sufficiently minimized. The trained model applies the learned experiences to current data to predict future events.
Supervised learning can be divided into two types of problems: classification and regression. Classification problems often refer to problems with qualitative outputs, such as cancer diagnosis, spam detection, etc. Common classification algorithms include support vector machines (SVM), linear classifiers, decision trees, k-nearest neighbors, and random forest. Regression problems, on the other hand, refer to problems with quantitative outputs. Quantitative outputs are often represented by numerical values, and these algorithms are commonly used to predict age, income, house prices, stock prices, etc. Common regression algorithms include polynomial

regression, support vector regression, gradient boosting regression, etc. In this section, the multi-layer perceptron regressor (MLPRegressor) from scikit-learn [28] is used to predict the reference value of current, which belongs to the regression problem in supervised learning.

The current prediction in this case can be easily achieved by applying supervised learning. The algorithm no longer relies on the mathematical model and specific parameters of the IPMSM, which means that complex calculations can be avoided. It only needs to use the data points obtained by the extended LuT-based method as the training dataset, and learn the mapping between different reference values of torque, speed, and voltage to their corresponding currents. Then it can output the appropriate currents based on the input reference values of torque, speed, and voltage in future applications.

4.1.1 Neural network structure

The MLP is a feedforward artificial neural network model and is the most basic structure of deep learning networks. It can learn the nonlinear relationship between complex input and output variables and is an effective method for solving regression problems. The characteristics of MLP are as follows [29]:

- It is a fully connected neural network model.
- It has a layered structure organized by neurons.
- Neurons within the same layer do not connect with each other.
- Neurons in adjacent layers connect with each other.
- Each layer only receives input from the previous layer and sends its output to the adjacent next layer.

Fig. 4.1 illustrates the neural network architecture used in this thesis. It consists of an input layer, three hidden layers, and an output layer. The input layer is used to represent the input features, here three inputs are used, defined as the reference value of torque, speed, and voltage. The output of the input layer serves as the input to the next hidden layer. The output layer receives values from the last hidden layer and transforms them into output values. The outputs of the network are defined as stator currents i_d and i_q.

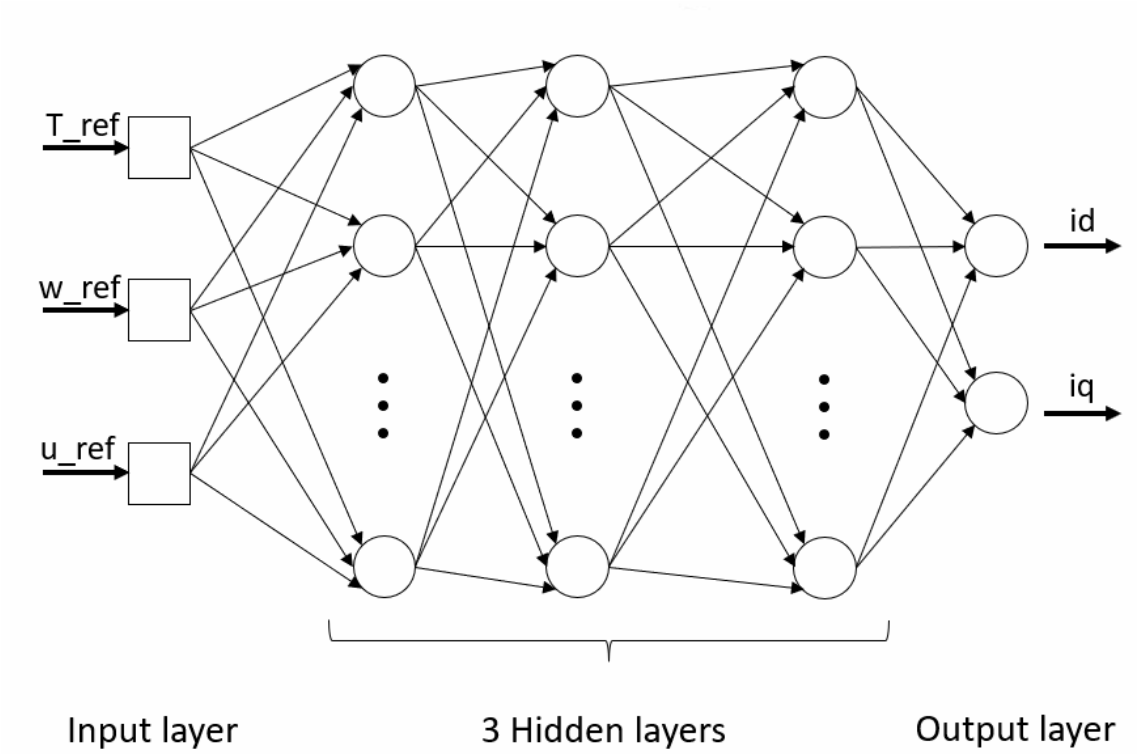


Figure 4.1: The used neural network architecture; three hidden layers, an input layer with reference values of torque, speed, and voltage, and an output layer with values of i_d and i_q.

The design of the hidden layer structure has a significant impact on the regression performance of the neural network. If the number of neurons in the hidden layer is too small, the learning and information-processing capabilities of the network may not be guaranteed. If the number of neurons in the hidden layer is too large, the probability of the network getting stuck in a local minimum will be amplified, and the running speed of the network will also be impeded. Therefore, selecting an appropriate number of hidden layer neurons and the number of hidden layers is an important part of improving and optimizing MLP. Appropriate design of the hidden layer structure can ensure accurate prediction of the stator current while maximizing the learning efficiency and running speed of the neural network.
The activation function is a type of function used to implement non-linear mapping, aiming to help the network learn complex relationships in the data. Similar to the neuron-based model in the human brain, the activation function defines the output of the current node given the input, ultimately determining the information to be passed on to the next neuron. Without introducing activation functions, each layer of the neural network can only perform linear operations, and even with multiple layers stacked on top of each other, it remains a linear operation. Linear mapping is usually not enough, and even a large-scale network often cannot handle complex problems. An activation function is a type of function used to implement non-linear mapping, and the introduction of non-linear factors can greatly enhance the mapping ability of the network, enabling the better fitting of the target function while impacting the computation time and memory usage. The backpropagation

process of the neural network needs to consider the risks of very large weight updates (gradient explosion) and the risks of the gradients getting smaller and smaller and approaching zero when the algorithm advances backward from the output layer towards the input layer, which is gradient vanishing.
This thesis chooses rectified linear unit (ReLU) [30] as the activation function. This function is a piecewise function with a mathematical expression of:

$$rectifier(x) = \begin{cases} x, & x \geq 0 \\ 0, & other \end{cases} \tag{4.1}$$

From the perspective of information processing, ReLU achieves a sparse representation of input data. This function can filter out irrelevant information or noise signals during the network learning process. When $x > 0$, the gradient of ReLU is 1, and its non-saturation can effectively solve the problem of gradient vanishing. Additionally, ReLU has a smaller computation cost compared to *sigmoid* and *tanh*, significantly reducing the computation time.

4.1.2 Model training

Table 4.1: Computer configuration

CPU	Intel(R) Xeon(R) CPU E5-2697 v3 @ 2.60GHz
Operating system	Linux
Memory	528251516 kB

4.1.2.1 Dataset

The dataset used in this study is operation points generated by the extended LuT-based method. The dataset comprises 3272 instances. Each instance contains three independent features and two dependent features. The three independent features are reference torque, speed, and voltage, denoted as T_{ref}, w_{ref}, and u_{ref}, respectively. The two dependent features are stator currents: i_d and i_q.

$$\begin{aligned} x &= (T_{ref}, w_{ref}, u_{ref})^T \\ y &= (i_d, i_q)^T \end{aligned} \tag{4.2}$$

The test set in the dataset accounts for 20% of the instances, while the training and validation sets are split in an 8:2 ratio. Therefore, there are 654 instances in the test set, 2094 instances in the training set, and 524 instances in the validation set. To improve the performance of the model and the stability of the training process, the data is standardized using *StandardScaler*() module [28] prior to training, which centers the values around the mean with a unit standard deviation.

4.1.2.2 Feedforward and backpropagation

MLP does not have feedback connections. All signals propagate forward from the current layer. This process can be represented using the following formula [31]:

$$h_i^{k+1} = \sigma(z_i^{k+1}) = \sigma(b_{k+1} + \sum_{j=1}^{n} w_{i,j}^k \cdot h_j^k) \tag{4.3}$$

where h_i^{k+1} is the output of a neuron i in the layer $k+1$, σ is the activation function (ReLU), b_{k+1} is the bias of the neurons in layer k+1, $w_{i,j}^k$ represents the weight that connects the neuron j in the layer k to the neuron i in the layer $k+1$.
The training process of a feedforward neural network can be viewed as the process of minimizing the loss function. The loss function is used to calculate the error between the network's predicted output and the true output from the training set. As the network learns, this error value becomes smaller and smaller. When the error reaches a predetermined stopping criterion, the network training is considered finished. Here, the loss function is set to be Mean Squared Error (MSE).

$$J = \frac{1}{n} \sum_{1}^{n} \|y_i - \hat{y}_i\|^2 \tag{4.4}$$

The loss J is used to update the weights and biases so that the predicted output of the network approaches the known true values as closely as possible. In order to determine the direction of these parameter changes, backpropagation is used to calculate the partial derivatives. Typically, the following formula is used for updating weights [31]:

$$\begin{aligned} w_{t+1} &= w_t - \epsilon \frac{\partial J}{\partial w} \\ \frac{\partial J}{\partial w_{i,j}^k} &= \frac{\partial z_i^{k+1}}{\partial w_{i,j}^k} \frac{\partial h_i^{k+1}}{\partial z_i^{k+1}} \frac{\partial J}{\partial h_i^{k+1}} \end{aligned} \tag{4.5}$$

For bias, the update rule is as follows:

$$\begin{aligned} b_{t+1} &= b_t - \epsilon \frac{\partial J}{\partial b} \\ \frac{\partial J}{\partial b_{k+1}} &= \frac{\partial z_i^{k+1}}{\partial b_{k+1}} \frac{\partial h_i^{k+1}}{\partial z_i^{k+1}} \frac{\partial J}{\partial h_i^{k+1}} \end{aligned} \tag{4.6}$$

The variable t represents the learning step, and ϵ is the learning rate, which determines the speed of updating weights and bias, and the step size of moving the gradient towards the optimal solution of the loss function in each iteration. The above formulas can be used to determine how quickly the loss changes when weights and biases are modified, helping to better understand the overall behavior of the network as weights and biases are adjusted.
The gradient descent algorithm aims to find the optimal weights and biases to minimize the loss function. Although it is one of the most popular optimization algorithms, it can run slowly and be relatively inefficient. For the training of this neural

network, the Adam optimizer [32] is chosen instead. It is an extension of stochastic gradient descent with characteristics such as ease of implementation, high computational efficiency, and low memory requirements. It has excellent optimization performance for problems with large amounts of data and parameters.

4.1.2.3 Hyperparameter tuning

After sweeping several combinations of hyperparameters, the final choice is:

Table 4.2: Parameters of the model

Parameter	Value
Training set	2094
Validation set	524
Test set	654
Iterations	500
Hidden layer sizes	$(30, 20, 20)$
Learning rate	0.0025

Considering the real-time application of the model in electrical drive systems, it is not advisable to use overly complex network structures. Therefore, after trying several hidden layer sizes, the parameters were set to $(30, 20, 20)$, which means there are 30 neurons for the first hidden layer, and 20 neurons for the second and third hidden layer respectively. This size can ensure the learning effect of the model without making the network too complex. The learning rate was set to 0.0025. A learning rate that is too small will slow down the convergence process and easily lead to overfitting. On the other hand, a learning rate that is too large can speed up the convergence process but lead to unstable convergence near the minimum value and even prevent the gradient from converging.

The goal of training a neural network is to achieve better generalization performance. As the performance of the network on the training set improves, the network may gradually perform worse on the test set, indicating the potential occurrence of overfitting. Early stopping is an effective method to prevent overfitting. It terminates the training process when the validation score stops improving.

4.1.3 Results of NN

The model evaluation is conducted from two aspects: the error between predicted values and true values, and the training time and prediction time. The former is quantified using the loss function and the R^2 score. The R^2 score indicates how much of the variation of a dependent variable is explained by an independent variable in a regression model [26]. The following graphs show the loss curve and the partial visualization (only MTPA) of the model:

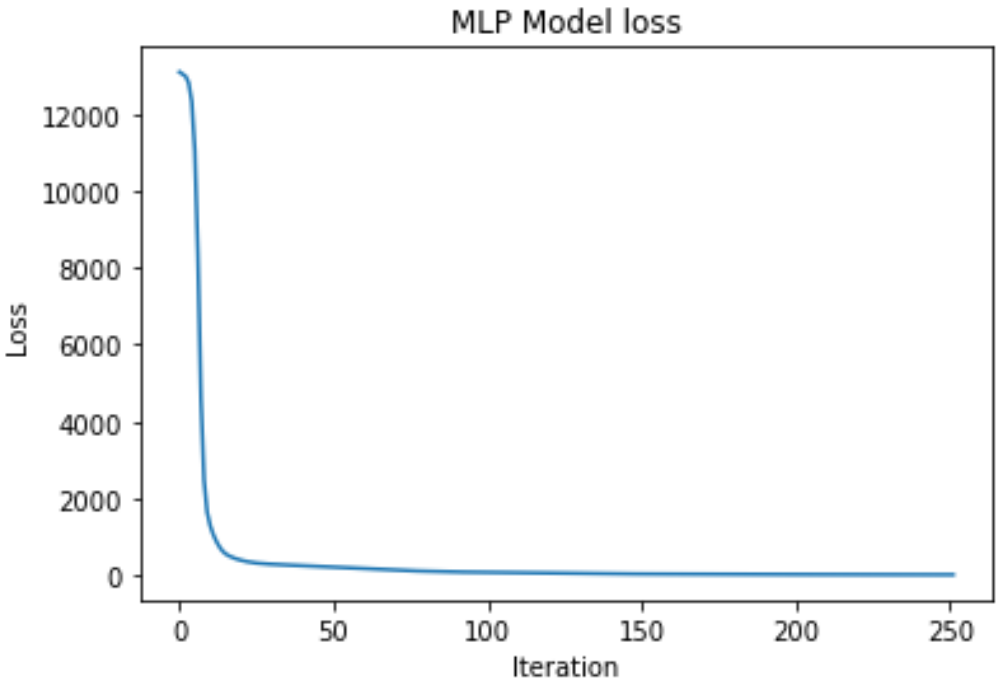


Figure 4.2: Loss curve of NN model

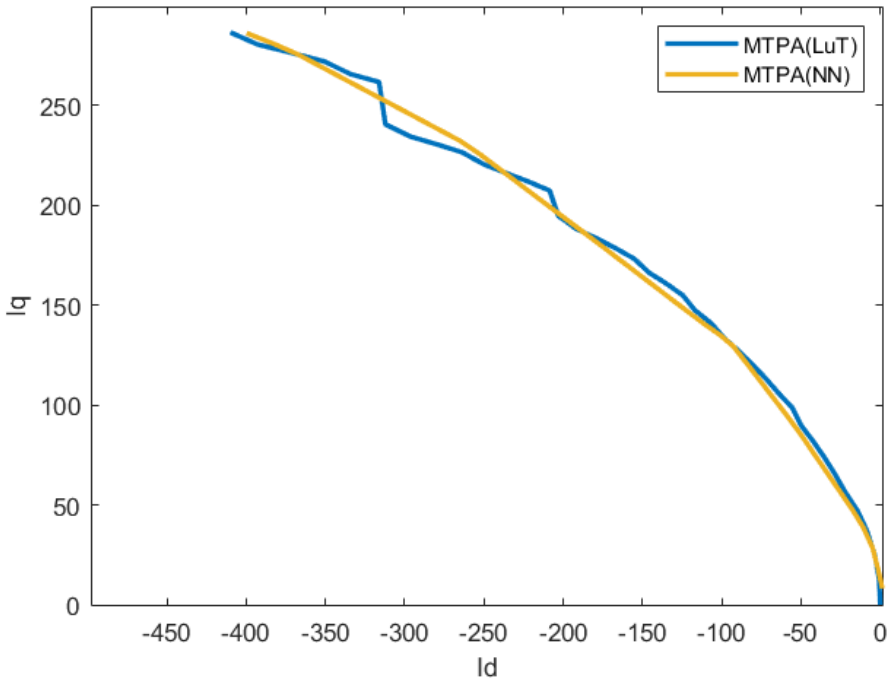


Figure 4.3: Comparison of the MTPA curve from LuT-based method and NN-based method

It can be observed that the loss curve has reached a stable state, and early stopping is executed at iteration 252. At the completion of training, the loss value is 14.2.

These results indicate the satisfactory performance of the trained model.
Upon visualizing the MTPA curve predicted by the model, it can be seen that the ML model has not completely fitted the intricate portions, suggesting the absence of overfitting. Furthermore, insights from Section 2.4.1 suggest that smoother curves are advantageous for improving performance during actual control.
The R^2 score on the test set is 0.996. The closer the R^2 score is to 1, the better the model's performance [26]. Therefore, a score of 0.996 indicates that the model has performed well in training.

4.2 Reinforcement learning

RL (Reinforcement Learning) is another branch of machine learning. As a recent research hotspot, it has been widely applied in various fields such as robot control, industrial manufacturing, game theory, parameter optimization, and autonomous driving [34]. Unlike supervised learning, which requires training based on data, the fundamental idea of reinforcement learning is that an agent learns the optimal policy of a dynamic system by perceiving the environment's evolution. Through trial and error, the agent continuously interacts with the environment to improve its behavior. And it has a low requirement for prior knowledge about the environment.
The task of reinforcement learning is to learn the mapping from the state space to the action space, essentially approximating the relationship between "state" and "action" using parameterized functions. Common algorithms in reinforcement learning, such as Q-learning, Temporal Difference learning (TD learning), and SARSA, share the characteristic of estimating only the value function. The value function is used to predict the expected future returns, and it facilitates the evaluation of different strategies in a convenient manner.
Compared to supervised learning, using reinforcement learning for IPMSM current control eliminates the need for building a LuT in the initial phase. The objective of the model in this section is to establish the MTPA curve for the linear IPMSM model. The core idea draws inspiration from the Grid World Game [33], which is one of the most fundamental and classic problems in reinforcement learning. Starting from this simple problem, improvements are made by setting the horizontal coordinate in the grid world as the value of i_d and the vertical coordinate as the value of i_q. The agent in the original model had four directions as actions. However, considering that a curve is required for this case, four diagonal directions (upper left, lower left, upper right, lower right) were added to the actions. The previously defined reward function was relatively simple, with specific numerical rewards assigned to different states. Specifically, the agent receives a reward of 10 if it reaches the final target, a reward of 5 if it reaches some secondary objectives, and all other actions result in a reward of -1. These values are not fixed, as long as their relative magnitudes remain essentially unchanged. This reward mechanism enables the agent to find the shortest path from the starting point to the final target. However, when applying the Grid World concept to finding the MTPA path, the existing reward mechanism cannot be directly transferred. This is because in this case, there is no known final target, and every point on the MTPA path is equally important as a target. The agent's task itself is to search for each individual target along the MTPA path.

Therefore, when constructing this model, the reward function needs to be redefined. Starting from the objective of aligning the agent's path with the MTPA curve, the key reward can be initially set as the torque per ampere. This means that when the agent finds the correct point, it receives the maximum reward. Meanwhile, other states of the agent are constrained by boundaries such as the current limit, and the reward for corresponding actions is set as a fixed negative constant. However, this setting is not optimal, and it is further explained in Section 4.2.2.

4.2.1 SARSA

SARSA is a classic on-policy reinforcement learning algorithm. The first step of the algorithm is to provide a policy π, and then estimate the value function for each state-action pair. The update strategy is as follows [34]:

$$Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma Q(S', A') - Q(S, A)] \tag{4.7}$$

where α is learning rate, γ is discount factor, S is current state, S' is next state, A is current action, A' is next action, R is reward. In SARSA, the agent determines the next action based on the current state-action pair and updates Q using the value function corresponding to the next action during the iterative process. The convergence performance of the SARSA algorithm depends on the dependence of the policy on the value function. Designing an effective exploration strategy is crucial for the algorithm's performance. The pseudo-code for SARSA is as follows:

```
Initialize α, γ, Q(S,A)

Loop for each episode:
    Initialize S
    Choose A from S using policy from Q (e.g.,ε − greedy)
    Q(S,A) ← Q(S,A) + α[R + γQ(S',A') − Q(S,A)]
    S ← S'; A ← A'
    until S is terminal
```

The ϵ-greedy strategy is a simple method used in the algorithm to determine whether to explore or exploit [34]. The exploration-exploitation trade-off is a challenge that the agent faces during the learning process. Exploitation refers to the agent's tendency to select actions that have high rewards based on the current known Q-value matrix in order to maximize immediate rewards. However, excessive exploitation can lead to the system getting stuck in a particular direction of search and may fail to converge to the optimal solution. On the other hand, exploration refers to the agent's need to explore all state-action pairs to acquire comprehensive and sufficient learning experiences, thus avoiding convergence to local optima. However, excessive exploration can result in an abundance of redundant information, reducing learning efficiency. The ϵ-greedy strategy balances exploration and exploitation by selecting exploration and exploitation actions with probabilities ϵ and $1 - \epsilon$, respectively.

4.2.2 Model training

Computer configuration is as reported in Section 4.1.

4.2.2.1 Hyperparameter tuning

Setting the right hyperparameters is crucial for achieving good performance in reinforcement learning algorithms. For the SARSA-based reinforcement learning model, the key hyperparameters that determine performance are the discount factor γ, learning rate α, and exploration rate ϵ. For determining γ and α, an initial attempt is made using the grid search method. The hyperparameters are varied within the range of 0 to 1 with an interval of 0.1. Due to the extensive computation time required for model execution, a preliminary approach is to set the number of episodes to 10,000 and run the model. The optimal hyperparameters determined using this method are $\gamma = 0.3$ and $\alpha = 0.9$. Based on experience, these parameter values are not suitable, and the results obtained from using them are completely unreasonable. The significant bias observed may be due to the small number of episodes used in the grid search. However, increasing the number of episodes for an exhaustive grid search would be impractical due to the substantial time requirements. Based on empirical knowledge, a more reasonable choice is to set $\gamma = 0.9$ and $\alpha = 0.1$. As for the exploration rate ϵ, this study adopts a decaying ϵ over time. Initially, ϵ starts at 1.0 and gradually approaches a very small value (0.00001) towards the end of training. This choice is made because, in the beginning, the agent needs to explore the state space to acquire sufficient experience, while in the later stages of training, it should use its learned knowledge effectively, resulting in significantly reduced exploration.

4.2.2.2 Model environment

The environment is a fundamental component of reinforcement learning. The agent interacts with the environment through actions, and the environment provides new states and reward feedback, prompting the agent to choose more appropriate strategies based on them to maximize the reward. A suitable environment is crucial for the performance of a reinforcement learning model. When constructing the environment class, reference was made to the composition of environments in OpenAI Gym [35].
First, the observation (state) space and action space are defined. The observation space corresponds to the agent's coordinates in the grid world and represents the stator currents i_d and i_q of the machine. The action space is represented by numbers 0-7, indicating the eight directions in which the agent can move. The parameters of the machine also need to be defined here. To alleviate the training difficulty of the RL model, the linear parameters described in Section 3.1.1 are used.
Subsequently, the step function is defined, which takes an action as input and returns the new observation and reward after performing the action. Based on the numerical value in the input action, the agent's movement is defined in the eight directions, and the new observation is returned. The reward function is crucial in reinforcement learning, and its detailed configuration is discussed in Section 4.2.2.3. Additionally, the reset function is used to reset the environment to its initial state after each

episode, and the maximum number of iterations within each episode is also reset at this point.

4.2.2.3 MTPA trajectory as reward function

The reward function defines the objective of the model. Since the agent learns by maximizing the cumulative reward, the construction of the reward function plays a crucial role in the model's performance. The reward should provide the agent with information about the goodness or badness of the current action taken. In this case, the reward function would ideally be defined such that the agent receives a positive reward when it finds an MTPA point and a negative reward (punishment) when it fails to find one. However, this is not feasible since the desired results cannot be known in advance. If the MTPA point were already known, there would be no need for a reinforcement learning model. Instead, an alternative approach is taken. The reward is directly set to the calculation formula of torque per ampere, which is: $T/\sqrt{i_d^2 + i_q^2}$. This reward function encourages the agent to take actions that maximize the torque per ampere, thereby achieving the goal of the model. Additionally, a penalty is defined when the agent exceeds the current limits or the specified quadrants, where the reward is set to -1000. However, this defined reward function did not yield useful results. Inspired by the objective function of the LuT-based method, a squared term is added to the torque in the reward function to further modify it: $T^2/\sqrt{i_d^2 + i_q^2}$. The following Fig. 4.4 illustrates the results:

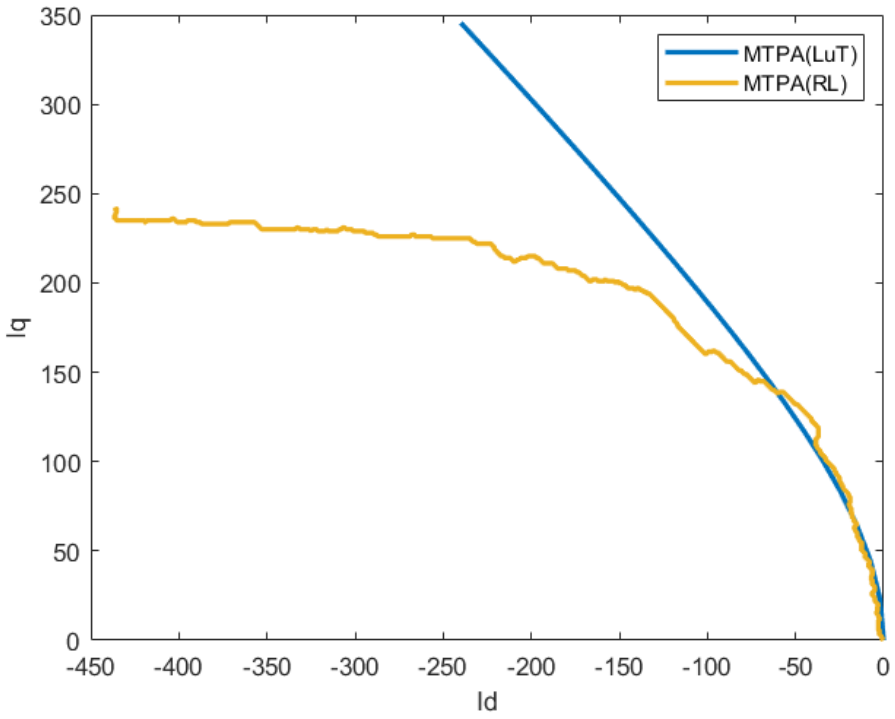


Figure 4.4: Comparison of the MTPA curve from LuT-based method and RL-based method

Here, the MTPA curve from LuT is used as the baseline, and it can be observed that the MTPA curve from the RL model only closely fits the baseline in the first one-third segment. However, compared to the initial model without the squared term, there has been a significant improvement. Therefore, it can be inferred that the performance of the model may be influenced by the power of the variable T in the reward function. A higher power of T indicates a larger weight, and an increase in T has a greater impact on the results. Consequently, the power of T is set to 4, 6, and 8, respectively, and the training results of the model are as follows:

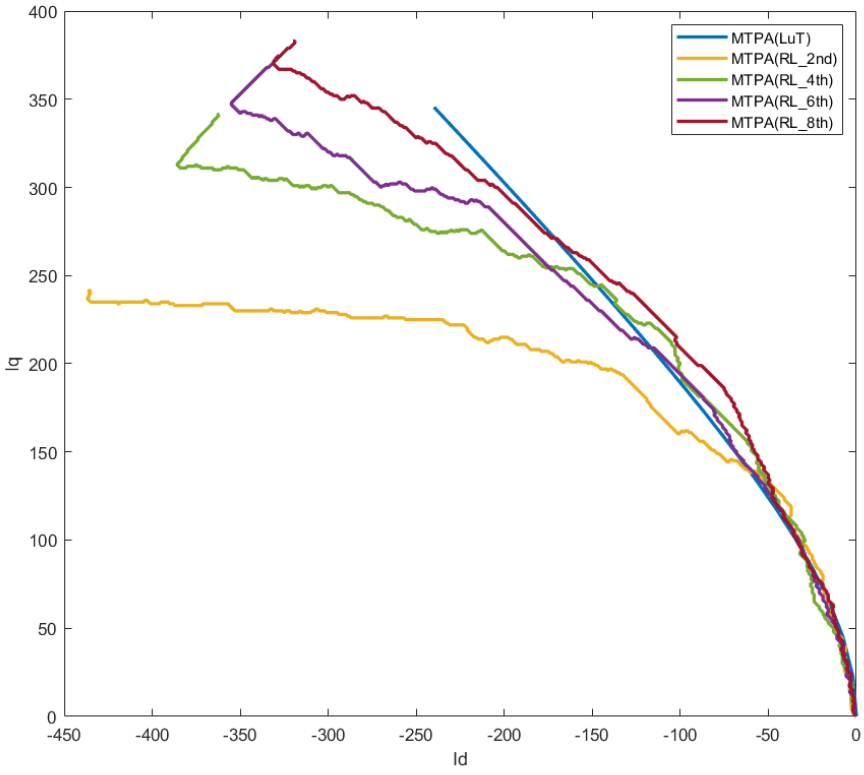


Figure 4.5: Comparison of the MTPA curve from LuT-based method and RL-based method with different powers of T.

By increasing the power of T, the results of the model improve progressively, and the MTPA curve from the RL model gets closer to the baseline in the last one-third segment. However, a consequence of this is that the middle one-third segment gradually deviates from the baseline. It is not advisable to blindly pursue higher powers of T as it may lead to undesirable outcomes and extended computational overhead.

Based on the power of T being 8, adjustments were made to the numerator using a polynomial form. Specifically, three variations were attempted: $\frac{T^8-T^6}{\sqrt{i_d^2+i_q^2}}$, $\frac{T^8+T^4}{i_d^2+i_q^2}$, and $\frac{T^8-T^6-T^4}{\sqrt{i_d^2+i_q^2}}$. The training results are presented below:

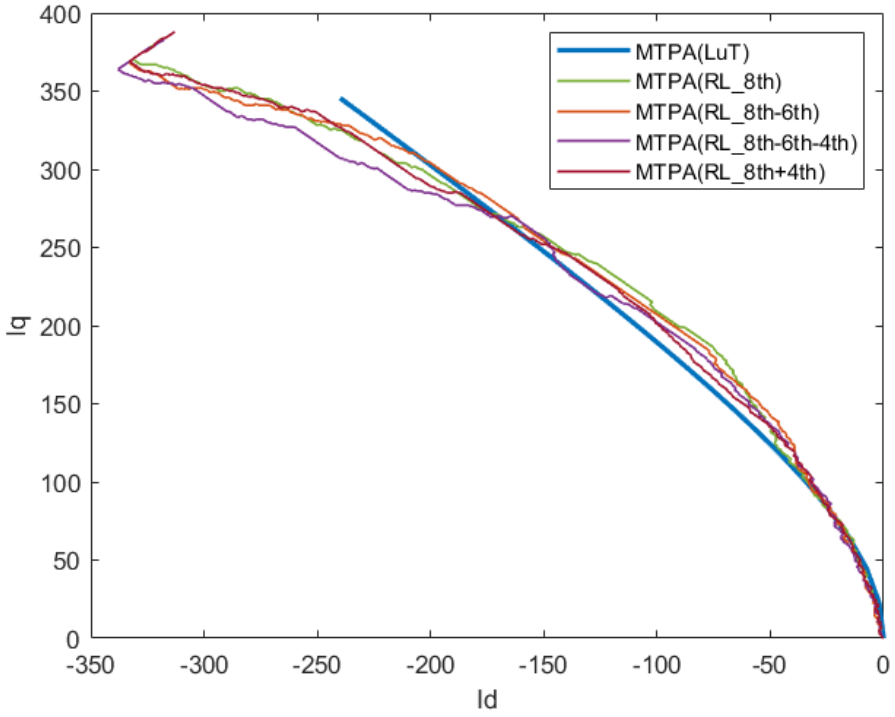


Figure 4.6: Comparison of the MTPA curve from LuT-based method and RL-based method with different numerators of the reward function.

In the reward function, the adoption of different polynomial forms for the numerator results yields highly similar outcomes. The next section focuses on quantitatively analyzing these results.

4.2.3 Results of RL

In the current models, the agent's movement step size is set to one Ampere. For the models, a finer resolution is more advantageous in obtaining optimal operation points during actual control. However, this also means that the training duration of the model will significantly increase. The small benefits gained from increasing the resolution are not sufficient to compensate for the substantial increase in training time. Therefore, the agent's step size is set to 1A, ensuring both accuracy and limiting the training duration within an acceptable range for this thesis work.

The quantitative evaluation of the model's results used two metrics: MSE and R^2 score. It should be noted that aiming to minimize the use of machine parameters in the RL model, the maximum torque parameter was not introduced. As a result, in Fig. 4.5, the MTPA curves from the RL model are consistently higher than the baseline. When comparing the results, it is assumed that the corresponding sections with the same x-axis coordinate as the baseline are selected, and curve fitting is performed.

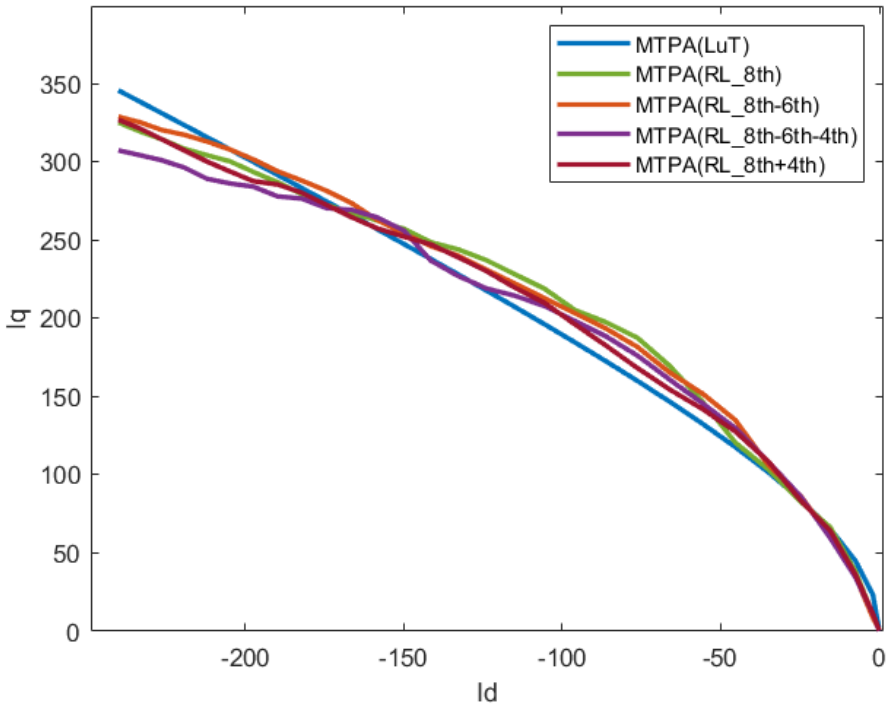


Figure 4.7: Comparison of the MTPA curve from LuT-based method and RL-based method

The evaluation results of different models are presented in the Table 4.3.

Table 4.3: Evaluation of different RL models

Models	MSE	R^2
8th	409.93	0.9790
8th-6th	291.07	0.9851
8th-6th-4th	524.38	0.9731
8th+4th	224.67	0.9885

Lower values of MSE indicate better performance, while higher values of R^2 score indicate better goodness-of-fit between the predicted and actual values. When comparing the three sets of results, it is observed that the model with the reward function

$$\frac{T^8 + T^4}{i_d^2 + i_q^2} \tag{4.8}$$

exhibits the lowest MSE and the highest R^2 score.

5

Results

This section comprehensively quantifies the results of LuT-based methods, NN-based methods, and RL-based methods.

Firstly, using the LuT-based method as a baseline, we compare the fitting performance of ML models and LuT. This is crucial as it directly determines the control performance. The following Fig 5.1 and Fig. 5.2 illustrate the fitting results of the NN model and RL model for different MTPA curve:

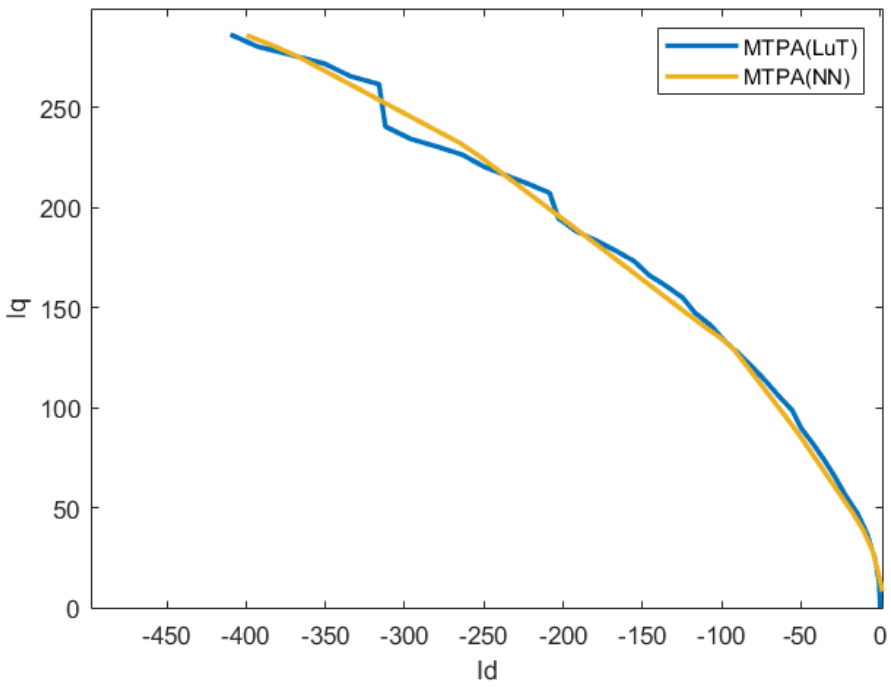


Figure 5.1: Comparison of the MTPA curve from extended LuT-based method and NN-based method

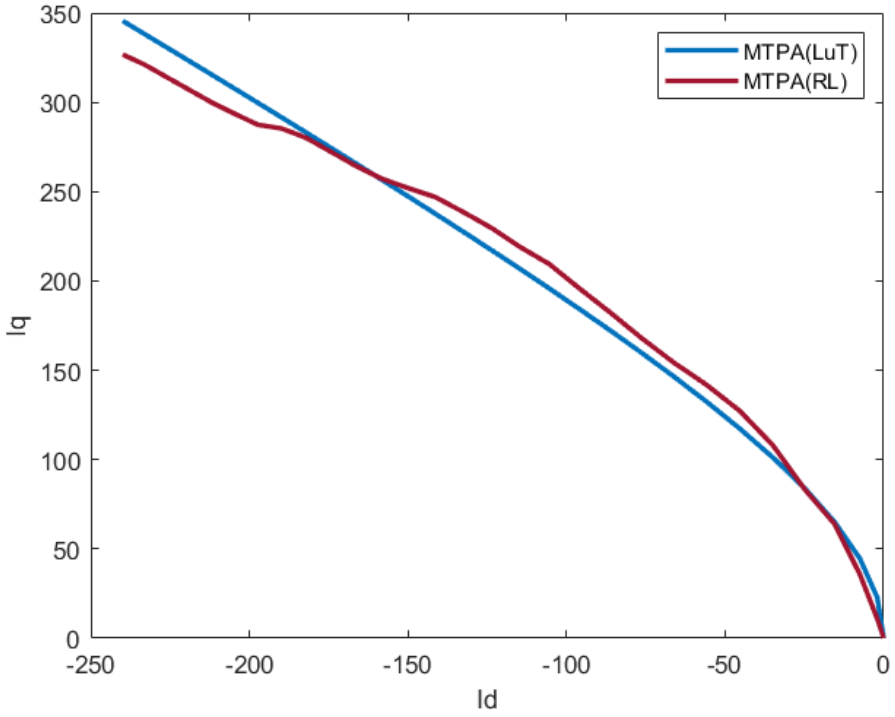


Figure 5.2: Comparison of the MTPA curve from basic LuT-based method and RL-based method

The R^2 score for the NN model is 0.996, while for the RL model, it is 0.9885. The NN model demonstrates a higher score, indicating better fitting performance. Moreover, the NN model applies to both the MTPA region and the field-weakening region, enabling control over the entire operating range. On the other hand, the RL model can only achieve control for the linear model in the MTPA operating region and exhibits a lower level of fitting compared to the NN model.
Table 5.1 presents the results for various models in terms of access time, model size, total memory used, and other relevant aspects:

Table 5.1: Performance of different models

	LuT(extended)	NN	RL
Pickled data access time	0.008s	0.001s	0.0002s
Storage space size	221KB	52KB	13KB
Memory	2412MB	172.2MB	252.2MB
Training time	238s	7s	85537s
Input data needed	Less	Large	Least
CO_2e emissions during training	0.1274g	0.0035g	45.8g

Since the models need to be applied in real-time on electric vehicles, the time required to obtain the optimal operating point is of great concern. From the table above, it can be observed that all three models have fast data access processes with

minimal differences. The RL model has the shortest access time, requiring only 0.0002s.
The following comparison focuses on the storage space requirements of different models. A larger storage space implies higher costs for storing the data. Although the table above indicates that the RL model requires the least storage space, this model is only applicable to the operating conditions of IPMSM under the same power supply voltage. Furthermore, it is limited to MTPA control and does not support field-weakening control. Therefore, what it can represent is quite limited. In comparison, the NN model is only 23.5% of the storage space of the LuT model. The size of the NN model is determined by its network structure, and under unchanged architecture, its size remains relatively fixed. On the other hand, the size of the LuT model increases as the stored content expands. In practical applications, LuTs require higher resolution, which further reduces the aforementioned percentage.
All three models are obtained through offline computations. The memory usage of a model training process is also an important metric, as lower memory consumption is advantageous in alleviating the burden on the computer and facilitating model computations. It can be observed that the NN model uses the least amount of memory. On the other hand, the LuT model consumes the most memory. In terms of training time, both the LuT model and the NN model require less time than the RL model. The NN model can complete training in a matter of seconds, while the LuT model takes longer, approximately 4 minutes, which is still within an acceptable range. However, the training time for the RL model is close to 24 hours. Compared to the previous two models, this model is excessively time-consuming, and its control capability is limited, which significantly affects the model's desirability. Regarding the input data required by the models, both the LuT model and RL model only need the parameters of the IPMSM as input, with the RL model using fewer parameters. In practical applications, obtaining the parameters of the IPMSM is often challenging, and the RL model has a significant advantage in this aspect, which is one of the main reasons for building this model. The NN model is a supervised learning model, so it needs to learn from a large amount of data, which reduces its advantages in the field of IPMSM control.
Environmental issues have become a major focus of global development. With the introduction of the concept of carbon neutrality, countries have started to control carbon emissions in national production and daily life. In this regard, this thesis considers carbon dioxide equivalent (CO_2e) emissions as one of the indicators for comparing different models. The processor used in this study is the *Intel Xeon E5-2697-v3*, with a Thermal Design Power (TDP) of 145W (0.145kW). By multiplying the TDP by the training time, we can estimate the electricity consumed during model training. The thesis was conducted in Sweden, where the country's average CO_2 emission per kilowatt-hour (kWh) of electricity generation is 13.3g [27]. Therefore, the formula for calculating CO2 emissions is:

$$TDP(kW) * Time(h) * 13.3\,(g/kWh).$$

According to the results, it is evident that the RL model has the highest carbon emissions due to its prolonged training time. On the other hand, the other two models exhibit relatively lower carbon emissions. If this thesis work were conducted

in other countries, the value of 13.3 (g/kWh) would be 682 (g/kWh) for China, 411 (g/kWh) for USA, and 705 (g/kWh) for India, respectively [27].
By comparing the results of different models, the following conclusions can be drawn: For the IPMSM current control, the commonly used LuT-based method offers stability, speed, and accuracy. However, when aiming for more precise and efficient control, LuT requires the introduction of additional dimensions, which leads to a rapid increase in storage space and longer computation time. To address this challenge, the established NN model effectively learns the complex nonlinear relationships among the data in the LuT. It achieves a high level of accuracy with a short training time and eliminates the data fluctuations present in the original LuT. These characteristics indicate that the NN model is efficient, accurate, and well-suited for data optimization and reduction purposes.
Among these models, the RL model utilizes the least amount of input data and establishes a model specifically for MTPA control under a single voltage. It also has the smallest storage space. However, creating a RL model is challenging, the training process also requires a significant amount of time. The aforementioned characteristics do not sufficiently compensate for the substantial investment required to build this model. In terms of carbon emissions, the value is directly proportional to the training time of the model. The RL model requires the longest training time, coupled with the extensive tuning process in the early stages, which does not align with the current pursuit of low-carbon and environmentally friendly approaches. Considering all the discussions, it can be concluded that the RL model exhibits limited control performance, poses challenges during the training process, and is not suitable for solving the current control problem of the IPMSM in this thesis.
To facilitate understanding, development, and application of the RL model, this thesis starts based on the grid world problem. However, this may be the fundamental reason for the suboptimal performance of the RL model. On one hand, determining the reward function that enables the agent to explore the optimal operating points within the complete control range (MTPA and field-weakening) is challenging. On the other hand, the grid world is a discrete problem, while the current control of IPMSM is inherently a continuous problem. An ideal model should address continuous action space and state space. The former corresponds to i_d and i_q, while the latter corresponds to T_{ref}, w_{ref}, and u_{ref}. Both of these factors would significantly increase the complexity of the model, making the establishment and training of the RL model more challenging.

6
Conclusion

The increasing presence of electric vehicles has driven the rapid development of propulsion systems. To achieve a longer driving range with minimum environmental impact, energy efficiency plays a vital role in IPMSM control. Therefore, MTPA control and field-weakening control strategies have been introduced. In this thesis, based on these two control strategies, a basic LuT-based method was established, and an extended LuT-based method with improved control performance was developed by incorporating nonlinear parameters and additional dimensions. To overcome some inherent characteristics of the LuT-based method, such as large storage space and strong dependence on mathematical models and machine parameters, this thesis compares the NN-based model and RL-based model. The former uses data from the LuT as input and can accurately fit the LuT in a short time. It excels at handling large amounts of data and serves as a fast and easily implementable data optimization and reduction method. The RL-based model takes motor parameters as input and does not rely on data. Although implementing RL through a grid-world-like approach may not seem direct enough, the RL model in this thesis can essentially achieve MTPA control using a relatively simple method based on discrete values. Its current usability is questionable due to the excessively long training duration. But if given a different set of reward functions and model environment, the agent may be possible to find the optimal operating points on the complete control range, this is left for future studies. This model can provide some insights and inspiration for future RL-based solutions to similar problems.
In summary, this thesis transitions from traditional numerical-based methods to emerging artificial intelligence-based methods. While meeting the basic control requirements, the pursuit of higher accuracy, better performance, and more convenience still requires further improvement of machine learning algorithms. The utilization of more complex algorithms, such as deep reinforcement learning, may be advantageous in achieving better control of IPMSMs.

Bibliography

[1] Patrick, L. M., Bohlin, N., & Anderson, A. (1974). Three-point harness accident and laboratory data comparison. SAE Transactions, 3632-3676.

[2] Holland, M., 2023. Plugin EVs Take Record 75% Share Of Sweden's Auto Market, <https://cleantechnica.com/>.

[3] A. Das, A. Konwar, A. Dalal and P. Kumar, "Investigation of PMSM motor performance with different magnet configurations and rotor surface profiling," 2015 IEEE International Transportation Electrification Conference (ITEC), Chennai, India, 2015, pp. 1-6

[4] T. Martin and R. Burke, "Practical field weakening current vector control calculations for PMSM in vehicle applications," 2013 World Electric Vehicle Symposium and Exhibition (EVS27), Barcelona, Spain, 2013, pp. 1-7

[5] J. R. Jayalekshmi and S. Kumar P R, "Speed Tracking Performance of PMSM Using Sliding Mode and Extended State Observer," 2022 Second International Conference on Next Generation Intelligent Systems (ICNGIS), Kottayam, India, 2022, pp. 1-5

[6] Y. A.-R. I. Mohamed and T. K. Lee, "Adaptive self-tuning MTPA vector controller for IPMSM drive system," in IEEE Transactions on Energy Conversion, vol. 21, no. 3, pp. 636-644, Sept. 2006

[7] SHI M.,et al., Research of Permanent Magnet Synchronous Motor Control Stability Improving in Depth Flux-weakening Area. ELECTRIC DRIVE FOR LOCOMOTIVES, 2015(01):P.22-25

[8] J. Zhao, W. Liu and B. Tan, "Research of Maximum Ratio of Torque to Current Control Method for PMSM Based on Least Square Support Vector Machines," 2010 International Conference on Electrical and Control Engineering, Wuhan, China, 2010, pp. 1623-1628

[9] D. Hu and L. Xu, "Characterizing the torque lookup table of an IPM machine for automotive application," 2014 IEEE Conference and Expo Transportation Electrification Asia-Pacific (ITEC Asia-Pacific), Beijing, China, 2014, pp. 1-6

[10] T. M. Jahns, G. B. Kliman and T. W. Neumann, "Interior Permanent-Magnet Synchronous Motors for Adjustable-Speed Drives," in IEEE Transactions on Industry Applications, vol. IA-22, no. 4, pp. 738-747, July 1986

[11] X. Qian, G. Xiaorui, Q. Haihong, Z. Ying and D. Yaowen, "Research on the Application of Flux-Weakening Control in PMSM with Wide Range Speed Variation," 2017 International Conference on Smart Grid and Electrical Automation (ICSGEA), Changsha, China, 2017, pp. 371-374

[12] Liwei Song, Daqian Jiang, Shumei Cui and Shan Sheng, "Reluctance torque analysis and reactance calculation of IPM for HEVs based on FEM," 2010 IEEE Vehicle Power and Propulsion Conference, Lille, France, 2010, pp. 1-4

[13] H. Bai, X. Yan and W. Ouyang, "MTPA control of interior fault-tolerant permanent magnet motor for the marine electric propulsion system," 2021 6th International Conference on Transportation Information and Safety (ICTIS), Wuhan, China, 2021, pp. 267-272

[14] T. Sun, J. Wang, M. Koc and X. Chen, "Self-learning MTPA control of interior permanent magnet synchronous machine drives based on virtual signal injection," 2015 IEEE International Electric Machines & Drives Conference (IEMDC), Coeur d'Alene, ID, USA, 2015, pp. 1056-1062

[15] L. Jiang, Y. Wu, H. Qin and L. Wang, "An Improved Virtual Constant Signal Injection MTPA Control for PMa-SynRM Drives," 2022 5th International Conference on Power and Energy Applications (ICPEA), Guangzhou, China, 2022, pp. 144-149

[16] Dong, Weizhen. Artificial Intelligence and Machine Learning for Control and Operation of Electric Vehicles and Machine Drives. The University of Alabama, 2021.

[17] F. -J. Lin, M. -S. Huang, S. -G. Chen and C. -W. Hsu, "Intelligent Maximum Torque per Ampere Tracking Control of Synchronous Reluctance Motor Using Recurrent Legendre Fuzzy Neural Network," in IEEE Transactions on Power Electronics, vol. 34, no. 12, pp. 12080-12094, Dec. 2019

[18] J. Chen, J. Li and R. Qu, "Maximum-Torque-per-Ampere and Magnetization-State Control of a Variable-Flux Permanent Magnet Machine," in IEEE Transactions on Industrial Electronics, vol. 65, no. 2, pp. 1158-1169, Feb. 2018

[19] S. Bhattacharjee, S. Halder, Y. Yan, A. Balamurali, L. V. Iyer and N. C. Kar, "Real-Time SIL Validation of a Novel PMSM Control Based on Deep Deterministic Policy Gradient Scheme for Electrified Vehicles," in IEEE Transactions on Power Electronics, vol. 37, no. 8, pp. 9000-9011, Aug. 2022

[20] E. A. Grunditz, S. T. Lundmark, M. Alatalo, T. Thiringer and A. Nordelöf, "Three traction motors with different magnet materials — Influence on cost, losses, vehicle performance, energy use and environmental impact," 2018 Thirteenth International Conference on Ecological Vehicles and Renewable Energies (EVER), Monte Carlo, Monaco, 2018, pp. 1-13

[21] Kim, Do-Yun, and Jung-Hyo Lee. "Low cost simple look-up table-based PMSM drive considering DC-link voltage variation." Energies 13.15 (2020): 3904.

[22] Fitzgerald, A. E., Kingsley, C., & Umans, S. D. (2003). Electric machinery.

[23] James, G., Witten, D., Hastie, T. and Tibshirani, R., 2013. An introduction to statistical learning (Vol. 112, p. 18). New York: springer.

[24] Nasteski, V. (2017). An overview of the supervised machine learning methods. Horizons. b, 4, 51-62.

[25] Lion, K., Ian B. (2019). Official Pickle Use Documentation <https://docs.python.org/3/library/pickle.html#data-stream-format>

[26] Wikipedia contributors. (2023, May 31). Coefficient of determination. In Wikipedia, The Free Encyclopedia. Retrieved 09:26, June 7, 2023,

<https://en.wikipedia.org/w/index.php?title=Coefficient_of_determination&oldid=1157897397>

[27] International Energy Agency, World Energy Outlook (Paris: IEA Publications,2019).

[28] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. the Journal of machine Learning research, 12, 2825-2830.

[29] Tang, J., Deng, C., & Huang, G. B. (2015). Extreme learning machine for multilayer perceptron. IEEE transactions on neural networks and learning systems, 27(4), 809-821.

[30] Schmidt-Hieber, J. (2020). Nonparametric regression using deep neural networks with ReLU activation function.

[31] Svozil, D., Kvasnicka, V., & Pospichal, J. (1997). Introduction to multi-layer feed-forward neural networks. Chemometrics and intelligent laboratory systems, 39(1), 43-62.

[32] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

[33] Jeremy, Z., 2019. Reinforcement Learning — Implement Grid World, <https://towardsdatascience.com/reinforcement-learning-implement-grid-world-from-scratch-c5963765ebff>.

[34] Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.

[35] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). OpenAI gym. arXiv preprint arXiv:1606.01540.

DEPARTMENT OF SOME SUBJECT OR TECHNOLOGY
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY