KASDI MERBAH UNIVERSITY-OUARGLA

Faculty of New Technologies of Information and Communication

Department of Electronic and Telecommunication

Dissertation Academic Master

Domain: Electronic

Specialty: Embedded Electronic Systems

Submitted by

Koull Oussama **Benchenna Youcef**

Title

DTC-SVM Control of induction motor Fed by PWM inverter using neural Networks

Before the jury

Mr.	**BOUZIDI Mansour**	**MCA**	**UKMO**	**President**
Mr.	**BENHELLAL Belkhir**	**MCB**	**UKMO**	**Examiner**
Mr.	**TAMISSA Younes**	**PHD**	**UKMO**	**Supervisor**
Mr.	**KADRI FArid**	**MAA**	**UKMO**	**Co-Supervisor**

2024_2025

الإهداء

بسم الله الرحمن الرحيم

والحمد لله رب العالمين الذي منّ علينا بإتمام هذا العمل، والصلاة والسلام على سيدنا محمد وعلى آله وصحبه أجمعين.

إلى من كانوا نوراً في حياتي وسنداً في دربي..

إلى والديّ الحبيبين، اللذين بذلا الغالي والنفيس من أجلي، فالأب الذي سهر الليالي يعمل بجد ليوفر لي سبل العلم، والأم التي ضحت براحتها وصحتها لترعاني وتهتم بكل تفاصيل حياتي. إليكما أهدي هذا الجهد المتواضع، الذي ما كان ليرى النور لولا دعواتكما الصادقة وصبركما الذي لا يعرف الكلل.

إلى إخوتي الأعزاء، الذين كانوا سندي في لحظات الضيق، و مصدر فرحي في أوقات النجاح.

إلى معلميّ الأجلاء الذين بذلوا جهودهم لتنمية معارفي و توسيع آفاقي عبر مراحل تعليمي المختلفة.

إلى كل من ساهم في رحلتي العلمية، ولو بكلمة طيبة أو نصيحة غالية

أسأل الله أن يجعل هذا العمل خالصاً لوجهه الكريم، وسبباً في رضاكم وفرحتكم، وأن يجزيكم خير الجزاء على ما قدمتموه لي.

الشكر

أتقدم بخالص الشكر والعرفان إلى:
الأستاذ المشرف قادري فريد على توجيهاته القيمة التي كانت بمثابة المنارة التي أضاءت طريقي، لصبره الجميل وتفانيه في متابعة العمل، لنصائحه الثمينة التي ساهمت في إثراء هذا البحث.
الأستاذ المساعد تميسة يونس لدعمه المستمر ومساعدته القيمة، لملاحظاته البناءة التي ساعدت في تحسين العمل.
أعضاء لجنة المناقشة الأفاضل الأستاذ بوزيدي منصور والأستاذ بن هلال بلخير لما قدموه من ملاحظات قيمة وتقييم موضوعي.
زملائي وأصدقائي الأعزاء: الذين كانوا عوناً لي في الأوقات الصعبة، الذين شاركوني لحظات الفرح والنجاح، الذين قدموا لي الدعم المعنوي والعون الفكري.
عائلتي الكريمة: لوقفتهم الدائمة خلفي، لدعائهم الصادق الذي كان سندي، لصبرهم وتفهمهم خلال فترة إعداد البحث.
كل من ساهم في إنجاح هذا العمل: المعلمين الأجلاء منذ الصغر، والمصادر العلمية، كل من قدم لي نصيحة أو معلومة مفيدة.
وأخيراً.. أسأل الله أن يجعل هذا العمل خالصاً لوجهه الكريم، وأن ينفع به الأمة، وأن يجعله في ميزان حسنات كل من ساهم فيه.
والحمد لله رب العالمين

كل أسامه
بن شنة يوسف

Table of content

List of Figures

Chapter 1

Chapter 2

List of Tables

Chapter 1

Chapter 2

List of Abbreviations

ASM : Asynchronous Machine
IM : Induction Motor
DTC : Direct Torque Control
SVM : Space Vector Modulation
PWM : Pulse Width Modulation
PI : Proportional Integral
OLTF : Open Loop Transfer Function
CLTF : Closed Loop Transfer Function
RT : Transient Regime
Var : Variation
Inv : Inversion
Clsq : Classical
IA : Artificial Intelligence
RV : Speed Regulator
RF : Flux Regulator
RC : Torque Regulator
CC : Current Calculator
CT : Voltage Calculator
EC : Torque Estimator
EF : Flux Estimator
ANN : Artificial Neural Network
MLP : Multi-Layer Perceptron

List of Symbols

General Symbols:

- ω: Angular speed [rad/s]
- ω_ref: Reference angular speed [rad/s]
- ω_r: Rotor angular speed [rad/s]
- θ: Angle [rad]
- θ_s: Stator flux angle [rad]
- t: Time [s]
- T_m: Modulation period [s]
- T_e: Electromagnetic torque [Nm]
- T_ref: Reference torque [Nm]
- T_0, T_1, T_2: Switching times [s]
- V: Voltage [V]
- V_dc: DC bus voltage [V]
- V_d, V_q: d-q axis voltages [V]
- V_α, V_β: α-β axis voltages [V]
- V_s: Stator voltage vector [V]
- I: Current [A]
- I_s: Stator current [A]
- I_α, I_β: α-β axis currents [A]
- φ: Flux [Wb]
- φ_s: Stator flux [Wb]
- φ_ref: Reference flux [Wb]
- φ_sα, φ_sβ: α-β components of stator flux [Wb]

Machine Parameters:

- R_s: Stator resistance [Ω]
- R_r: Rotor resistance [Ω]
- L_s: Stator inductance [H]
- L_r: Rotor inductance [H]
- M: Mutual inductance [H]
- J: Moment of inertia [kg·m²]
- p: Number of pole pairs
- f: Frequency [Hz]
- F: Friction coefficient [N·m·s/rad]

Control Symbols:

- K_p: Proportional gain
- K_i: Integral gain
- ε: Damping coefficient
- η: Learning rate (ANN)
- σ: Leakage coefficient
- τ: Time constant [s]
- tr: constant time of rotor
- ts: constant time of stator
- S_a, S_b, S_c: Inverter switch states (0 or 1)
- SVM: Space Vector Modulation
- DTC: Direct Torque Control
- ANN: Artificial Neural Network

ANN-Specific Symbols:

- w_{ij}: Weight between neuron i and j
- b_j: Bias term for neuron j
- x_i: Input feature i
- z_j: Net input to neuron j
- a_j: Activation output of neuron j
- ϕ: Activation function
- E: Error function
- R: Correlation coefficient
- MSE: Mean Squared Error
- RMSE: Root Mean Squared Error

Vector Notation:

- $\bar{v}$: Vector quantity (e.g., $\bar{v} = v_\alpha + jv_\beta$)
- $|\bar{v}|$: Magnitude of vector
- $\angle\bar{v}$: Angle of vector

Subscripts/Superscripts:

- ref: Reference value
- est: Estimated value
- α, β: α-β frame components
- d, q: d-q frame components
- s: Stator-related
- r: Rotor-related
- (n): Layer n in ANN

General Introduction

Electric motors, particularly induction motors, have become an essential component in a wide range of industrial applications due to their simple construction, robustness, and cost-effectiveness. However, the complexity of their dynamic behavior poses significant challenges when precise control over torque and flux is required, especially in variable speed and load environments. Therefore, the development of advanced control strategies for induction motors remains an important area of research in modern electrical engineering.

Among the different control techniques, Direct Torque Control (DTC) has gained prominence for its fast dynamic response and relatively simple implementation. Unlike traditional field-oriented control (FOC), DTC directly controls the motor torque and stator flux without requiring complex transformations or coordinate decoupling. Despite its advantages, conventional DTC suffers from notable drawbacks such as high torque and flux ripples, variable switching frequency, and limited steady-state accuracy.[**1**]

To address these limitations, the DTC-SVM (Direct Torque Control with Space Vector Modulation) approach was introduced. By incorporating Space Vector Modulation, the switching frequency becomes more consistent, and the voltage vectors are better utilized, leading to smoother torque and flux waveforms. Nevertheless, DTC-SVM still relies heavily on precise tuning of controllers and accurate modeling of the motor, which can be difficult in real-world scenarios where system parameters may vary over time.[**2**]

In recent years, Artificial Neural Networks (ANNs) have emerged as a promising solution for intelligent control due to their ability to approximate nonlinear functions, learn from historical data, and adapt to changing system dynamics. Neural networks offer a flexible and data-driven approach that can potentially replace conventional controllers in complex control tasks.

This thesis proposes the integration of ANN-based regulators within the DTC-SVM control strategy of an induction motor powered by a PWM inverter. The primary objective is to enhance the overall control performance by reducing torque and flux ripples, minimizing overshoot, and improving response time, all while ensuring robustness against system uncertainties.[**3**]

To achieve this goal, the study begins with a detailed review of the fundamental principles of induction motor operation, DTC methodology, SVM techniques, and artificial neural networks. A hybrid control structure is then developed, where traditional torque and flux PI regulators are replaced by trained neural networks. These networks are trained using simulation datasets that represent the motor's dynamic behavior under various operating conditions.

Simulations are conducted in MATLAB/Simulink environment to validate the effectiveness of the proposed control scheme. Comparative performance analysis between the classical DTC-SVM and the ANN-enhanced version is performed based on key criteria such as rise time, overshoot, steady-state error, and ripple reduction.

Through this work, we aim to demonstrate that the integration of neural networks into DTC-SVM control not only simplifies the controller design process but also significantly enhances the dynamic and steady-state performance of induction motor drives. The findings of this study contribute to the broader effort of adopting machine learning techniques in advanced motor control systems.

This thesis is structured into two main chapters as follows:

- **Chapter 1** provides a comprehensive overview of Direct Torque Control (DTC) and its enhancement through Space Vector Modulation (SVM). It presents the theoretical foundation, system modeling, and control structure, as well as the implementation of the classical DTC-SVM approach using MATLAB/Simulink. Simulation results under various speed and load conditions are discussed to serve as a performance reference.
- **Chapter 2** introduces the integration of Artificial Neural Networks (ANNs) into the DTC-SVM control scheme. It explains the neural network architecture, training process, and simulation implementation. This chapter also compares the performance of the ANN-based DTC-SVM controller to the classical method, focusing on speed regulation, torque ripple, and stator flux behavior.

The thesis concludes with a summary of the main results and proposes potential directions for future research to improve intelligent control of induction motors.

Chapter 1

Direct Torque Control and Spatial Vector Modulation For Induction Motor

I.1. Introduction

The induction motor is a type of electric motor widely used in industrial and domestic applications. It uses a rotating magnetic field to drive a rotor without the use of direct electrical contact, making it more reliable and less susceptible to wear than DC motors [**4-5**].

Induction motor is called asynchronous because the rotor speed is not synchronized with the frequency of the alternating current that feeds the stator; this speed difference is known as slip and enables the motor to maintain high torque even at variable loads [**6**].

Induction motor is also economical in cost and maintenance, as it requires no brushes to transmit electrical energy to the rotor. It can be used in a wide variety of applications, from small electromechanical devices to large industrial machines [**5,7**].

Direct torque control (DTC) and space vector modulation (SVM) are two advanced asynchronous motor control techniques popular for their high performance [**8**]. The DTC-SVM combines these techniques to provide asynchronous motor control with excellent torque, speed, and stability performance. It also enables dynamic response to system disturbances and precise speed and torque control, useful in industrial applications such as transportation systems, machine tools, and wind power generation [**7,9-10**].

I.2. Induction Motor Constitution

The induction motor consists of a fixed stator and a rotating rotor (**Figure I.1**). Unlike synchronous and DC machines, only the stator windings are connected to a power supply, whose voltages define the magnetic state of the air gap. The rotor windings are self-connected. The induction motor has no excitation windings or permanent magnets. The rotor flux required to generate electromagnetic torque is produced from induction [**11-12**]. Mechanically, the induction motor is subdivided into:

- **Stator**: the stationary part where the power supply is connected.
- **Rotor**: the rotating part.
- **Bearings**: support the motor shaft.
- **Air gap**: non-magnetic gap between rotor and stator [**13**].

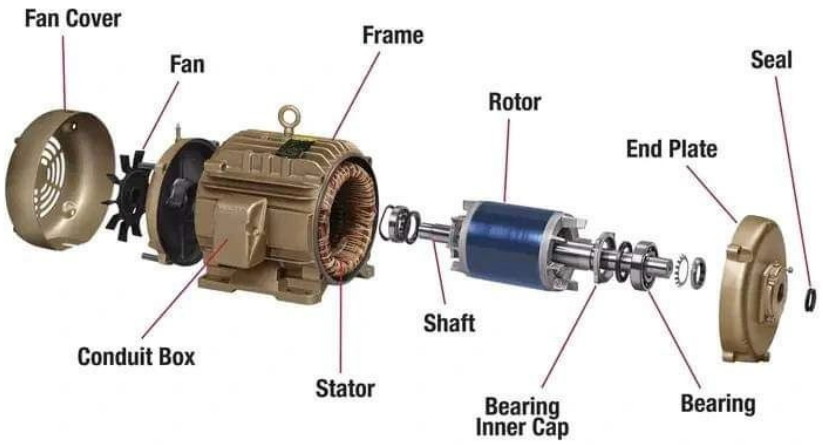


Figure I.1 Architecture of Induction Motor

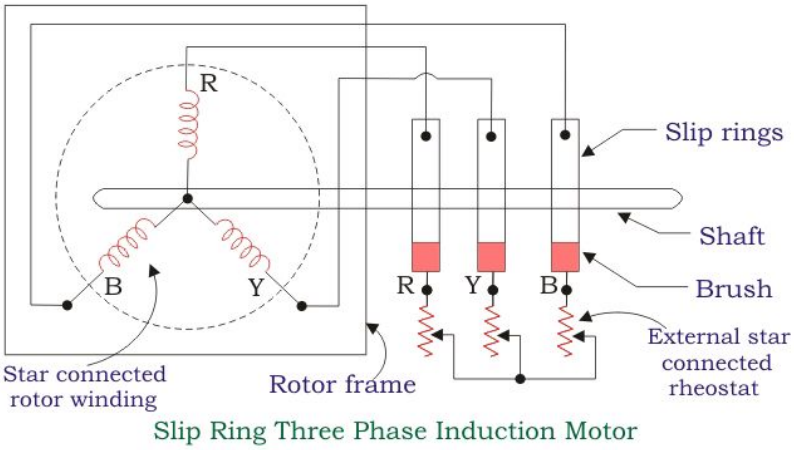


Figure I.2 Slip Ring Three Phase Induction Motor

I.3. General Principle of DTC Control

Direct Torque Control (DTC) is based on the stator flux's orientation, eight basic instantaneous voltage vectors, two of which are zero. These vectors are selected from a switching table based on the errors of the flux and torque and the position of the stator flux vector. In this technique, the rotor position is no longer required to choose the voltage vector. This particularity defines DTC as a method well-suited for sensorless control of alternating current machines. [**14-24**]

I.4. General Structure of a DTC Control

For a DTC control:

- The stator flux and torque are calculated.

- The reference torque is calculated from the reference speed and the actual speed of the motor.
- The amplitude of the reference stator flux and the torque are compared to the estimated flux and torque, and the flux error and torque error are sent to the hysteresis regulator.

The general structure of a DTC control is illustrated in **Figure I.3**:

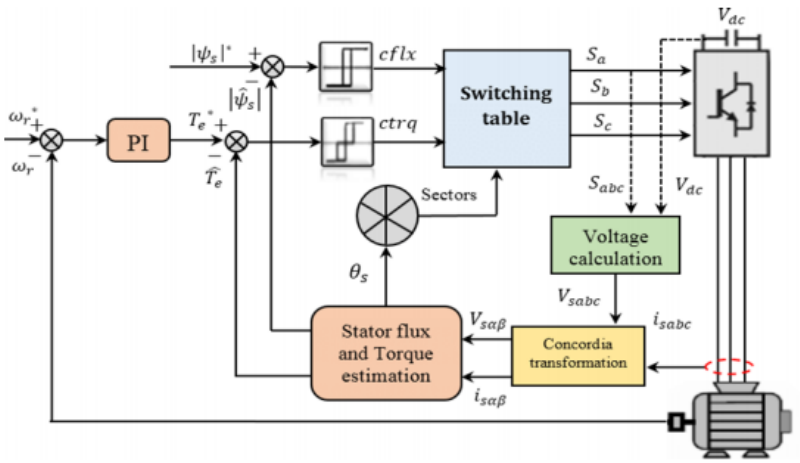


Figure I.3 General structure of DTC control.

I.5. Principle of DTC-SVM Control

The DTC-SVM control is an advanced method for electric motor control that combines two techniques: Direct Torque Control (DTC) and Space Vector Modulation (SVM). [**7,25**]

The principle of DTC is to control the torque and speed of the motor in real-time by using a switching table to select the best switching state of the power converter to produce the desired torque. This control method allows for a fast and precise motor response but can also lead to significant switching losses and torque ripples. [**26**]

The SVM technique is used to minimize switching losses and improve the energy efficiency of the system. This technique employs a nonlinear classification algorithm to determine the optimal voltage vectors to apply to the motor phases, depending on the motor's load and speed. This optimizes the system's energy efficiency while reducing switching losses and minimizing torque ripples. [**14,27-28**]

I.6. Structure of DTC-SVM Control

This technique retains the basic idea of the classical DTC control method. The control voltages can be generated by the stator flux and torque PI regulators and imposed by the PWM vector (SVM). [**29**]

The block diagram of the DTC-SVM control technique applied to the induction machine is shown in **Figure I.4**. In this structure, two PI regulators are used for torque and stator flux control instead of hysteresis regulators (as in the classical DTC control

structure). These regulators calculate the required reference voltage components, V_{d_ref} and V_{q_ref}, in the (d-q) reference frame. These components are then transformed into the stationary reference frame (α-β), and the resulting components $\mathbf{V}_{\alpha,\mathbf{ref}}$ and $\mathbf{V}_{\beta,\mathbf{ref}}$ are fed into the SVM vector modulation block, which in turn generates the inverter control signals. [**30**]

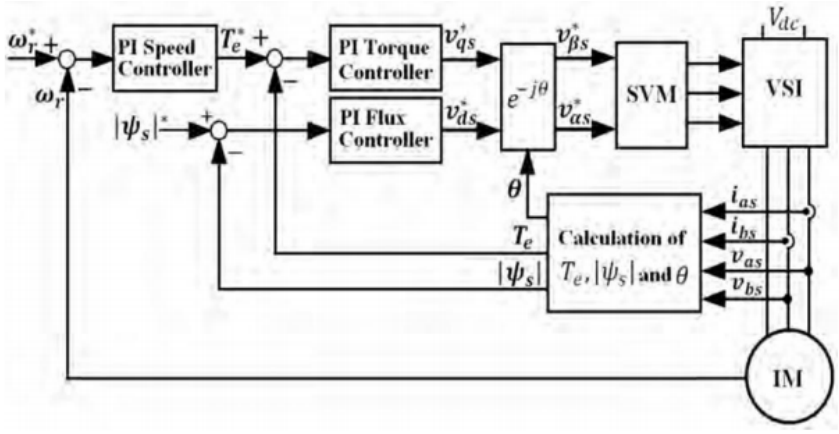


Figure I.4 DTC-SVM control structure for induction motor. [**31**]

I.6.1. Voltage Source of the Inverter

Figure I.5 represents the voltage source of the inverter that supplies the three-phase induction machine. The inverter converts direct current (DC) into alternating current (AC) using power electronics. The circuit is activated by switching S_a, S_b, S_c. The inverter uses two pairs of complementarily controlled switches in each phase or leg of the inverter, as shown in **Figure I.5**. It is assumed that the two switches in each phase or leg of the inverter operate in a balancing pair to avoid short-circuiting the DC source. [**7,31**]

Table I.1 shows the different states of the inverter and the coordinates of the output voltage vector $\mathbf{v}_s$ corresponding to each state in the (α, β) reference frame. [**4**]

Table I.1Inverter states and coordinates of vector V_s in the (α, β) plane

S_a	S_b	S_c	v_α	v_β	v_i
0	0	0	0	0	v_0
1	0	0	$\sqrt{\frac{2}{3}}v_{dc}$	0	v_1
1	1	0	$\sqrt{\frac{1}{6}}v_{dc}$	$\sqrt{\frac{1}{2}}v_{dc}$	v_2
0	1	0	$-\sqrt{\frac{1}{6}}v_{dc}$	$\sqrt{\frac{1}{2}}v_{dc}$	v_3
0	1	1	$-\sqrt{\frac{2}{3}}v_{dc}$	0	v_4
0	0	1	$-\sqrt{\frac{1}{6}}v_{dc}$	$\sqrt{\frac{1}{2}}v_{dc}$	v_5
1	0	1	$\sqrt{\frac{1}{6}}v_{dc}$	$-\sqrt{\frac{1}{2}}v_{dc}$	v_6
1	1	1	0	0	v_7

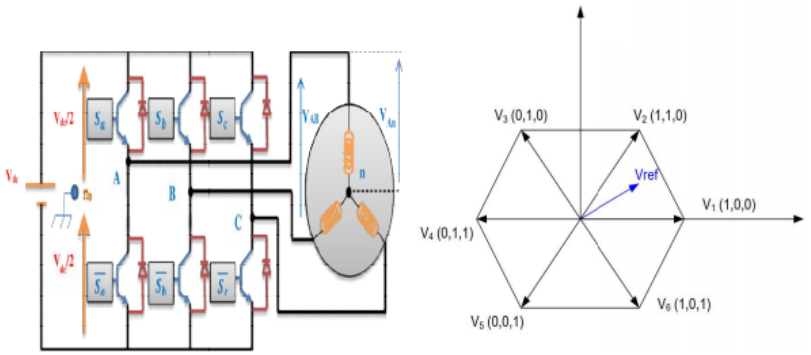


Figure I.5 Voltage source of the inverter.

The inverter is controlled using Boolean logic. The switch state is: $S_i = 1$ $(i = a, b, c)$: $S_i = 1$ when the upper switch is closed and the lower switch is open.

$S_i = 0$: When the upper switch is open and the lower switch is closed. The stator voltage vector $\mathbf{V}_s$ can be written as: [**32**]

$$\mathbf{V}_s = \sqrt{\frac{2}{3}} V_{dc} \left[S_a + S_b \exp\left(j\frac{2\pi}{3}\right) + S_c \exp\left(j\frac{4\pi}{3}\right) \right] \tag{1.1}$$

The different combinations of the three values (S_a, S_b, S_c) generate eight positions including two zero vectors ($\mathbf{V}_0$ and $\mathbf{V}_7$). Here, (S_a, S_b, S_c) represent the logical states of the three switches. We aim to control the flux and torque through the selected voltage vector, which is determined by the switch configuration. With three switches, there are eight possible configurations, including two null vectors.

I.6.2. Flux and Torque Estimation

I.6.2.1 Stator Flux Estimator

The flux can be calculated from measurements of the stator current and voltage of the machine. [**33-34**]

From the equation:

$$\overline{\phi}_s = \int_0^t (\mathbf{V}_s - R_s \mathbf{I}_s) dt \tag{1.2}$$

We obtain the α and β components of the flux vector $\overline{\phi}_s$:

$$\overline{\phi}_s = \phi_{s\alpha} + j\phi_{s\beta} \tag{1.3}$$

$$\begin{aligned} \phi_{s\alpha} &= \int_0^t (V_{s\alpha} - R_s I_{s\alpha}) dt \quad \text{and} \\ \phi_{s\beta} &= \int_0^t (V_{s\beta} - R_s I_{s\beta}) dt \end{aligned} \tag{1.4}$$

The stator flux magnitude is given by:

$$\phi_s = \sqrt{\phi_{s\alpha}^2 + \phi_{s\beta}^2} \tag{1.5}$$

The flux angle is given by:

$$\theta_s = \arctan\left(\frac{\phi_{s\beta}}{\phi_{s\alpha}}\right) \tag{1.6}$$

The stator voltages $V_{s\alpha}$ and $V_{s\beta}$ in the (α-β) reference frame are determined as follows:

$$\begin{aligned} V_{s\alpha} &= \sqrt{\frac{2}{3}} V_{dc}\left(S_a - \frac{1}{2}(S_b + S_c)\right) \quad \text{and} \\ V_{s\beta} &= \frac{1}{\sqrt{2}} V_{dc}(S_b - S_c) \end{aligned} \tag{1.7}$$

I.6.2.2 Electromagnetic Torque Estimation

The electromagnetic torque can be calculated from the estimated flux amplitudes $\phi_{s\alpha}$, $\phi_{s\beta}$ and calculated current amplitudes $I_{s\alpha}$, $I_{s\beta}$ using the following equation:

$$T_e = \frac{3}{2} p(I_{s\beta}\phi_{s\alpha} - I_{s\alpha}\phi_{s\beta}) \tag{1.8}$$

where p is the number of pole pairs of the induction motor.[**33**]

The measured currents (I_a, I_b, I_c) can be transformed into a two-dimensional vector ($I_{s\alpha}$, $I_{s\beta}$) by:

$$I_{s\alpha} = \frac{\sqrt{2}}{3}\left(I_a - \frac{1}{2}I_b - \frac{1}{2}I_c\right) \quad \text{and} I_{s\beta} = \frac{\sqrt{2}}{3}\left(\frac{\sqrt{3}}{2}I_b - \frac{\sqrt{3}}{2}I_c\right) \tag{1.9}$$

I.7 Implementation of SVM Blocks

The PWM (SVM) block can be implemented by following these steps:

- Reference voltage determination
- Sector determination
- Calculation of inverter state application times
- Calculation of switching cycles for each sector

I.7.1 Reference Voltage Determination

The outputs of the PI controllers are voltages in the (d-q) reference frame. A transformation is applied between this frame and the (α-β) reference frame:

$$\begin{aligned} V_\alpha &= V_d\cos(\theta) - V_q\sin(\theta) \\ V_\beta &= V_d\sin(\theta) + V_q\cos(\theta) \end{aligned} \tag{1.10}$$

with the reference voltage magnitude:

$$V_{ref} = \sqrt{V_\alpha^2 + V_\beta^2} \tag{1.11}$$

The reference vector is evaluated over the modulation period by generating an average vector determined by applying adjacent inverter command vectors and null vectors. We use all eight available vectors. [**31**]

I.7.2 Sector Determination

From the coordinates V_α, V_β, and the angular position θ, we determine in which sector (S) and region (R) the vector is located in the (α-β) plane:

$$(N-1)\frac{\pi}{3} \leq \theta_N \leq N\frac{\pi}{3}, \quad N = 1, \ldots, 6 \tag{1.12}$$

1.7.2.1 Calculation of Inverter State Application Times

The determination of times T1 and T2 is given by a simple projection on the α and β axes (**Figure I.6**)[**31**]:

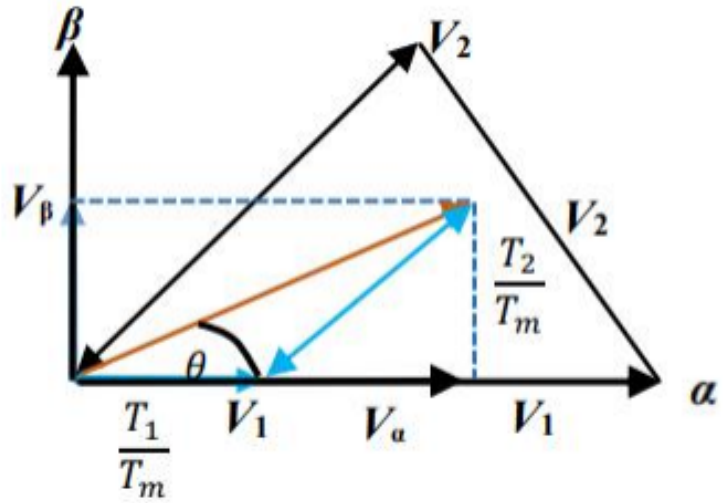


Figure I.6 Projection of the reference vector on α-β axes

$$\overline{V_{ref}} = \frac{1}{T_m}\left(T_1 \cdot V_1 + T_2 \cdot V_2 + T_3 \cdot V_{(0,7)}\right) \tag{1.13}$$

$$\overline{V_{ref}} = \frac{T_1}{T_m} \cdot V_1 + \frac{T_2}{T_m} \cdot V_2 + \frac{T_3}{T_m} \cdot V_{(0,7)} \tag{1.14}$$

$$|V_{ref}| \cdot \begin{bmatrix}\cos(\theta)\\ \sin(\theta)\end{bmatrix} = \frac{T_1}{T_m} \cdot \sqrt{\frac{2}{3}}V_{dc}\begin{bmatrix}1\\0\end{bmatrix} + \frac{T_2}{T_m} \cdot \sqrt{\frac{2}{3}}V_{dc}\begin{bmatrix}\cos\left(\frac{\pi}{3}\right)\\ \sin\left(\frac{\pi}{3}\right)\end{bmatrix} \tag{1.15}$$

$$\begin{aligned}\frac{T_1}{T_m} &= \frac{|V_{ref}|}{\frac{2}{3}V_{dc}} \cdot \frac{\sin\left(\frac{\pi}{3} - \theta\right)}{\sin\left(\frac{\pi}{3}\right)}\\ \frac{T_2}{T_m} &= \frac{|V_{ref}|}{\frac{2}{3}V_{dc}} \cdot \frac{\sin(\theta)}{\sin\left(\frac{\pi}{3}\right)}\end{aligned} \tag{1.16}$$

$$\frac{T_3}{T_m} = 1 - \left(\frac{T_1 + T_2}{T_m}\right) \tag{1.17}$$

I.7.3 Calculation of switching cycles for each sector

the calculate of cycles rapports

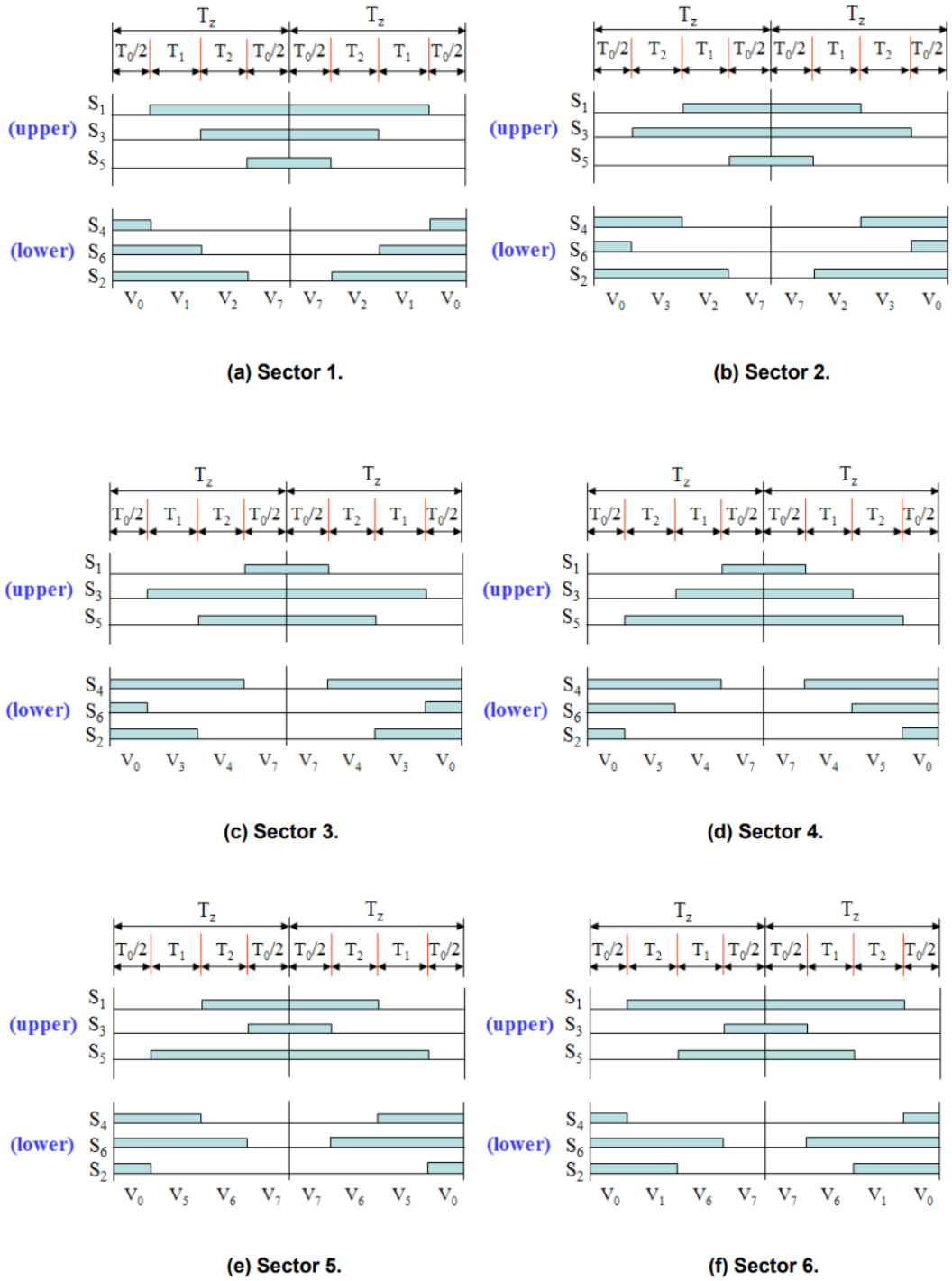


Figure I.7 Complete set of switching patterns and voltage vectors

Table I.2 Conduction time of the upper switches (S1, S3, S5) by sector

Secteur	**Commutateur haut** S_1, S_3, S_5
1	$S_1 = T_1 + T_2 + \frac{T_0}{2}$ $S_3 = T_2 + \frac{T_0}{2}$ $S_5 = \frac{T_0}{2}$
2	$S_1 = T_2 + \frac{T_0}{2}$ $S_3 = T_1 + T_2 + \frac{T_0}{2}$ $S_5 = \frac{T_0}{2}$
3	$S_1 = \frac{T_0}{2}$ $S_3 = T_1 + T_2 + \frac{T_0}{2}$ $S_5 = T_2 + \frac{T_0}{2}$
4	$S_1 = \frac{T_0}{2}$ $S_3 = T_1 + \frac{T_0}{2}$ $S_5 = T_1 + T_2 + \frac{T_0}{2}$
5	$S_1 = T_2 + \frac{T_0}{2}$ $S_3 = \frac{T_0}{2}$ $S_5 = T_1 + T_2 + \frac{T_0}{2}$
6	$S_1 = T_1 + T_2 + \frac{T_0}{2}$ $S_3 = \frac{T_0}{2}$ $S_5 = T_2 + \frac{T_0}{2}$

I.8 PI Controller Development

I.8.1 PI Speed Controller's Development

In order to avoid undesired load changes, speed control is a crucial industrial necessity. We employ a (PI) corrector for this closed-loop control, which enhances both transient and steady-state speed response by combining proportional and integral action. The PI speed controller's block diagram is displayed in **Figure I.8**.[**31,34**]

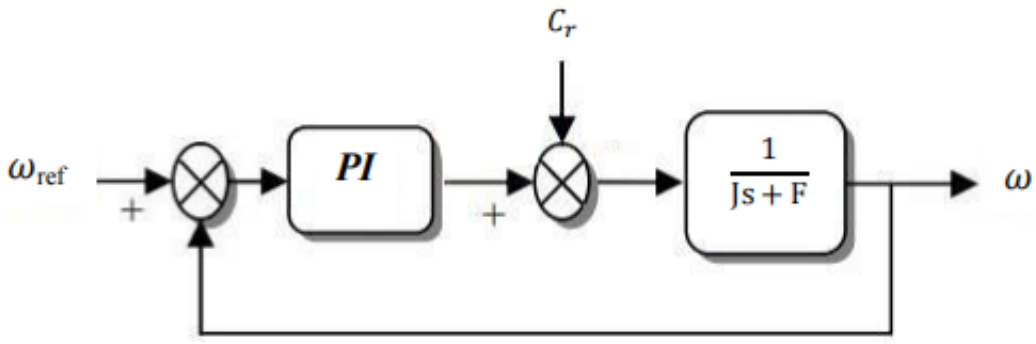


Figure I.8 Conventional speed controller

The equation in the time domain for this model is given as follows:

$$\mathbf{u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau} \tag{1.18}$$

Where $e(t)$, $u(t)$, K_p, and K_i respectively represent the error at time t, the generated control signal, and the gains of the controller.

The corresponding transfer function is given by:

$$PI(s) = K_p + \frac{K_i}{s} = K_p \left(1 + \frac{1}{\tau s}\right) \tag{1.19}$$

Where s is the Laplace operator, and $\tau = \frac{K_p}{K_i}$ is the time constant.

The closed-loop transfer function (for $C_r = 0$) is given by:

$$CLTF = \frac{PI(s)\frac{1}{Js+f}}{1 + PI(s)\frac{1}{Js+f}} \tag{1.20}$$

From[**31,35**], we have:

$$G(s) = \frac{1}{Js + f} \tag{1.21}$$

j : Motor factor

f : Friction coefficient

Substituting (**equation 1.19**) into (**equation 1.20**), with $C_r = 0$, after simplification we obtain:

$$CLTF = \frac{(1 + \tau s)}{\frac{J}{K_i} s^2 + \left(\frac{f + K_p}{K_i}\right) s + 1} \tag{1.22}$$

To control the closed-loop system, choosing the coefficients K_p and K_i is necessary. In this case, the pole placement method is used. The transfer function of a second-order closed-loop system is characterized by:

$$H(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{1.23}$$

The characteristic equation is:

$$1 + \frac{2\varepsilon}{\omega_n}S + \frac{1}{\omega_n^2}S^2 \tag{1.24}$$

Where: ε is the damping coefficient and ω_n the natural angular frequency of the system. By identifying (**equation 1.23**) we will have the system as follows:

$$\frac{1}{\omega_n^2} = \frac{J}{K_i} \Rightarrow K_i = J\omega_n^2 \tag{1.25}$$

$$\frac{2\varepsilon}{\omega_n} = \frac{K_p + f}{K_i} \Rightarrow K_p = \frac{2\varepsilon K_i}{\omega_n} - f \tag{1.26}$$

The controller gains are obtained to achieve a minimal response time while ensuring no overshoot. This technique consists in imposing values of damping and pulsation ε and ω_n to determine the coefficients K_p and K_i.

I.8.2 Design of the PI flux controller

Thus, the stator flux can be controlled by the d-component of the stator voltage. (**Figure I.9**)) shows the relationship between Φ_s and V_{ds}, an equivalent second-order system with a perturbation E_d. A PI controller allows achieving the desired performances and maintaining the stator flux at its reference value Φ_{sref}[**31**].

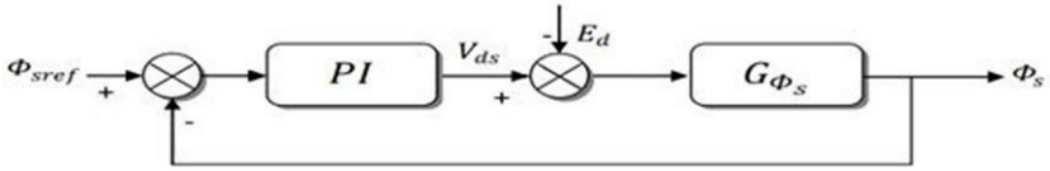


Figure I.9 Conventional flux controller

The PI transfer function is given by:

$$C(s) = K_{p\phi} + \frac{K_{i\phi}}{s} = K_{p\phi}\left(1 + \frac{1}{\tau_\phi s}\right) = K_{p\phi}\frac{(1+\tau_\phi s)}{\tau_\phi s} \tag{1.27}$$

Where $K_{p\phi}$, $K_{i\phi}$ respectively denote the proportional and integral gains of the controller, and $\tau_\phi = \frac{K_{p\phi}}{K_{i\phi}}$ is a time constant of the flux.

The open-loop transfer function (OLTF) is given by:

$$OLFT = C(s) \cdot G_{\phi s} \tag{1.28}$$

$$OLTF = K_{p\phi}\frac{(1+\tau_\phi s)}{\tau_\phi s} \cdot \frac{t_s(1+\tau_r s)}{1 + (t_r + t_s)s + \sigma t_r t_s s^2} \tag{1.29}$$

Using the `roots` command in MATLAB, the roots of the polynomial

$$1 + (t_r + t_s)s + \sigma t_r t_s s^2 \tag{1.30}$$

can be found as follows:

$$P_1 = f(t_r, t_s, \sigma) \tag{1.31}$$

$$P_2 = f(t_r, t_s, \sigma) \tag{1.32}$$

By eliminating the dominant pole, which is the pole closest to the secondary axis (let it be pole P_2), we can write the (**equation 1.29**) in the "pole-zero" form as follows:

$$OLTF = K_{p\phi}\left(\frac{1}{\tau_\phi}s + 1\right) \cdot \frac{\sigma t_r t_s(1 + \sigma t_r t_s)}{(s - P_1)(s - P_2)} \tag{1.33}$$

We have from[**31,35**]:

$$G_{\phi S} = \frac{\sigma t_r t_s(1 + \sigma t_r t_s)}{(s - P_1)(s - P_2)} \tag{1.34}$$

To eliminate the dominant pole, we set:

$$(s - P_2) = \left(\frac{1}{\tau_\phi} + s\right) \tag{1.35}$$

Thus, we have:

$$OLTF = \frac{K_{p\phi}}{s} \cdot \frac{t_s(1 + \sigma t_r t_s)}{s - P_1} \tag{1.36}$$

The closed-loop transfer function (CLTF) is written as:

$$CLTF = \frac{OLTF}{1 + OLTF} \tag{1.37}$$

Substituting (**equation 1.36**) into (**equation 1.37**) and simplifying, we get:

$$CLTF = \frac{K_{p\phi} t_s(1 + \sigma t_r t_s)}{s^2 + (K_{p\phi}\sigma t_r - P_1)\, s + K_{p\phi} t_s} \tag{1.38}$$

To control the closed-loop system, it is necessary to select the coefficients $K_{p\phi}$ and $K_{i\phi}$. For this, we use the pole placement method. The transfer function of a standard second-order system is characterized by:

$$F(s) = \frac{\omega_n^2}{s^2 + 2\epsilon\omega_n s + \omega_n^2} \tag{1.39}$$

Where ϵ and ω_n are the damping ratio and the natural frequency of the system. By comparing expressions (**equation 1.38**) and (**equation 1.39**) and considering (**equation 1.35**), we find:

$$(s - P_2) = \left(s + \frac{1}{\tau_\phi}\right) \Rightarrow \frac{1}{\tau_\phi} = -P_2 \Rightarrow K_{i\phi} = -K_{p\phi} P_2 \tag{1.40}$$

$$2\epsilon\omega_n = (K_{p\phi}\sigma t_r - P_1) \Rightarrow K_{p\phi} = \frac{2\epsilon\omega_n + P_1}{\sigma t_r} \tag{1.41}$$

Thus, the controller gains are obtained to ensure a minimum response time while also providing proper damping. This technique involves imposing damping and pulsation values to determine the coefficients $K_{p\phi}$ and $K_{i\phi}$.

I.8.3 Development of PI torque

Just as torque can be controlled by the stator frequency, there exists a relationship between T_e and ω_n. A PI controller makes it possible to achieve the desired performance and maintain the reference torque value $T_{e_{ref}}$[**31**].

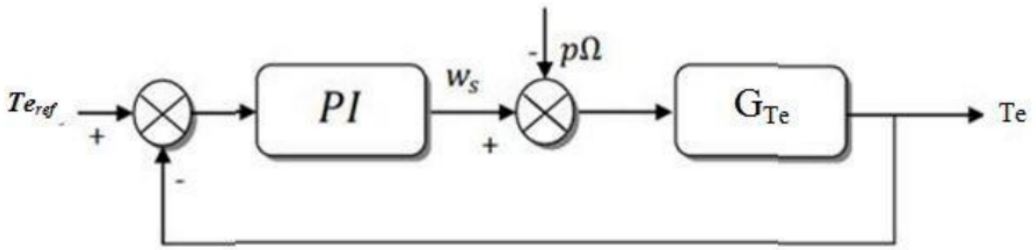


Figure I.10 Conventional torque controller

The transfer function of the PI controller is given by:

$$C(s) = K_{pTe} + \frac{K_{iTe}}{s} = K_{pTe}\left(1 + \frac{1}{t_{iTe}s}\right) = K_{pTe} \cdot \frac{(1 + t_{rTe}s)}{t_{rTe}s} \tag{1.42}$$

Where K_{pTe} and K_{iTe} represent the proportional and integral gains of the controller, and $t_{rTe} = \frac{K_{pTe}}{K_{iTe}}$ is the time constant.

The closed-loop transfer function (CLTF) is given by:

$$CLTF = \frac{C(s) \cdot G_{Te}}{1 + C(s) \cdot G_{Te}} \tag{1.43}$$

We have from [**35**]:

$$G_{Te} = p \cdot \frac{t_r(1-\sigma)}{L_s} \cdot \frac{\phi^2}{(1 + 2\sigma t_r s)} \tag{1.44}$$

Substituting equations (**equation 1.44**) and (**equation 1.42**) into (**equation 1.43**), and after simplification, we obtain:

$$CLTF = \frac{K_{iTe}Pt_r(1+\sigma) + \frac{2}{s}(1 + t_{rTe}s)}{2\sigma t_r L_s s^2 + (K_{iTe}t_{rTe}Pt_r(1-\sigma)\phi^2 + L_s)s + K_{iTe}Pt_r(1-\sigma)\phi^2} \tag{1.45}$$

$$CLTF = \frac{1 + t_{rTe}s}{\frac{2\sigma t_r L_s}{K_{iTe}Pt_r(1-\sigma)\phi^2}s^2 + \frac{(K_{iTe}t_{rTe}Pt_r(1-\sigma)\phi^2 + L_s)}{K_{iTe}Pt_r(1-\sigma)\phi^2}s + 1} \tag{1.46}$$

The CLTF in the form of a second-order system is then characterized as:

$$F(s) = \frac{K}{\frac{s^2}{\omega_n^2} + \frac{2\xi}{\omega_n}s + 1} \tag{1.47}$$

Identification by equations (**equation 1.46**) and (**equation 1.47**) will give:

$$\frac{1}{\omega_n^2} = \frac{2\sigma T_r L_s}{K_i T_e P_r(1-\sigma)\phi_s^2} \implies K_{iTe} = \frac{2\sigma T_r \omega_n^2 L_s}{P_r(1-\sigma)\phi_s^2} \tag{1.48}$$

$$\frac{2\xi}{\omega_n} = \frac{K_{iTe}P_r(1-\sigma)\phi_s^2 + L_s}{K_i T_e P_r(1-\sigma)\phi_s^2} \implies \tau_{Te} = \frac{K_{pTe}}{K_{iTe}} = \frac{2\xi}{\omega_n} - \frac{L_s}{K_{iTe}P_r(1-\sigma)\phi_s^2} \tag{1.49}$$

The rotor and stator time constants are defined as $t_r = \frac{L_r}{R_r}$ and $t_s = \frac{L_s}{R_s}$, respectively. The leakage coefficient is given by $\sigma = 1 - \frac{M^2}{L_s L_r}$.

The technique always consists of imposing the values of gains and pulsations and damping coefficients to determine the coefficients K_{iTe} and K_{pTe}.

I.8.4 Parameters of the PI controller for DTC-SVM

The parameters of the different controllers after calculation are given as follows:

- Speed PI controller: $K_p = 1.5$, $K_i = 20$
- Torque PI controller: $K_p = 10$, $K_i = 100$
- Flux PI controller: $K_p = 8000$, $K_i = 3500$

I.9 SIMULINK Model of DTC-SVM Control Structure

(**Figure I.11**) shows the block diagram of the DTC-SVM control structure under Matlab Simulink.

The block diagram shows the following blocks:

- Three-Phase Asynchronous Motor block.
- Voltage Inverter block.
- Continuous Power Supply block.
- Control Variable Estimator block.
- Command Generation block.

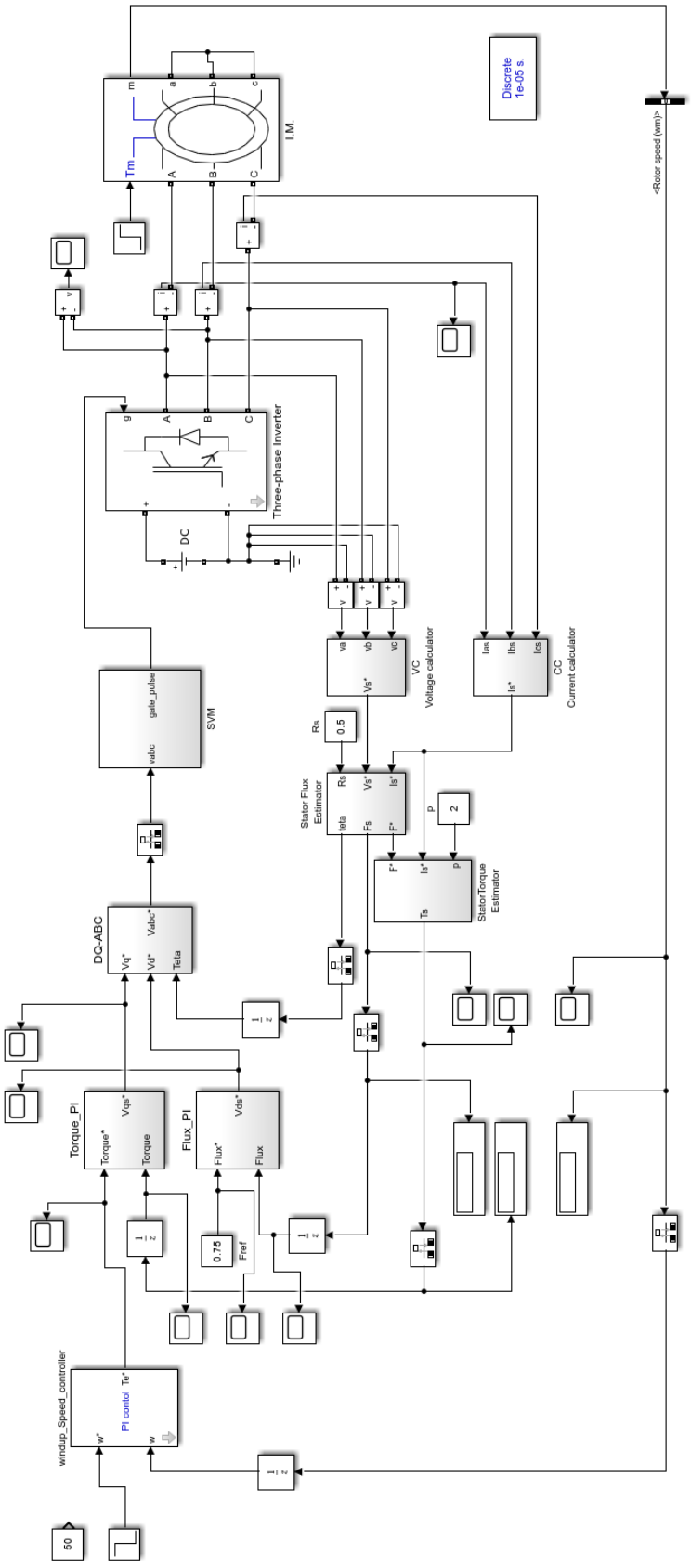


Figure I.11 SIMULINK Diagram for Classical DTC-SVM

I.9.1 The Three-Phase Asynchronous Motor Block

This is a pre-established block that corresponds to the induction motor to be regulated. The properties of the motor used are presented in (**Table I.3**) as follows:

Table I.3 Properties of the Asynchronous Machine

F (Hz)	P (kW)	p	V (V)	Rs (Ω)	Rr (Ω)	Ls=Lr (H)	M (H)	J (kgm^2)
50	1.5	2	280	0.5	0.5	0.005	0.1	0.002

I.9.2 The Voltage Inverter and Continuous Power Supply Blocks

The DC power supply provides power to the voltage inverter. The voltage inverter block, represented in **Figure I.12**, is a block that we have constructed and corresponds to a three-phase half-bridge inverter, using MOSFETs for the switching.

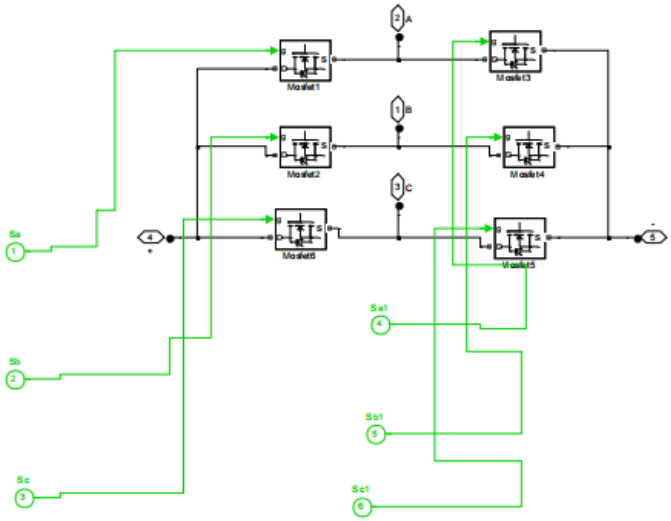


Figure I.12 SIMULINK model of inverter block

I.9.3 The Control Variable Estimator Block

The control variable estimator block plays an essential role in the system. It allows for the precise estimation of the electromagnetic torque as well as the amplitude and position of the stator flux vector. This estimation is crucial for the proper functioning of the system and to ensure optimal performance.

It is composed of the following constructed blocks:

- *CC (Current Calculator)*: calculator of the current vector.
- *VC (Voltage Calculator)*: calculator of the voltage vector.
- *TE (Torque Estimator)*: estimator of the electromagnetic torque.
- *FE (Flux Estimator)*: estimator of amplitude and position of the stator flux vector.

I.9.4 The Command Generation Block

The command generation block allows for the comparison of the reference values with those that have been estimated. This comparison makes it possible to determine the voltage vector to apply to the voltage inverter, following an implemented algorithm. The precision of this step is essential to guarantee the proper functioning of the system and to ensure effective control of the motor's electrical variables.

It is composed of the following blocks:

a) *SC (Speed Controller)*: The speed PI controller is a PI controller used to control the speed of a system by using the speed error signal.

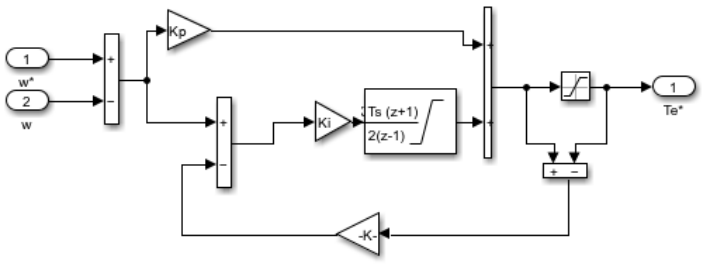


Figure I.13 SIMULINK model of speed regulator

b) FR (Flux Regulator): The PI flux regulator for an asynchronous motor adjusts the frequency of the electrical supply to control the motor flux.

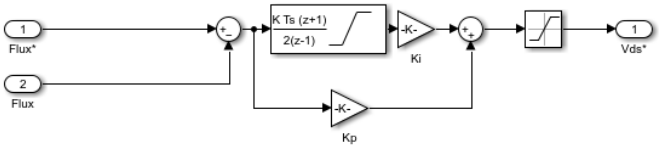


Figure I.14 SIMULINK model of flux regulator

c) TR (Torque Regulator): The PI torque regulator for an asynchronous motor adjusts the frequency of the electrical supply to control the motor torque.

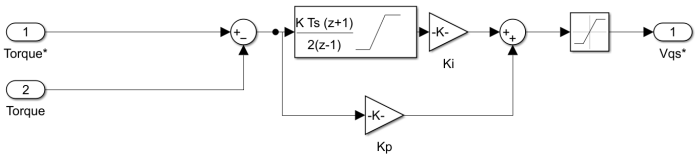


Figure I.15 SIMULINK model of torque regulator

d) SVM Block: The SVM block is used to generate a control signal for the inverter that powers the machine.

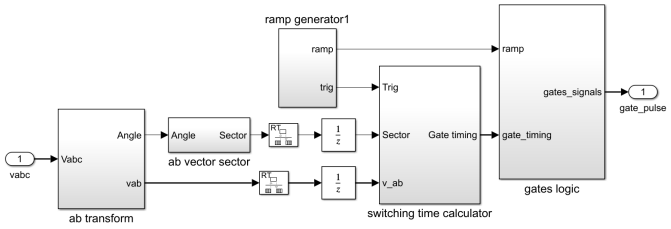


Figure I.16 SIMULINK model of SVM block

I.10 Simulation Results Presentation

To enrich the theoretical study presented previously, this section details the results of numerical simulations demonstrating the operation of the Direct Torque Control with Space Vector Modulation (DTC-SVM) structure applied to an induction motor model, equipped with a two-level inverter.

The dynamic behavior of the system, from startup through transient phases to its steady-state operation, is observed. We analyze the system's performance until it stabilizes.

To evaluate the effectiveness of the DTC-SVM control, two distinct tests were conducted:

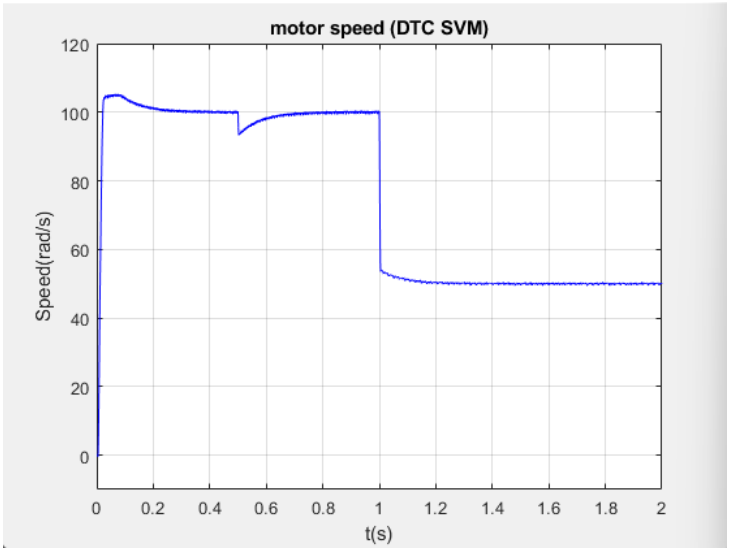
- **Reference Speed Variation Test with Load Torque:** This test assesses the system's ability to track a change in speed reference while under a constant load.
- **Reference Speed Reversal Test with Load Torque:** This test examines the system's robustness and dynamic performance during a complete reversal of the speed reference, also under a load.

For each test, the following key system variables are visualized: motor speed, stator flux, electromagnetic torque, phase-to-phase voltage (between phases 'a' and 'b'), stator current in the alpha-beta frame, and stator current of phase 'a'. The simulation results for each test are presented in the subsequent subsections.

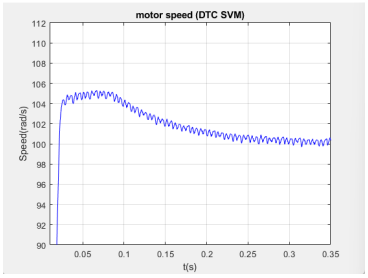
I.10.1 Speed Variation Test with Load Torque (100 rad/s to 50 rad/s)

In this test, the motor is initially accelerated to 100 rad/s. A constant load torque of 10 Nm is introduced at $t = 0.5$ s, and then the speed reference is changed to 50 rad/s at $t = 1.0$ s. This scenario allows for the evaluation of the controller's ability to maintain stable operation and track speed commands under varying load conditions and speed references.

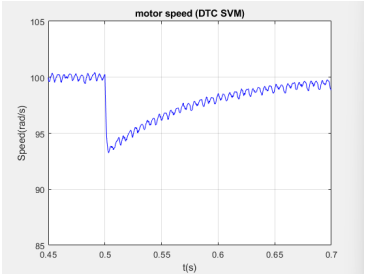
The following figures illustrate the system's dynamic response during this speed variation test: -



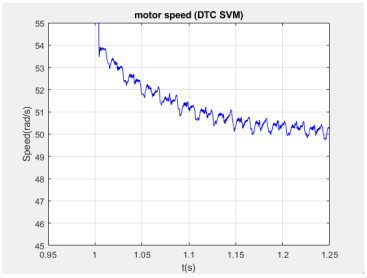
(a) Overall motor speed response (Full view).



(b) Initial transient (0 to 0.35 s).

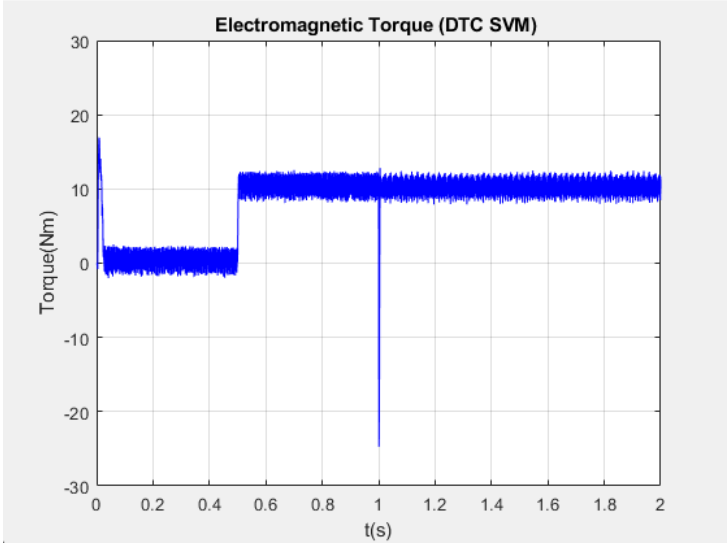


(c) Load application at ($t = 0.5$ s).

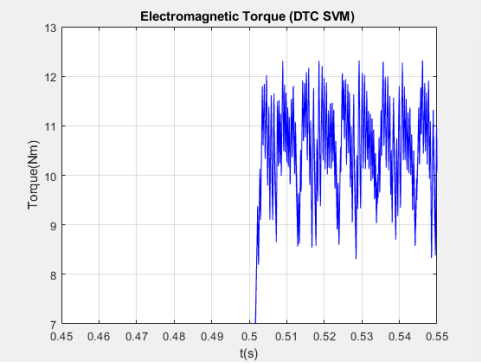


(d) Speed change at ($t = 1.0$ s).

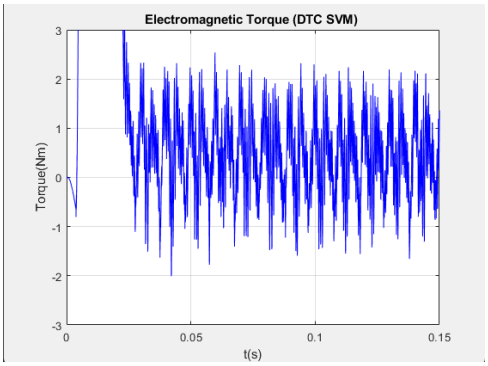
Figure I.17 Simulated motor speed response for a step change from 100 rad/s to 50 rad/s, including 10 Nm load torque application, in the DTC-SVM system.



(a) Overall electromagnetic torque response(Full view).

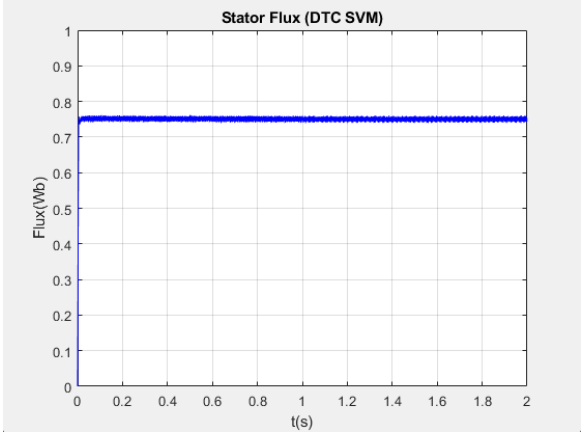


(b) Zoomed view at $t = 0.5$ s (load application).

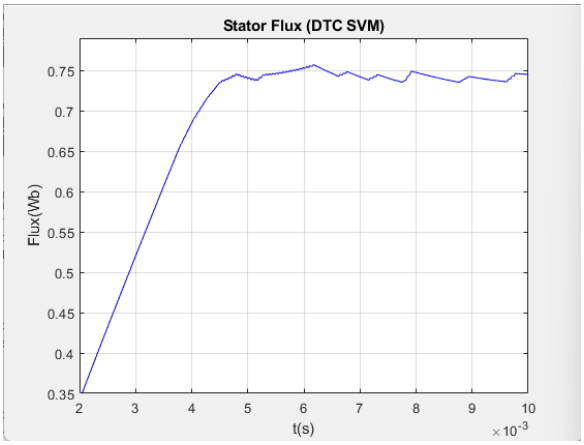


(c) Zoomed view of initial torque ripple (0 to 0.15 s).

Figure I.18 Simulated electromagnetic torque response during the speed variation test from 100 rad/s to 50 rad/s.



(a) Overall stator flux response(Full view).



(b) Zoomed-in view of initial flux transient .

Figure I.19 Simulated stator flux response in the DTC-SVM system during the speed variation test from 100 rad/s to 50 rad/s with 10 Nm load torque.

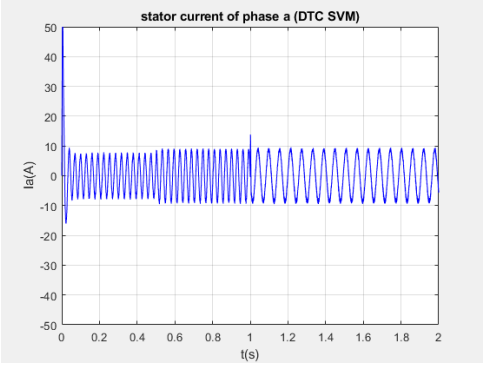


Figure I.20 Stator current (phase 'a') for speed variation test.

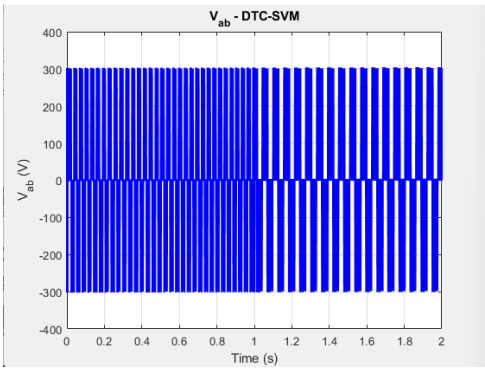


Figure I.21 Stator voltage components V_α and V_β for speed variation test with DTC-SVM control.

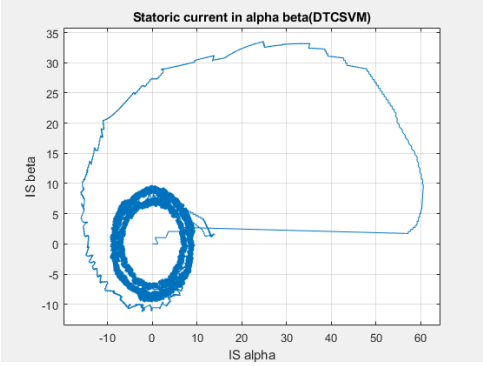


Figure I.22 Stator current vector locus ($I_{s\alpha}$ vs $I_{s\beta}$).

► **Interpretations of the results :**

- **Speed Response:** The motor speed rapidly reaches the initial reference of 100 rad/s with minimal overshoot, demonstrating the effectiveness of the speed control loop during startup. At $t = 0.5$ s, the application of a 10 Nm load causes a momentary, small dip in speed, which is quickly compensated by the controller, showing good load disturbance rejection. When the speed reference changes to 50 rad/s at $t = 1.0$ s, the motor smoothly decelerates and settles accurately at the new reference, again with fast response and minimal transient oscillations. This indicates precise speed tracking capability.

- **Torque Response:** The electromagnetic torque exhibits a sharp transient during startup to accelerate the motor to 100 rad/s, confirming the controller's ability to generate high dynamic torque. At $t = 0.5$ s, the torque instantaneously rises to compensate for the applied load, effectively maintaining the speed. When the speed reference drops to 50 rad/s at $t = 1.0$ s, the torque reverses its sign momentarily to actively decelerate the motor (regenerative braking), before settling at the value required to maintain 50 rad/s under load. The torque ripple observed in steady-state is relatively low, which is characteristic of DTC-SVM.

- **Stator Flux Response:** The stator flux magnitude is well-regulated and maintained close to its reference value throughout the operation, including during transients caused by load application and speed change. The zoomed view at startup shows a rapid build-up of flux, followed by stable regulation, highlighting the efficient flux control provided by DTC-SVM. The circular trajectory (not explicitly shown in the plots but implied by DTC-SVM) of the stator flux vector is expected to be maintained with minimal distortion, ensuring optimal motor operation.

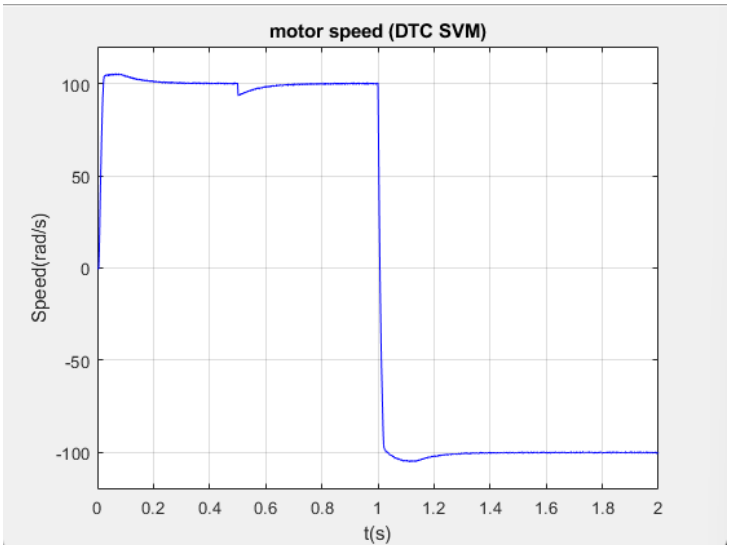
- **Stator Current (Phase 'a'):** The stator current waveform demonstrates a healthy sinusoidal shape in steady-state, indicative of good current quality. During transient periods, such as startup, load application, and speed change, the current amplitude appropriately adjusts to support the required torque, reflecting the dynamic current response of the system. The absence of significant overshoots or distortions during these transients confirms robust current control.

- **Phase-to-Phase Voltage (V_{ab}):** The phase-to-phase voltage V_{ab} shows a pulse-width modulated (PWM) waveform, typical for a two-level inverter. Its amplitude and frequency adapt dynamically to the motor's operating point, providing the necessary voltage to control the speed and flux. The presence of high-frequency switching harmonics is expected, but the overall voltage delivery effectively supports the desired motor performance.

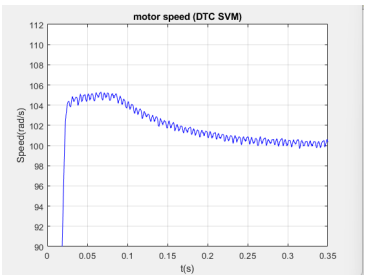
I.10.2 Reference Speed Reversal Test with Load Torque (100 rad/s to -100 rad/s)

This scenario rigorously tests the system's performance under a complete reversal of the speed reference, transitioning from a positive 100 rad/s to a negative -100 rad/s. A constant load torque of 10 Nm is applied at $t = 0.5$ s. This scenario is particularly challenging as it involves passing through zero speed and potentially reversing the motor's direction, highlighting the robustness and control precision of the DTC-SVM algorithm during regeneration or rapid direction changes. The analysis will focus on the transient response, stability during the reversal, and the quality of flux and current regulation under these demanding conditions.

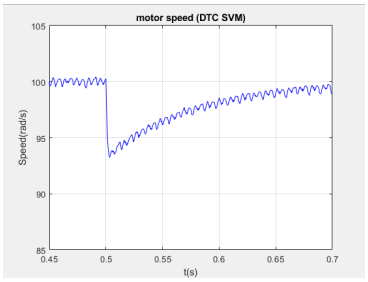
The following figures illustrate the system's dynamic response during this speed reversal test:



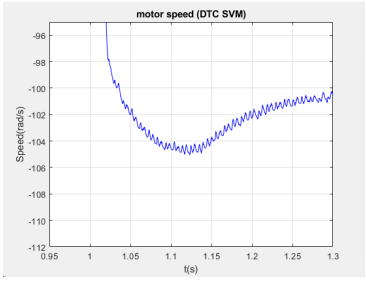
(a) Overall motor speed response (Full view).



(b) Initial transient (0 to 0.35 s).

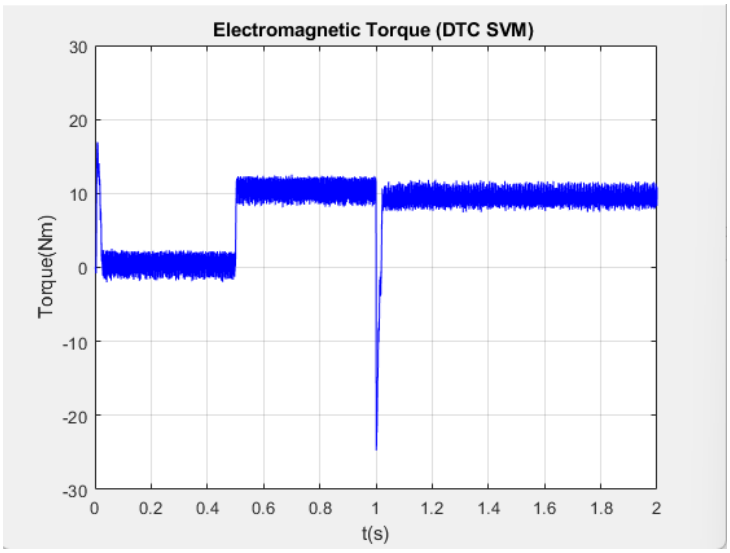


(c) Load application at ($t = 0.5$ s).

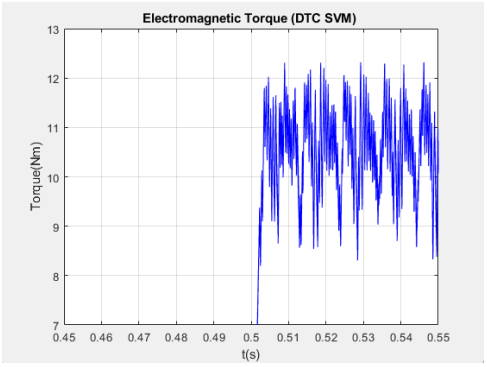


(d) Speed reversal at ($t = 1.0$ s).

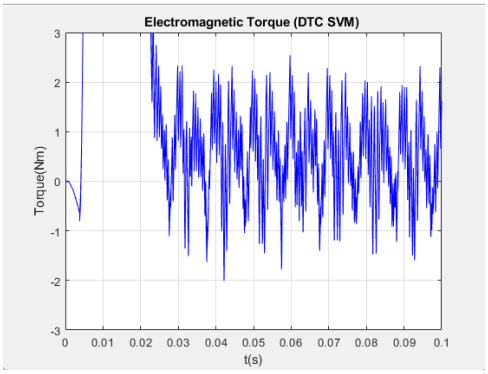
Figure I.23 Simulated motor speed response for a step reversal from 100 rad/s to -100 rad/s, including 10 Nm load torque application, in the DTC-SVM system.



(a) Overall electromagnetic torque response(Full view).

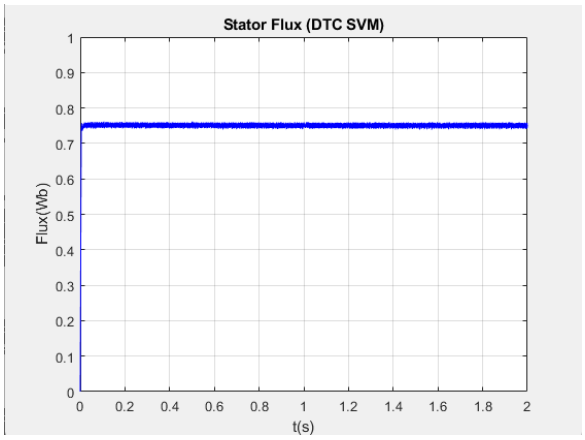


(b) Zoomed view at $t = 0.5$ s (load application).

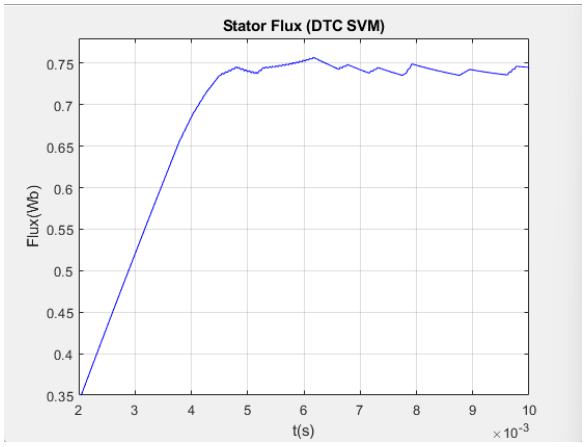


(c) Zoomed view of initial torque ripple (0 to 0.15 s).

Figure I.24 Simulated electromagnetic torque response during the speed reversal test from 100 rad/s to -100 rad/s.



(a) Overall stator flux response(Full view).



(b) Zoomed-in view of initial flux transient .

Figure I.25 Simulated stator flux response in the DTC-SVM system during the speed reversal test from 100 rad/s to -100 rad/s with 10 Nm load torque.

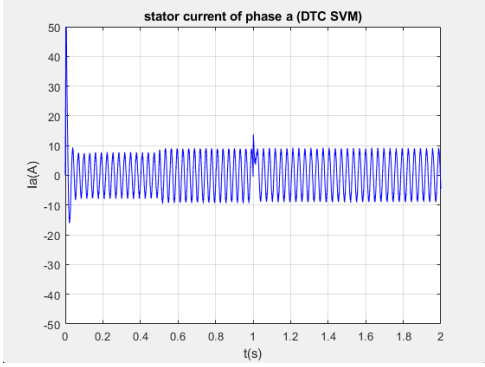


Figure I.26 Stator current (phase 'a') for speed reversal test.

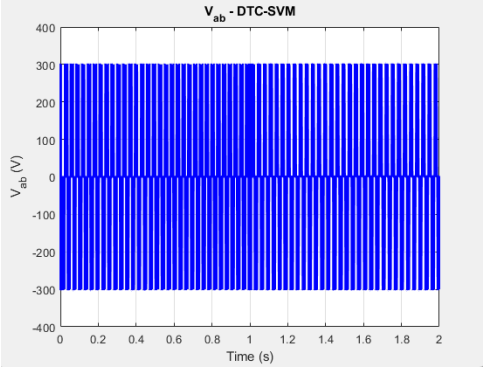


Figure I.27 Stator voltage components V_α and V_β for speed reversal test with DTC-SVM control

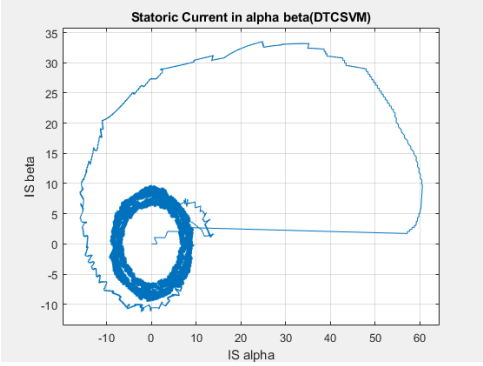


Figure I.28 Stator current vector locus ($I_{s\alpha}$ vs $I_{s\beta}$).

► **Interpretations of the results :**

- **Speed Reversal Performance:** The speed reversal from 100 rad/s to -100 rad/s at $t = 1.0$ s is executed smoothly and rapidly. The motor decelerates through zero speed and accelerates to the negative reference with a fast settling time and negligible overshoot, demonstrating the excellent dynamic capabilities of the DTC-SVM in handling challenging speed trajectories. The control effectively manages the transition, maintaining stability throughout the reversal.

- **Torque Dynamics during Reversal:** During the speed reversal, the electromagnetic torque undergoes significant, rapid changes. It becomes highly negative to provide strong braking torque for deceleration, then reverses sign to accelerate the motor in the opposite direction. This dynamic torque response is crucial for achieving fast and precise speed reversal, highlighting the immediate and effective torque control offered by DTC-SVM. The torque ripple remains within acceptable limits even during these aggressive transients.

- **Stator Flux Stability:** Despite the abrupt change in speed and torque direction, the stator flux magnitude remains constant and well-regulated around its reference value. This robust flux control is a key advantage of DTC-SVM, ensuring that the magnetic state of the motor is maintained, which contributes to the smooth operation and high dynamic performance during reversal.

- **Stator Current Waveforms:** The stator currents show large transient amplitudes during the speed reversal, which are necessary to generate the high braking and accelerating torques. Importantly, the current waveforms remain symmetrical and free from excessive distortions or uncommanded oscillations, indicating effective current regulation by the DTC-SVM. In steady-state, both before and after reversal, the currents are nearly sinusoidal, confirming good power quality.

- **Comparison to Speed Variation Test:** The speed reversal test is a more demanding scenario than a simple speed step change. The DTC-SVM system demonstrates superior robustness and dynamic performance in this test, handling the transition through zero speed and direction change with high precision and stability. This underscores the algorithm's suitability for applications requiring frequent and rapid changes in operating conditions, including regenerative braking and bidirectional operation.

I.11 Conclusion

When applied to a induction motor, the DTC-SVM control offers a strong blend of stability and robustness. When it comes to precisely controlling torque and speed in induction motor, this control strategy has several benefits.
It also enables for precise and dynamic torque control, which is essential for many industrial applications.
Nevertheless, DTC-SVM control has many drawbacks and defects in addition to its many benefits.
Couple oscillations are present.
The presence of oscillations in torque. - It could be challenging for the DTC-SVM control to produce strong torque at low Applications requiring a start or low speed operation may find the DTC-SVM control's performance limited due to its potential inability to generate strong torque at low speeds.

Chapter 2

Comparative Analysis of DTC-SVM Developed with ANN and Classic DTC-SVM

II.1 Introduction

Modern industrial, automotive, and robotics applications put a high premium on precise and efficient motor control to achieve high performance, energy efficiency, and reliability of operation. Traditional approaches to motor control, while robust in the majority of applications, are prone to severe limitations when confronted with inherent non-linearities, parameter uncertainties, and complex dynamic characteristics of electrical machines in modern applications. These constraints require the application of more advanced control techniques which are able to adapt to variable operating conditions and to optimize the performance on-line. During the past few years, the rapid advancement of artificial intelligence, particularly in the field of machine learning, has opened up new avenues for the development of intelligent control systems. Among them, Artificial Neural Networks (ANNs) have been a highly promising paradigm, offering unprecedented capabilities regarding modeling complex non-linear relationships, learning from experience, and adapting to new environments. This thesis investigates the application of ANNs to enhance the accuracy and robustness of motor control systems, seeking to go beyond the limitations of conventional approaches. By taking advantage of the pattern-recognition and adaptive properties of ANNs, this research is expected to demonstrate significant improvements in some of the main problems of motor control, towards the development of more powerful and intelligent control methodologies for a broad range of engineering applications.

II.2 Introduction to Artificial Neural Networks

Artificial Neural Networks (ANNs) are mathematical constructs similar to the brain's neural structures which form the basis for advanced machine learning systems and AI technologies. ANNs are capable of undertaking intricate data-driven activities including classification, decision-making, and even regression [**36**].

ANNs include numerous nodes called 'neurons' that constitute the sublayers of the neural network including the output layer, input layer and one or more hidden layers. The connections between the neurons also called synapses bear weights which are further modified in the training phase [**37**]. There are more optimization techniques that can be used with ANN for example: gradient descent and backpropagation which allow continuous refinement to enhance the prediction capabilities of the network.

Speech recognition, natural language processing, computer vision, and autonomous systems are just some of the areas where neural networks are already functioning [**38**]. The development of deep learning increases the sophistication with which ANNs are applied such as implementing Convolutional and Recurrent Neural Networks (CNNs and RNNs respectively) allowing for rapid advancements in AI research and expansions.

II.2.1 Biological Neural Network Inspiration

The general idea behind Artificial Neural Networks (ANNs) is a direct result of the intricate biological neural networks responsible for the brain and nervous system of living organisms. These are founded on specialized cells called **neurons** which are the minimum processing units. The intricate interconnectivity of biological neural networks is much superior to that used in artificial neural computing to date.

A typical biological neuron comprises four fundamental components, as illustrated in **Figure II.1**:

- **Dendrites:** These specialized branching structures are primarily responsible for receiving electrochemical signals and information from other neurons.
- **Soma:** Also known as the cell body, the soma integrates and processes the information collected by the dendrites.
- **Axon:** This elongated projection serves as the primary conduit through which the neuron transmits electrical signals and information to other neurons.
- **Synapses:** These are the specialized junctions that facilitate communication, forming the connection point between the axon of one neuron and the dendrites (or soma) of another.

Neurons communicate by sending and receiving information via electrochemical signals at these synaptic junctions. The ability to learn, process information, adapt to new data, and perform intricate cognitive functions in biological systems provides the essential template for artificial intelligence. This remarkable capacity for learning and adaptation is made possible through complex parallel processing and adaptive learning mechanisms, notably **synaptic plasticity**, where the strength of connections can be dynamically strengthened or weakened over time [**39**].

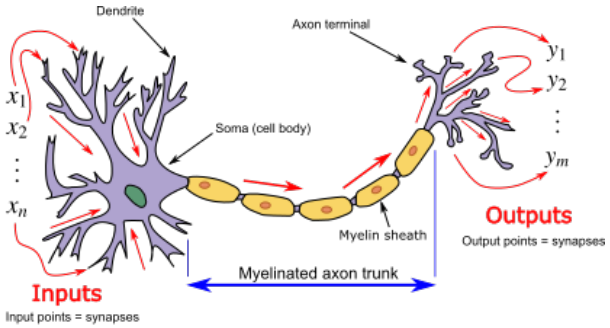


Figure II.1 Schematic of a Biological Neuron.

II.2.2 Essential Principles of Artificial Neural Networks (ANNs)

Underpinned by computational intelligence lies an Artificial Neural Network (ANN), systematically designed based on the intricate structure and function of biological neural networks found in the brain. At their core, ANNs comprise numerous interconnected processing units, often referred to as **nodes** or **neurons**. Each neuron receives inputs (x_i), multiplies them by corresponding **weights** (w_i), sums these weighted inputs, and incorporates a **bias term** (b). This aggregated input, known as the net input (z), is then transformed by a non-linear **activation function** (ϕ) to produce an output (a). Mathematically[**40**], the output of a single neuron can be expressed as:

$$\begin{aligned} z &= \sum_{i=1}^{N} w_i x_i + b \\ a &= \phi(z) \end{aligned} \tag{2.1}$$

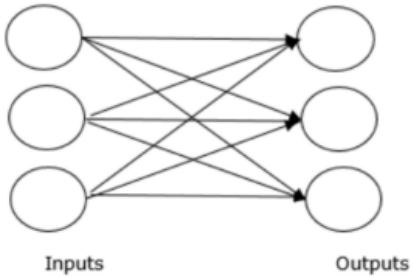


Figure II.2 Schematic of a Single Neuron.

These neurons are systematically organized into distinct **layers**: an **input layer** that directly receives raw data, one or more intermediate **hidden layers** responsible for complex feature extraction and transformation, and an **output layer** that generates the network's final prediction or decision. The non-linear nature of activation functions (e.g., sigmoid, ReLU, tanh) is critical, as it enables ANNs to model and approximate highly complex, non-linear relationships within the data, which linear models cannot achieve.

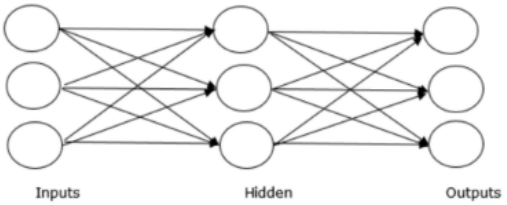


Figure II.3 General Architecture of a Multi-Layer Perceptron (MLP).

The learning process within an ANN primarily involves two phases: the **feedforward pass**, during which input data propagates sequentially from the input layer through the hidden layers to the output layer, yielding an initial prediction. Subsequently, **backpropagation**, a gradient-based optimization algorithm, computes the error of this prediction (e.g., mean squared error) and propagates this error backward through the network. This backward propagation facilitates the iterative adjustment of the synaptic weights (w_{ij}) and biases (b_j) of each connection using a gradient descent approach to minimize the objective function [**37**]. The update rule for a weight w_{ij} for instance, can be generally represented as:

$$w_{ij}(t+1) = w_{ij}(t) - \eta \frac{\partial E}{\partial w_{ij}} \tag{2.2}$$

where E is the error function and η is the learning rate. This continuous refinement of weights and biases allows the network to progressively enhance its ability to perform the designated task.

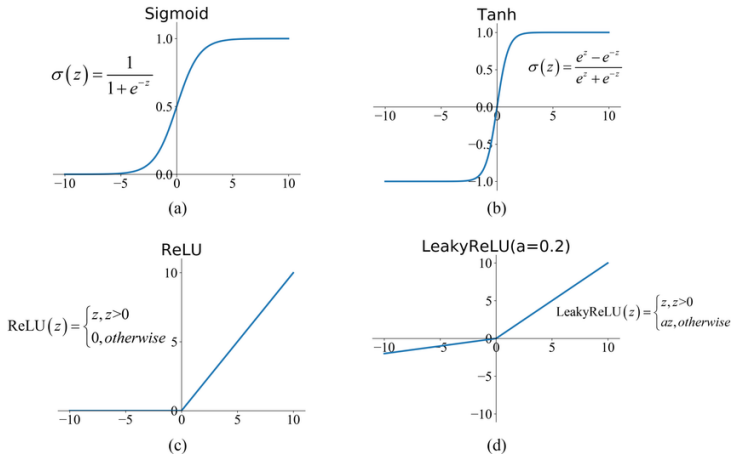


Figure II.4 Common Activation Functions. Plots of sigmoid, ReLU, and tanh functions illustrating their non-linear behavior.

II.2.3 Learning Paradigms

The process by which Artificial Neural Networks adapt their internal parameters (weights and biases) to perform a specific task is governed by various learning paradigms, each suited for different data characteristics and problem types:

- **Supervised Learning:** This is the most prevalent paradigm for training ANNs, particularly for tasks like classification and regression. In supervised learning, the network is trained on a labeled dataset, meaning each input instance is paired with its corresponding correct output (ground truth). The network learns by minimizing a predefined **loss function** that quantifies the discrepancy between its predictions and the true outputs. The learning signal is derived directly from these explicit input-output mappings. Your current motor control application, which involves predicting voltage commands (V_{ds}, V_{qs}) from known motor states and references, exemplifies a supervised learning problem.[**33**]

- **Unsupervised Learning:** In contrast to supervised learning, unsupervised learning deals with unlabeled data. The primary objective is to discover inherent patterns, structures, or relationships within the input data itself without explicit guidance. Common tasks include clustering (grouping similar data points), dimensionality reduction (finding lower-dimensional representations), and density estimation. Autoencoders are a classic example of ANNs often trained using unsupervised learning principles to learn efficient data encodings.[**33**]

- **Reinforcement Learning (RL):** This paradigm involves an intelligent agent learning to make sequential decisions by interacting with an environment. The agent performs actions and receives feedback in the form of numerical rewards or penalties, with the goal of maximizing its cumulative reward over time. ANNs frequently serve as function approximators within RL agents (e.g., as policy networks or value networks), enabling them to learn complex behavioral strategies in dynamic and uncertain environments. While distinct from typical regression/classification tasks, RL combined with ANNs (Deep Reinforcement Learning) is a powerful approach for complex control problems.[**33**]

II.2.4 Feedforward and Backpropagation

The core mechanism for training most supervised ANNs involves two interconnected phases: the **feedforward pass** and **backpropagation**. During the feedforward pass, input data propagates sequentially from the input layer, through the hidden layers (where each neuron performs its weighted sum and activation function calculation), to the output layer, resulting in an initial prediction. This prediction is then compared to the true target value using a chosen **loss function** (e.g., Mean Squared Error). The calculated error is then propagated backward through the network during the **backpropagation** phase. This process efficiently computes the gradient of the loss function with respect to each weight and bias in the network. Finally, a **gradient descent** optimizer (or one of its variants) uses these gradients to iteratively adjust the network's parameters, thereby minimizing the error and allowing the network to progressively learn the underlying relationships in the data. This iterative refinement continues over multiple epochs until the network's performance reaches a satisfactory level or converges.[**33**]

II.2.5 Types of Artificial Neural Networks

While all Artificial Neural Networks share the fundamental principles of interconnected neurons and adaptive learning, their architectural configurations vary significantly, leading to different types optimized for specific data structures and tasks. Understanding these variations is crucial for selecting the appropriate network design for a given problem. The most prevalent ANN architectures include:

II.2.5.1 Feedforward Neural Networks (FNNs) / Multi-Layer Perceptrons (MLPs)

Feedforward Neural Networks, often referred to as Multi-Layer Perceptrons (MLPs), represent the foundational architecture of ANNs. In these networks, information flows strictly in one direction: from the input layer, through one or more hidden layers, and finally to the output layer. There are no feedback loops, and connections only move forward. Each neuron in a layer is typically connected to every neuron in the subsequent layer. FNNs are widely used for a broad range of tasks, including classification, regression, and pattern recognition, making them a versatile choice for mapping complex input-output relationships. The ANNs developed in this study for flux and torque estimation are based on this feedforward architecture.

II.2.5.2 Recurrent Neural Networks (RNNs)

Recurrent Neural Networks are uniquely designed to process sequential data, where the order and context of information over time are critical. Unlike FNNs, RNNs incorporate feedback loops, allowing information to persist from previous steps of a sequence. This "memory" capability enables them to capture temporal dependencies. Common variants like Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs) address the vanishing gradient problem inherent in vanilla RNNs, making them more effective for learning long-term dependencies. RNNs are extensively applied in natural language processing (e.g., machine translation, speech recognition), time series prediction (e.g., stock market forecasting), and sequential data analysis.

II.2.5.3 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks are primarily renowned for their exceptional performance in processing grid-like data, most notably images. Their architecture is inspired by the visual cortex of animals and incorporates specialized layers, such as convolutional layers and pooling layers. Convolutional layers apply learnable filters to input data, effectively extracting spatial hierarchies of features (e.g., edges, textures, objects). Pooling layers then downsample these feature maps, reducing dimensionality and making the representations more robust to variations. CNNs have revolutionized fields such as computer vision (e.g., image classification, object detection), medical imaging, and even certain types of time-series analysis with spatial correlations.

II.2.5.4 Other Architectures

Beyond these primary types, the field of neural networks is rich with specialized architectures designed for distinct purposes. These include:

- **Autoencoders:** Networks designed to learn efficient data encodings (representations) in an unsupervised manner, often used for dimensionality reduction or anomaly detection.
- **Generative Adversarial Networks (GANs):** Comprising a generator and a discriminator network that compete, GANs are powerful for generating new data instances that resemble the training data (e.g., realistic images, audio).
- **Radial Basis Function (RBF) Networks:** Networks that use radial basis functions as activation functions, often employed for function approximation and interpolation tasks.

The choice of ANN type heavily depends on the characteristics of the input data and the specific problem being addressed, leveraging each architecture's strengths to optimize performance.

II.2.6 Advantages of ANNs Over Traditional Models

Artificial Neural Networks offer compelling advantages over conventional statistical and traditional machine learning models, making them exceptionally well-suited for addressing complex, real-world problems. A primary benefit is their inherent capacity for **nonlinear modeling**. Unlike many classical algorithms that often assume linear relationships between features and targets, or necessitate extensive manual feature engineering to capture nonlinearities, ANNs, through their hierarchical structure and the application of non-linear activation functions ($\phi(z)$) at each neuron, can intrinsically learn and approximate highly intricate non-linear mappings. This enables them to capture complex patterns that are intractable for linear models.

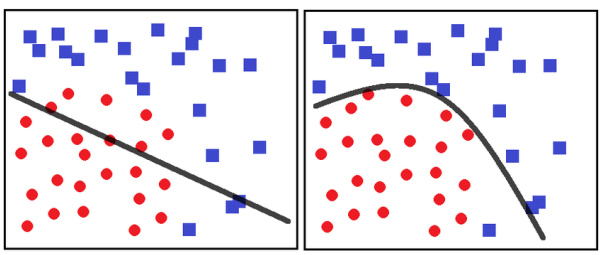


Figure II.5 Illustration of Linear vs. Non-linear Decision Boundaries.

Furthermore, their remarkable **adaptability** allows ANNs to learn effectively from vast and often noisy datasets without requiring explicit, rule-based programming for every conceivable scenario. The iterative nature of their training process, driven by optimization algorithms like stochastic gradient descent, allows ANNs to continuously adjust their internal parameters (w_{ij}, b_j) to minimize prediction errors, thereby enabling them to generalize robustly to unseen data and even adapt to shifts in data distribution over time. Finally, a significant superiority, particularly evident in deep learning architectures, is their ability for **feature learning** (or representation learning). Instead of relying on painstakingly hand-crafted features—a process that is frequently labor-intensive, domain-specific, and prone to human bias—ANNs can automatically identify and extract hierarchical, discriminative features directly from raw input data. For example, in image processing, lower layers might learn edge detectors, while higher layers combine these to form more complex patterns like shapes or object parts [**38**]. This autonomous and data-driven feature extraction empowers ANNs to achieve unparalleled performance in highly complex domains such as computer vision, natural language processing, and speech recognition, where traditional methodologies often prove insufficient [**37**].

II.2.7 Applications of ANNs in Motor Control

Within the domain of **motor control**, Artificial Neural Networks (ANNs) have emerged as an increasingly vital tool, particularly due to their exceptional capabilities in handling the inherent non-linear dynamics and complexities of modern electrical machines. Traditional motor control techniques, such as Proportional-Integral-Derivative (PID) controllers, often face limitations when confronted with significant parameter variations, unmodeled disturbances, or the intrinsic non-linearities arising from magnetic saturation, friction, or cross-coupling effects within motors and power electronics. ANNs, however, offer a powerful alternative by leveraging their ability to learn intricate motor models directly from data, bypassing the need for precise analytical derivations.

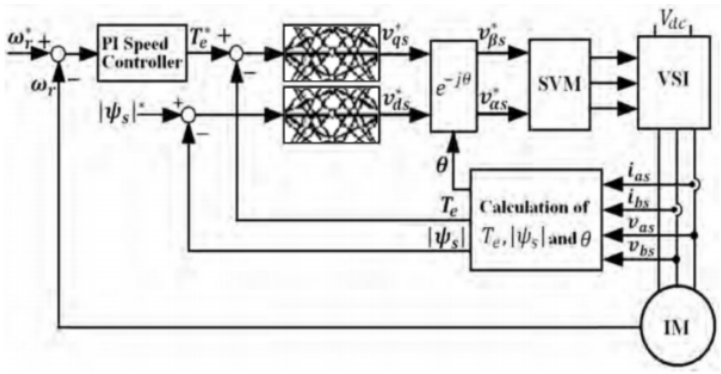


Figure II.6 Block Diagram of an ANN-based Motor Control System.

For instance, ANNs are extensively employed in advanced control strategies such as **predictive torque control** and **direct torque control (DTC)**. In these applications, ANNs can accurately estimate real-time motor states (e.g., flux Ψ, speed ω_r, torque T_e) and predict future system behavior based on current inputs and learned dynamics. The estimated output for a given state $X(t)$ and input $U(t)$ might be represented as:

$$\hat{X}(t+1) = \text{ANN}(X(t), U(t)) \tag{2.3}$$

This capability of predicting outcomes enables faster responses to transients, better performance in the steady-state, and much lower torque and flux ripple in contrast to heuristic or look-up table methods. Moreover, ANNs' **adaptability** directly makes them suitable for controlling changes in a system, such as to motor parameters due to temperature rise, aging, and even manufacturing tolerances, which ensures that the performance is unaffected under different conditions. ANNs are also very competent in **fault detection and diagnosis** in motor drive systems. By familiarizing themselves with the complex patterns associated with normal operating conditions, ANN systems can quite easily detect emerging problems such as stator winding faults, bearing failures, or even sensor faults, often before those problems would lead to main catastrophic failures.

All of this powerful non-linear mapping together with robust learning techniques makes it easy to claim that ANNs technologies built for automated control of motors to be used in industrial, automotive, and robotic tasks are highly resilient and efficient.

II.3 ANN Architecture Design

II.3.1 Model Structure

The design of an Artificial Neural Network (ANN) model structure is a critical phase that dictates its capacity to learn complex patterns and generalize effectively. It involves defining the number of layers, the neuron count within each layer, and the choice of activation functions. This section outlines the specific architectural components employed for both the flux and torque estimation models developed in this study.

II.3.1.1 Common Architectural Components for Both Flux and Torque ANNs:

- **Input Layer:** For the **flux estimation ANN**, the input layer consists of **two neurons**, receiving the reference flux (Ψ_{ref}) and the actual flux (Ψ). Similarly, for the **torque estimation ANN**, the input layer also has **two neurons**, taking the reference torque (T_{ref}) and the actual motor torque (T_e). This direct mapping ensures that each network receives the specific relevant state variables for its respective estimation task.

- **Hidden Layers:** Both ANNs are structured with **two hidden layers**. The first hidden layer contains **16 neurons**, while the second hidden layer consists of **8 neurons**. This progressively decreasing neuron count through the hidden layers is a common design strategy, enabling the network to learn increasingly abstract and compact representations of the input data. This approach also helps manage model complexity and computational burden.

- **Activation Functions (Hidden Layers):** Distinct activation functions were chosen for each hidden layer to enhance the network's ability to model complex non-linearities. The first hidden layer utilizes the **hyperbolic tangent sigmoid ('tansig')** transfer function, which squashes outputs between -1 and 1, providing a strong non-linearity. The second hidden layer employs the **log-sigmoid ('logsig')** transfer function, mapping outputs between 0 and 1. The combination of these non-linear functions is crucial for allowing the network to approximate arbitrary non-linear mappings.

- **Output Layer:** For the **flux estimation ANN**, the output layer comprises a single neuron, tasked with predicting the direct axis voltage (V_{ds}). For the **torque estimation ANN**, the output layer also contains a single neuron, predicting the quadrature axis voltage (V_{qs}). In both cases, the output layer uses a **linear activation function**. This choice is essential as both V_{ds} and V_{qs} are continuous, unbounded voltage commands, and a linear activation allows the network to output any real number without constraint, making it suitable for regression tasks.

II.3.1.2 Summary of ANN Model Architecture:

The specific configuration of the ANNs developed is summarized in the table below, highlighting the consistent internal structure across both the flux and torque estimation models.

Table II.1 Summary of ANN Model Architecture

Component	Flux ANN	Torque ANN	Rationale / Justification
Input Layer	2 (Ψ_{ref}, Ψ)	2 (T_{ref}, T_e)	Direct mapping to relevant input features.
Hidden Layer 1 Neurons	16	16	Sufficient capacity for initial feature extraction.
Hidden Layer 1 Activation	`tansig`	`tansig`	Introduces non-linearity; maps to $[-1, 1]$.
Hidden Layer 2 Neurons	8	8	Further processes features; aids information abstraction.
Hidden Layer 2 Activation	`logsig`	`logsig`	Adds non-linearity; maps to $[0, 1]$ for complex pattern recognition.
Output Layer Neurons	1 (V_{ds})	1 (V_{qs})	Corresponds to single continuous output variable.
Output Layer Activation	Linear	Linear	Outputs unconstrained continuous values, suitable for regression tasks like voltage prediction.

II.3.2 Mathematical Formulation

The operational core of the designed Artificial Neural Networks for flux and torque estimation relies on specific mathematical formulations for forward propagation, loss computation, and parameter optimization.

II.3.2.1 Forward Propagation Equations

The forward propagation phase describes how input data is processed through the network to produce an output. For a neuron in layer l receiving inputs from layer $l-1$, the net input $z_j^{(l)}$ to neuron j in layer l is calculated as the weighted sum of outputs from the previous layer, plus a bias term:

$$z_j^{(l)} = \sum_{i=1}^{N_{l-1}} w_{ji}^{(l)} a_i^{(l-1)} + b_j^{(l)} \tag{2.4}$$

where N_{l-1} is the number of neurons in the previous layer, $w_{ji}^{(l)}$ is the weight connecting neuron i in layer $l-1$ to neuron j in layer l, and $b_j^{(l)}$ is the bias for neuron j in layer

l. The output (activation) $a_j^{(l)}$ of neuron j in layer l is then determined by applying the activation function $\phi^{(l)}$ to the net input:

$$a_j^{(l)} = \phi^{(l)}(z_j^{(l)}) \tag{2.5}$$

For the specific network architecture employed, the hidden layers use the hyperbolic tangent sigmoid ($\phi^{(1)}$ = tansig) and log-sigmoid ($\phi^{(2)}$ = logsig) activation functions, respectively, while the output layer uses a linear activation function ($\phi^{(3)}$ = linear).

II.3.2.2 Loss Function

To quantify the discrepancy between the network's predictions and the actual target values, a suitable loss function is employed. For regression tasks such as voltage prediction in motor control, the **Mean Squared Error (MSE)** is a widely adopted choice. The MSE loss function L for a given set of M training samples is defined as:

$$L = \frac{1}{M}\sum_{k=1}^{M}(y^{(k)} - \hat{y}^{(k)})^2 \tag{2.6}$$

And its square root, the Root Mean Squared Error (RMSE), is given by:

$$\text{RMSE} = \sqrt{\frac{1}{M}\sum_{k=1}^{M}(y^{(k)} - \hat{y}^{(k)})^2} \tag{2.7}$$

where M is the number of samples, $y^{(k)}$ is the actual target value for sample k, and $\hat{y}^{(k)}$ is the network's predicted value for sample k.

The optimization of the network's weights and biases (θ) is achieved through backpropagation. This algorithm efficiently computes the gradient of the loss function with respect to each parameter by propagating the error backwards through the network using the chain rule. The parameters are then updated iteratively using the Levenberg-Marquardt algorithm.

II.3.2.3 Backpropagation and Gradient Descent

The process of optimizing the network's weights and biases to minimize the loss function is performed using the **backpropagation algorithm** in conjunction with **gradient descent**. Backpropagation efficiently computes the gradient of the loss function with respect to each weight and bias in the network by propagating the error derivatives backward from the output layer to the input layer. For a weight $w_{ji}^{(l)}$, the gradient $\frac{\partial L}{\partial w_{ji}^{(l)}}$ indicates the direction and magnitude of change required to reduce the loss. The weights and biases are then updated iteratively using a gradient descent rule:

$$\begin{aligned} w_{ji}^{(l)}(t+1) &= w_{ji}^{(l)}(t) - \eta\frac{\partial L}{\partial w_{ji}^{(l)}} \\ b_j^{(l)}(t+1) &= b_j^{(l)}(t) - \eta\frac{\partial L}{\partial b_j^{(l)}} \end{aligned} \tag{2.8}$$

where t denotes the current iteration (epoch), and η is the learning rate, a hyperparameter that controls the step size of the updates. This iterative optimization process enables the network to learn the underlying relationships within the data.

II.3.3 Visual Representation

A clear visual representation of the designed Artificial Neural Network architecture is essential for understanding its topology and the flow of information. While the specific input and output variables differ, both the flux estimation ANN and the torque estimation ANN utilize an identical underlying feedforward neural network structure, as detailed in Section II.3.1.

Figure II.7 provides a schematic diagram of this common architecture. It illustrates the progression of data from the input layer through the hidden layers to the output layer, highlighting the neuron count and activation functions at each stage.

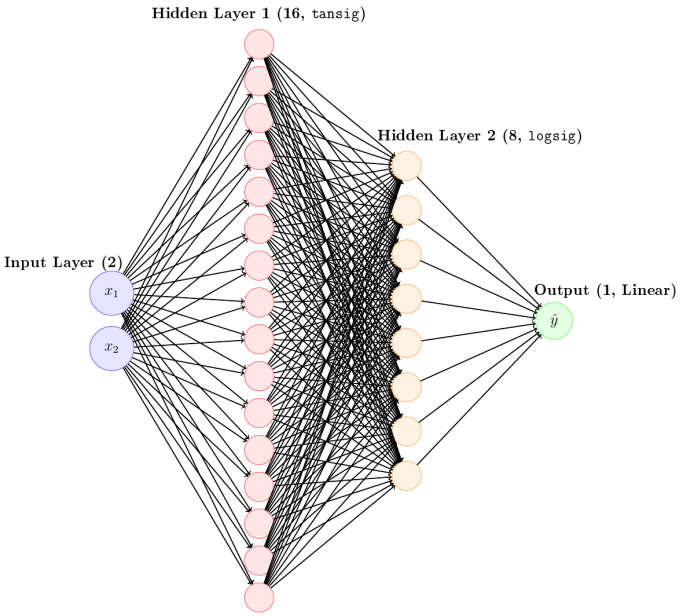


Figure II.7 Proposed ANN Architecture Diagram. This diagram illustrates the shared feedforward structure for both the flux and torque estimation models: a 2-neuron input layer, two hidden layers with 16 (`tansig`) and 8 (`logsig`) neurons respectively, and a single-neuron output layer with a linear activation function.

The diagram clearly shows the two input nodes, which receive features such as (Ψ_{ref}, Ψ) for the flux network or (T_{ref}, T_e) for the torque network. These inputs feed into the first hidden layer of 16 neurons, followed by the second hidden layer of 8 neurons. Finally, the output layer consists of a single neuron, producing either V_{ds} (for flux) or V_{qs} (for torque). The specified activation functions for each layer are critical for the network's ability to learn complex non-linear mappings.

II.4 DataPreparation and Preprocessing

II.4.1 Dataset Description

The data utilized for training and validating the Artificial Neural Network models was generated through **simulations conducted in a MATLAB/Simulink environment**. This approach allowed for the systematic generation of comprehensive operational data across a wide range of motor states and dynamic conditions, which is crucial for training robust and generalized ANN models.

The raw dataset comprises a total of **666,667 samples**. Each sample in this dataset consists of several key operational parameters representing the motor's state and desired commands, which serve as potential input features, along with their corresponding output voltage commands. Specifically, the comprehensive dataset includes:

For the training of the individual ANN models:

- The **Flux Estimation ANN** utilizes Ψ_{ref} and Ψ as its two input features to predict the V_{ds} output.
- The **Torque Estimation ANN** utilizes T_{ref} and T_e as its two input features to predict the V_{qs} output.

The large volume of samples is instrumental in enabling the ANNs to capture the complex non-linear relationships inherent in electrical machine dynamics and generalize effectively across diverse operating points and transient behaviors.

II.4.2 Preprocessing Steps

To ensure optimal network training and performance, several crucial preprocessing steps were applied to the raw dataset. These steps are vital for enhancing model stability, accelerating convergence during training, and improving generalization capabilities.

- **Normalization:** All input features and target output variables were normalized to a common scale. Specifically, **Min-Max scaling** was applied to transform the data into the range of $[0, 1]$. This method scales the values to a specific range by subtracting the minimum value of the feature and dividing by the range (maximum minus minimum). The formula for Min-Max normalization is:

$$x_{\text{normalized}} = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{2.9}$$

  where x is the original value, $x_{\min}$ is the minimum value of the feature across the dataset, and $x_{\max}$ is the maximum value of the feature across the dataset. This standardization helps prevent features with larger numerical ranges from disproportionately influencing the learning process and aids in accelerating gradient descent optimization.

- **Train-Test-Validation Split:** The comprehensive dataset was meticulously partitioned into training, validation, and testing subsets to facilitate robust model development and unbiased evaluation. An interleaved indexing strategy was employed to ensure that data points from across the entire operational range were represented in each subset, preventing any bias from sequential data. Specifically, approximately **50% of the data was allocated for training**, **25% for validation**, and the remaining **25% for testing**. This split ensures that the model is trained on a substantial portion of the data, tuned and monitored for generalization on the validation set, and finally assessed on completely unseen data in the test set to provide an unbiased estimate of its true performance.

II.4.3 Feature Engineering

The meticulous selection of input features is a critical aspect of designing an effective Artificial Neural Network, as it directly impacts the model's ability to learn and accurately

predict complex system behaviors. For the purpose of intelligent motor control, the chosen input features for our ANNs are directly derived from fundamental electrical machine theory and practical control system requirements, ensuring their high relevance to the desired voltage outputs.

The primary objective of these ANNs is to predict the direct-axis voltage (V_{ds}) and quadrature-axis voltage (V_{qs}) components, which are the fundamental control signals necessary to precisely regulate the motor's magnetic flux and electromagnetic torque. These outputs are inherently complex, arising from the non-linear and coupled dynamics of the motor, including magnetic saturation, inductive effects, and back-EMF.

The selected input features, used individually for the specialized flux and torque ANNs, provide the necessary state information to enable these predictions:

- For the **Flux Estimation ANN**, the input features are the **reference flux (Ψ_{ref})** and the **actual motor flux (Ψ)**. These two variables are direct indicators of the magnetic state of the machine and the desired flux level, making them crucial for determining the appropriate V_{ds} command required to control the direct-axis current and, consequently, the flux.

- For the **Torque Estimation ANN**, the input features are the **reference torque (T_{ref})** and the **actual motor torque (T_e)**. These inputs directly relate to the mechanical output and the desired torque, providing the necessary context for the ANN to predict the V_{qs} command, which primarily controls the quadrature-axis current responsible for torque production.

By providing these directly relevant and physics-informed input features, the ANNs are empowered to automatically extract and learn the intricate, non-linear mapping functions that link the desired and actual motor states to the precise voltage commands. This data-driven approach alleviates the need for explicit analytical models or complex lookup tables, which often struggle with the real-world complexities and dynamic nature of motor drive systems.

II.5 Training Methodology

II.5.1 Training Setup

The training of the Artificial Neural Network models was systematically conducted within a defined computational environment, ensuring both efficiency and reproducibility of the training process. The selection of specific hardware, software, and hyperparameters was crucial for achieving optimal model performance and convergence. Examples of the MATLAB Deep Learning Toolbox graphical user interface (GUI) used for monitoring and managing the training process for both flux and torque ANNs are depicted in **Figure II.8** and **Figure II.9**.

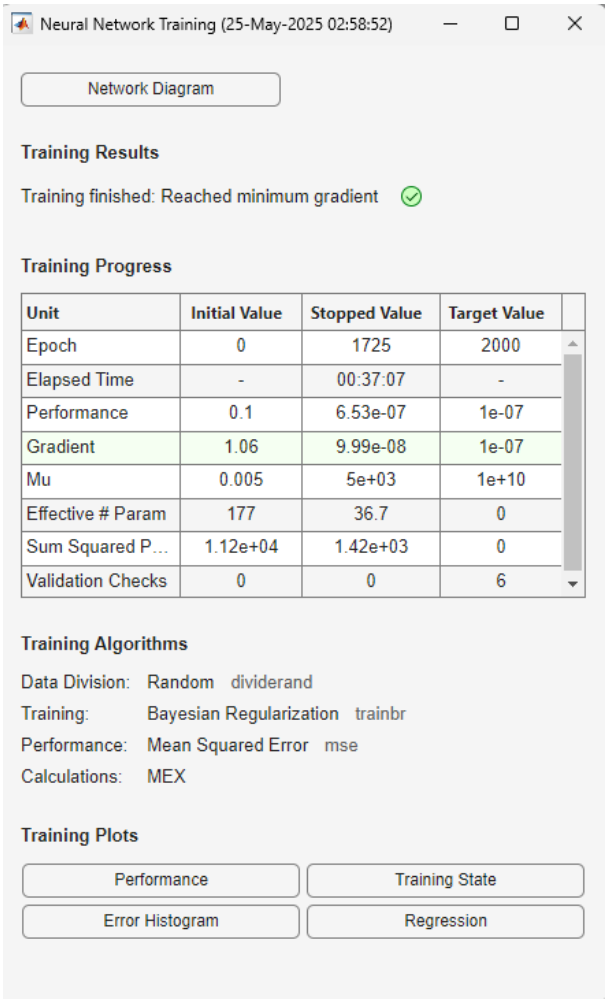


Figure II.8 MATLAB Neural Network Training GUI Overview for Flux Estimation ANN, showing training results and algorithms.

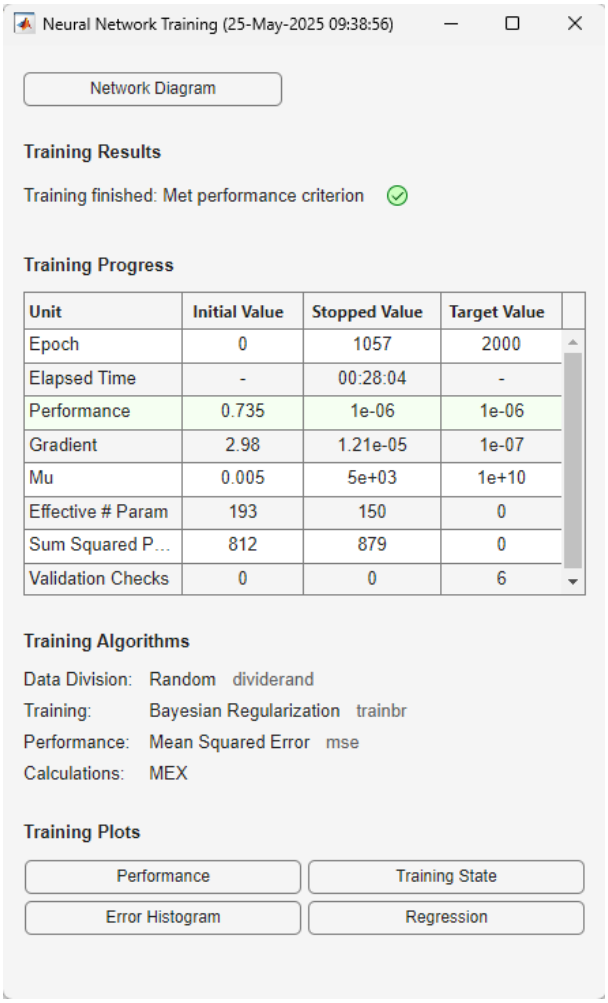


Figure II.9 MATLAB Neural Network Training GUI Overview for Torque Estimation ANN, showing training results and algorithms.

- **Hardware:** The computationally intensive training procedures were significantly accelerated by leveraging the capabilities of an **NVIDIA RTX 2050 GPU**. This graphics processing unit, equipped with **CUDA acceleration**, enabled parallel processing of numerical computations, thereby substantially reducing the overall training time required for processing large datasets and complex network architectures.
- **Software:** The implementation, training, and subsequent deployment of the ANNs were performed using the **MATLAB Deep Learning Toolbox**. This specialized toolbox provides a robust and comprehensive framework tailored for neural network development, offering optimized algorithms for data management, model training, and performance analysis within the MATLAB environment.
- **Hyperparameters:** A set of critical hyperparameters was carefully configured to guide the training process and facilitate effective learning:
  - **Epochs:** The training was configured to run for a maximum of **2000 epochs**. An epoch signifies one complete pass through the entire training dataset. This limit was set to allow sufficient iterations for the models to learn complex patterns and converge towards an optimal solution.

- **Optimizer:** The **Levenberg-Marquardt backpropagation algorithm** (corresponding to the `trainbr` function in MATLAB) was chosen as the training function. This algorithm is known for its efficiency and good performance in training feedforward networks, particularly for regression tasks with moderate-sized datasets. It combines the advantages of steepest descent and the Gauss-Newton algorithm. **Training Goal (Performance):** The training process was set to terminate when the Mean Squared Error (MSE) performance reached a predefined goal. Specifically, a goal of 1×10^{-7} was set for the flux estimation network, and a goal of 1×10^{-6} for the torque estimation network. These low error thresholds indicate a high degree of desired accuracy for the trained models.
- **Validation Checks (Early Stopping):** To prevent overfitting, an early stopping mechanism was implemented. The training was configured to halt if the validation error (performance on the validation set) failed to decrease for **6 consecutive iterations** (equivalent to `max_fail = 6` in MATLAB). This criterion ensures that the training stops when the model's ability to generalize to unseen data begins to degrade, even if the training error continues to decrease.

II.5.2 Training Process

The training of the Artificial Neural Network models is an iterative optimization process aimed at minimizing the Mean Squared Error (MSE) between the network's predictions and the actual target values. This process involves repeatedly feeding the network with training data, calculating the loss, and adjusting the network's internal weights and biases through the Levenberg-Marquardt backpropagation algorithm.

II.5.2.1 Flux ANN Training Results

The training progress for the Flux Estimation ANN is critically monitored through its performance curve and error distribution. The final Mean Squared Error (MSE) values obtained after training were 6.5378×10^{-7} for all data, 6.5487×10^{-7} for the validation set, and 6.5472×10^{-7} for the test set. **Figure II.10** further illustrates the MSE convergence across epochs for the training, validation, and test datasets. The rapid initial decrease in MSE signifies efficient learning, followed by a plateau indicating convergence towards the set performance goal of 1×10^{-7}. The consistency between training, validation, and test performance curves suggests good generalization.

The distribution of prediction errors for the Flux ANN is presented in the error histogram in **Figure II.11**. A narrow distribution centered near zero indicates that the model's predictions are highly accurate and unbiased. Further details on the training state parameters, such as gradient and Mu, are provided in **Figure II.12**, offering insights into the optimization trajectory and stability.

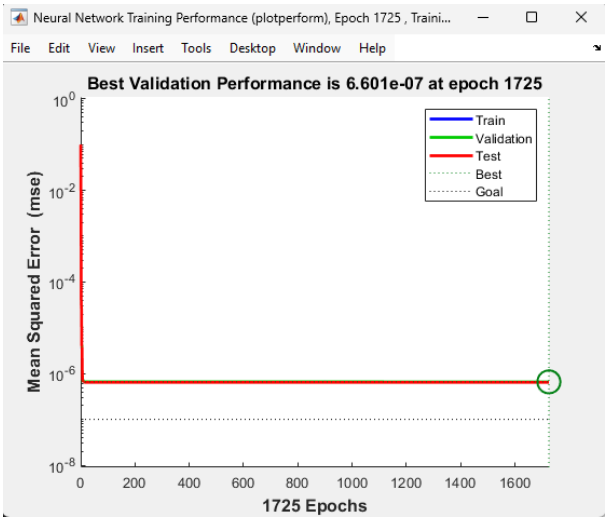


Figure II.10 Flux Estimation ANN Training Performance: Mean Squared Error (MSE) progression over epochs for training, validation, and test sets. Best validation performance reached 6.601e-07 at epoch 1725.

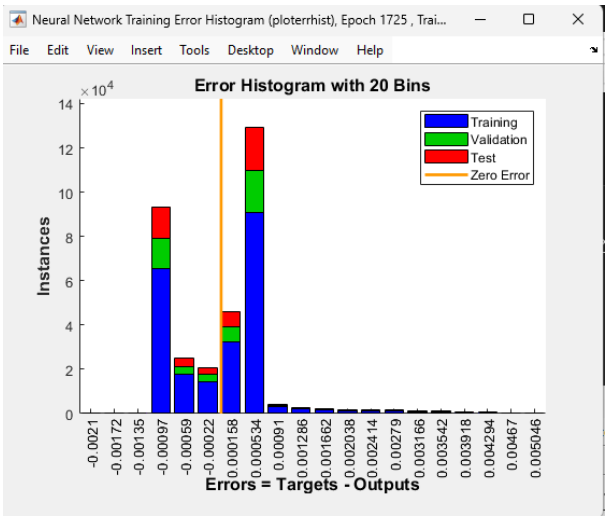


Figure II.11 Error Histogram of the Flux Estimation ANN after training, showing the distribution of prediction errors with 20 bins.

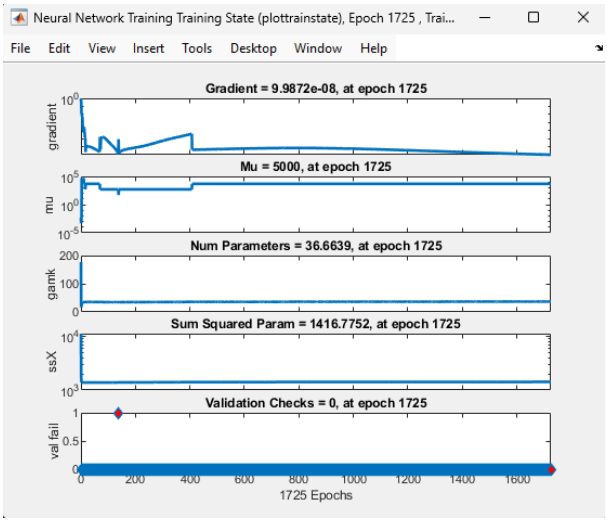


Figure II.12 Flux Estimation ANN Training State: Evolution of gradient, Mu, number of parameters, sum squared parameters, and validation checks during training.

II.5.2.2 Torque ANN Training Results

Similarly, the training performance for the Torque Estimation ANN is presented to demonstrate its learning characteristics. The final Mean Squared Error (MSE) values obtained after training were 1.0297×10^{-6} for all data, 1.0537×10^{-6} for the validation set, and 1.0668×10^{-6} for the test set. **Figure II.13** illustrates the Mean Squared Error (MSE) convergence curve, showing how the network effectively minimized prediction errors over epochs, reaching a performance goal of 1×10^{-6}. The close alignment of training, validation, and test curves confirms the model's robustness and generalization capability.

The error histogram for the Torque ANN, depicted in **Figure II.14**, provides a visual summary of the prediction error distribution, which is tightly centered around zero, indicating high accuracy. The training state parameters, including gradient and Mu, are detailed in **Figure II.15**, offering further insights into the optimization process for the torque estimation model.

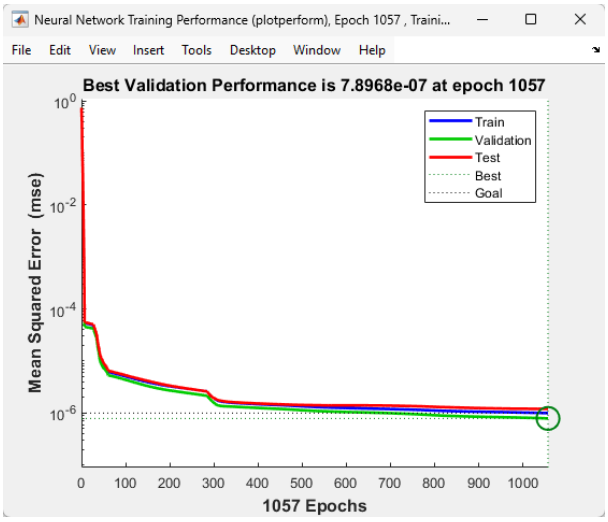


Figure II.13 Torque Estimation ANN Training Performance: Mean Squared Error (MSE) progression over epochs for training, validation, and test sets. Best validation performance reached 7.8968e-07 at epoch 1057.

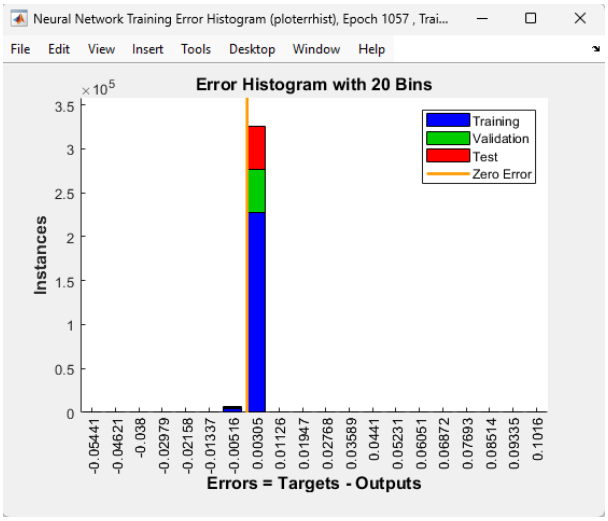


Figure II.14 Error Histogram of the Torque Estimation ANN after training, showing the distribution of prediction errors with 20 bins.

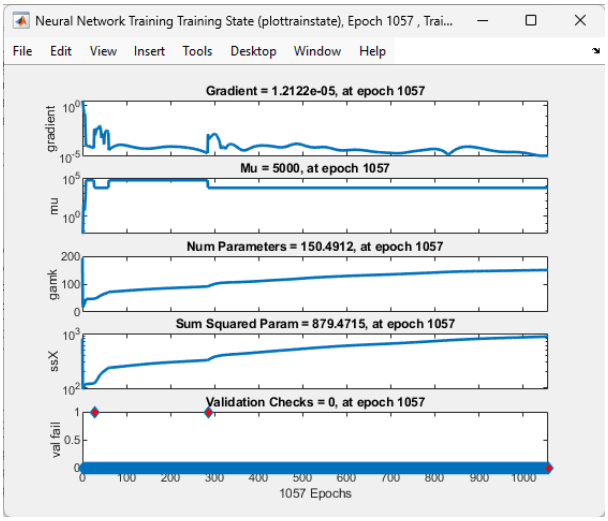


Figure II.15 Torque Estimation ANN Training State: Evolution of gradient, Mu, number of parameters, sum squared parameters, and validation checks during training.

II.5.2.3 Final Model Performance and Regression Fit

A comprehensive evaluation of the trained ANN models includes not only their training dynamics but also their final predictive capabilities on unseen data. Beyond numerical metrics, visual inspections of the model's predictive accuracy provide qualitative insights into its performance. **Figure II.16** and **Figure II.17** illustrate the regression fits for the flux and torque estimation ANNs, respectively. These plots compare the network's outputs against the true targets for the training, validation, test, and overall datasets. A close alignment of the data points along the ideal $Y = T$ (Output=Target) line, coupled with high R-values, confirms the strong correlation and predictive capability of the models across different data subsets.

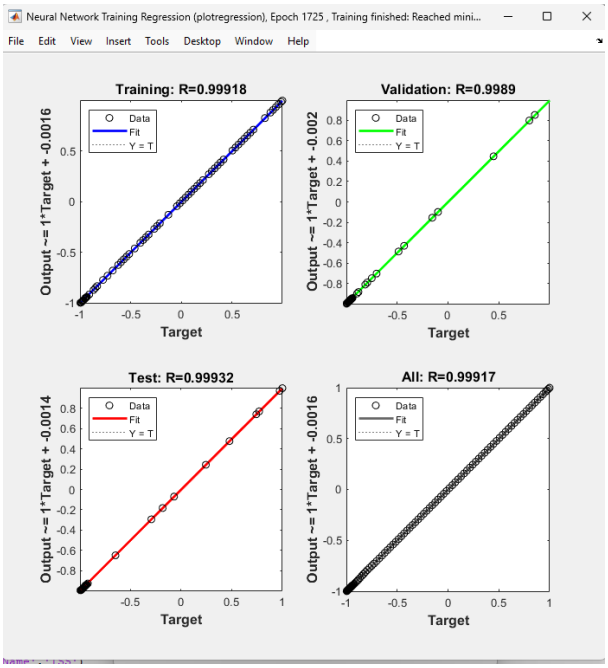


Figure II.16 Flux Estimation ANN Regression Plot showing the relationship between network outputs and targets for training, validation, test, and all data, along with R values. Note: R-values for training, validation, test, and all data are 0.99918, 0.9989, 0.99932, and 0.99917 respectively.

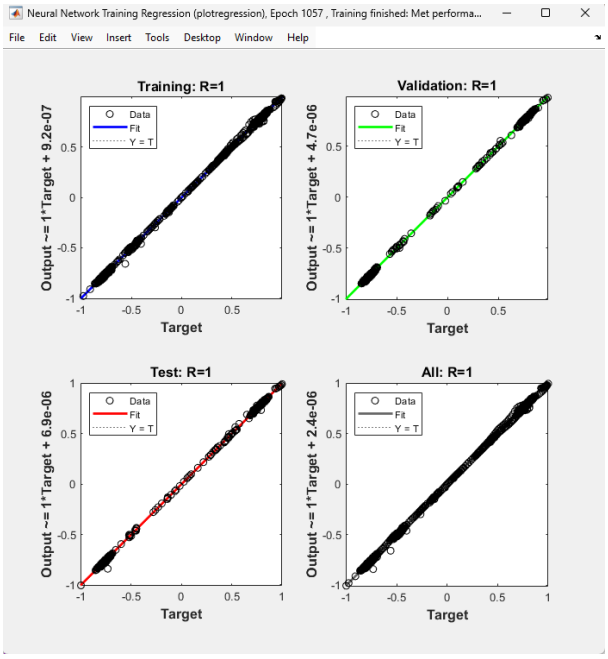


Figure II.17 Torque Estimation ANN Regression Plot showing the relationship between network outputs and targets for training, validation, test, and all data, along with R values. Note: R-values for training, validation, test, and all data are 1, 1, 1, and 1 respectively.

II.5.2.4 Overfitting Prevention

A critical challenge in training ANNs is preventing **overfitting**, a phenomenon where the model performs exceptionally well on the training data but poorly on unseen data due to memorizing noise or specific training examples rather than generalizing underlying patterns. To mitigate this, an **early stopping** strategy was effectively employed, as configured by the `max_fail` parameter in the training setup.

The training process was configured to halt if the performance on the designated validation dataset failed to improve (i.e., the validation error did not decrease) for **6 consecutive iterations** (equivalent to `max_fail = 6` in MATLAB). This mechanism ensures that the training ceases at the point where the model's generalization capability is optimized, even if the training error might still show a slight decreasing trend. By stopping when validation performance plateaus or degrades, the network avoids learning overly specific features of the training set that would hinder its ability to perform accurately on new, unobserved data. Other common regularization techniques, such as dropout, were not explicitly implemented in this specific network configuration, with early stopping serving as the primary overfitting prevention mechanism.

II.6 Case Study – ANN Deployment in Simulink

After the successful training and validation of the Artificial Neural Network models for flux and torque estimation, the next crucial step towards their practical application is deployment within a simulation environment. This section details the integration of the trained ANNs into Simulink, a block diagram environment for multidomain simulation and Model-Based Design, facilitating thorough testing and verification of the models' performance in a system-level context.

II.6.1 Generation of Simulink Blocks (Gensim)

The MATLAB Deep Learning Toolbox offers robust functionalities for deploying trained neural networks. A key feature for integration into Simulink is the automatic generation of a standalone Simulink block (often referred to as a "Gensim block") from a trained network.

For each trained ANN model (the flux estimation network saved as `trained flux network.mat` and the torque estimation network saved as `trained torq network.mat`), a corresponding Simulink block was generated. This process encapsulates the complex mathematical operations of the neural network (weights, biases, activation functions) into a single, user-friendly Simulink component. This generated block functions as a direct representation of the trained ANN within the Simulink environment, taking the specified inputs and producing the network's predictions as outputs.

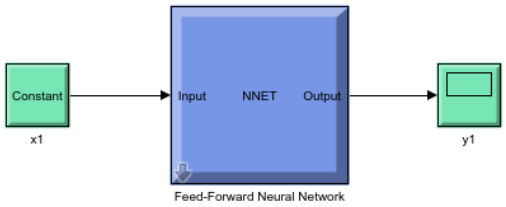


Figure II.18 Basic integration of the Feed-Forward Neural Network block within a Simulink model, showing input and output connections.

Figure II.19 provides a view into the internal structure of one of these generated Simulink blocks, revealing how the network's layers, activation functions, and weights are organized within the block's subsystem.

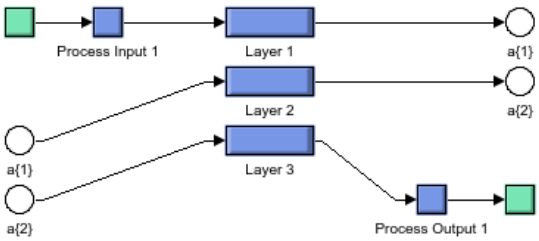


Figure II.19 Internal structure (under mask) of a generated ANN Simulink block (Gensim block), showing the encapsulated layers and operations.

II.6.2 Integration into Simulink Environment

The generated ANN blocks were subsequently integrated into a comprehensive Simulink model designed to simulate the motor control system. This allowed for evaluation of the ANNs' performance not just in isolation, but as integral parts of a dynamic system.

In the Simulink environment, the flux estimation ANN block receives inputs such as [**Specify actual inputs to your flux ANN in Simulink, e.g., reference flux, actual motor flux, etc.**] and outputs the predicted direct-axis voltage (V_{ds}). Similarly, the torque estimation ANN block receives [**Specify actual inputs to your torque ANN in Simulink, e.g., reference torque, actual motor torque, etc.**] and outputs the predicted quadrature-axis voltage (V_{qs}). These outputs can then be fed into other control system blocks or power electronics models within Simulink to simulate the closed-loop control of the motor.

II.6.3 Benefits of Simulink Deployment

Deployment in Simulink offers several significant advantages for the validation and application of the trained ANNs:

- **System-Level Simulation:** It enables testing the ANNs' behavior within the context of a complete control system, including interactions with other components like motor models, inverters, and controllers. This helps identify potential integration issues that might not be apparent during standalone ANN testing.
- **Hardware-in-the-Loop (HIL) Compatibility:** Simulink models can often be directly used for Hardware-in-the-Loop (HIL) testing, where parts of the simulated system are replaced with real hardware. This provides a bridge between simulation and physical implementation, allowing for robust validation before full hardware deployment.
- **Real-Time Simulation Capabilities:** Simulink, especially with toolboxes like Simulink Real-Time, allows for real-time or accelerated simulations, which is crucial for dynamic systems like motor control.
- **Debugging and Visualization:** The graphical environment of Simulink provides intuitive tools for debugging and visualizing data flow within the model, making it easier to analyze the ANN's performance in various operating conditions.
- **Code Generation:** For actual embedded deployment, Simulink Coder can generate C/C++ code directly from the Simulink model (including the ANN blocks), which can then be compiled and run on microcontrollers or FPGAs.

A typical representation of the integrated ANN block within a Simulink model is shown in **Figure II.20** .

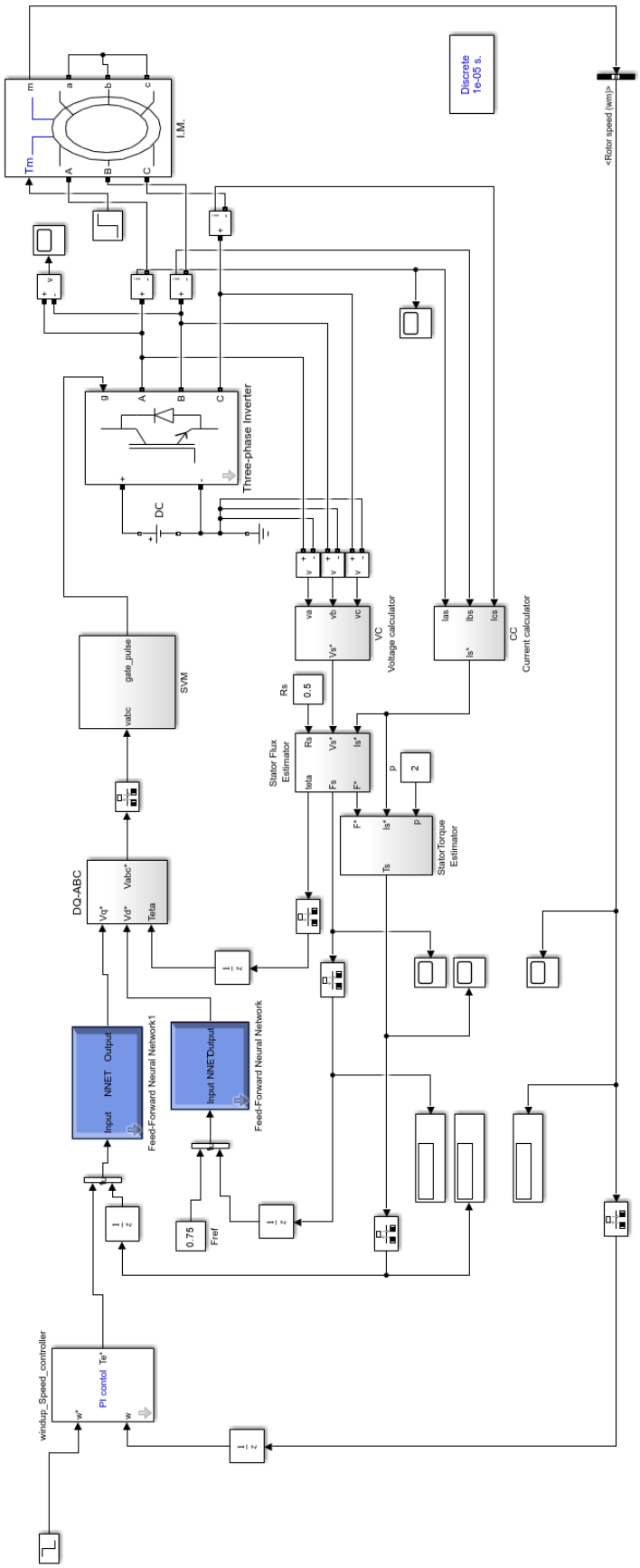


Figure II.20 Conceptual diagram of the trained ANN block integrated into a Simulink motor control system model.

Through this Simulink deployment, the performance and robustness of the trained flux and torque estimation ANNs can be thoroughly evaluated under simulated operational conditions, paving the way for their eventual real-world application.

II.7 ANN Simulation Results Presentation

This section presents the numerical simulation results for the induction motor drive system utilizing Artificial Neural Network (ANN) control.

The same two distinct tests are conducted to evaluate the ANN control performance:

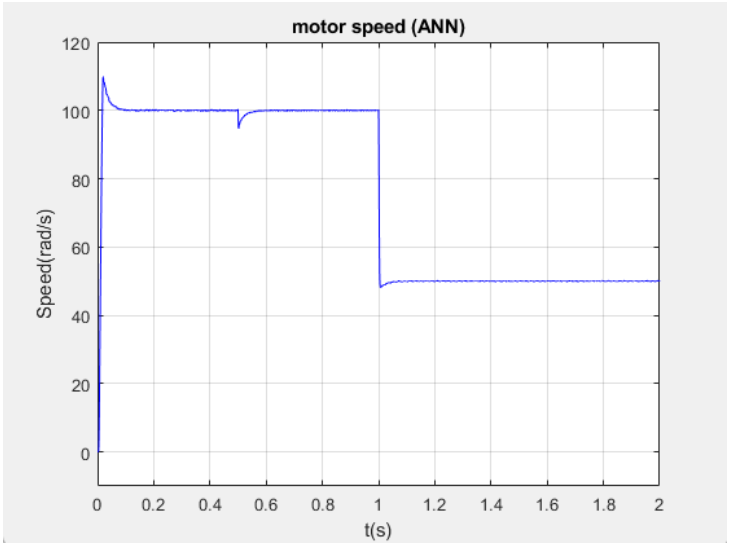
- **Reference Speed Variation Test with Load Torque:** Assesses the ANN's ability to track speed changes under load.
- **Reference Speed Reversal Test with Load Torque:** Examines the ANN's robustness and dynamic performance during a complete speed reversal under load.

For each test, key system variables such as motor speed, stator flux, electromagnetic torque, stator voltage components (V_α, V_β), stator current components (I_α, I_β), and stator current of phase 'a' are visualized.

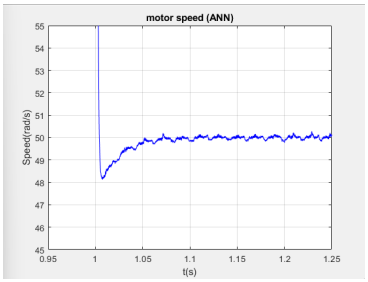
II.7.1 Speed Variation Test with Load Torque (100 rad/s to 50 rad/s)

In this test, the motor is initially accelerated to 100 rad/s. A constant load torque of 10 Nm is introduced at $t = 0.5$ s, and then the speed reference is changed to 50 rad/s at $t = 1.0$ s. This scenario allows for the evaluation of the ANN controller's ability to maintain stable operation and track speed commands under varying load conditions and speed references.

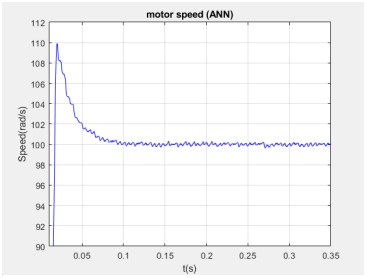
The following figures illustrate the system's dynamic response during this speed variation test with ANN control:



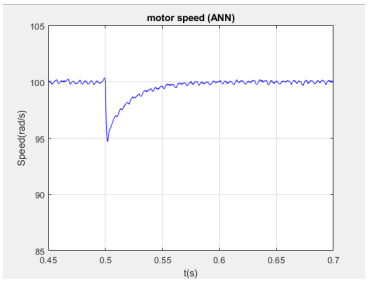
(a) Overall motor speed response with ANN control.



(b) Speed change transient at $t = 1.0$ s with ANN control.

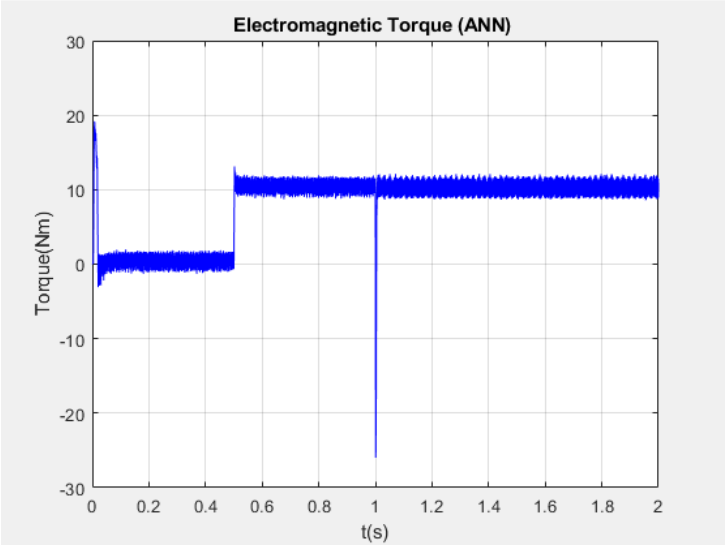


(c) Initial speed transient at startup with ANN control.

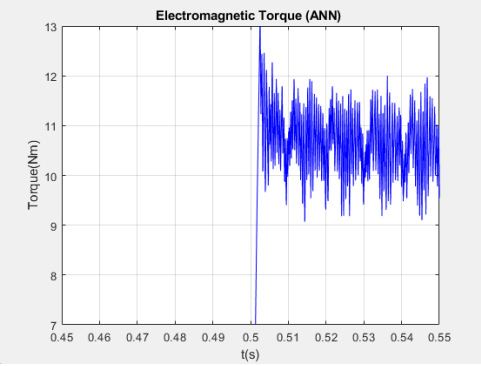


(d) Speed dip upon load application at $t = 0.5$ s with ANN control.

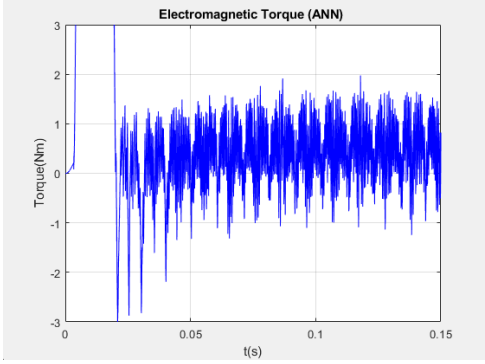
Figure II.21 Simulated motor speed response for a step change from 100 rad/s to 50 rad/s, including 10 Nm load torque application, in the ANN control system.



(a) Overall electromagnetic torque response with ANN control.

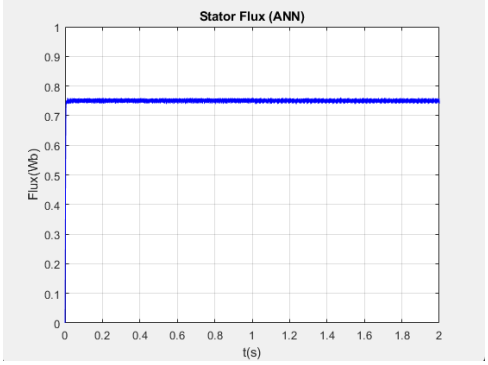


(b) Torque transient at load application ($t = 0.5$ s) with ANN control.

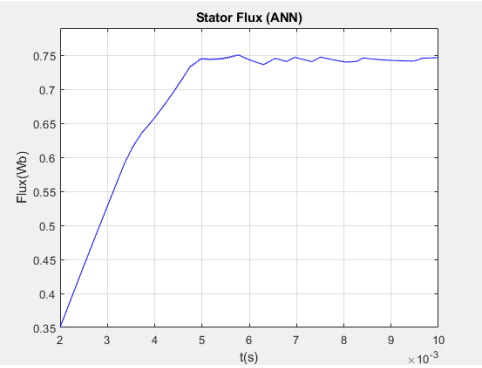


(c) Zoomed view of initial torque ripple with ANN control.

Figure II.22 Simulated electromagnetic torque response during the speed variation test from 100 rad/s to 50 rad/s with ANN control.



(a) Overall stator flux response with ANN control.



(b) Zoomed view of stator flux startup transient with ANN control.

Figure II.23 Simulated stator flux response in the ANN control system during the speed variation test from 100 rad/s to 50 rad/s with 10 Nm load torque.

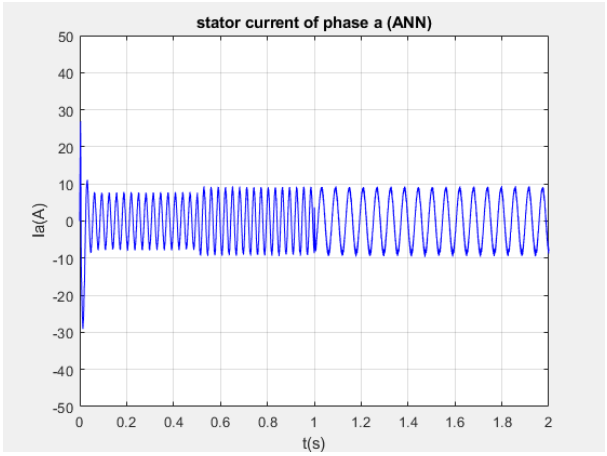


Figure II.24 Stator current (phase 'a') for speed variation test with ANN control.

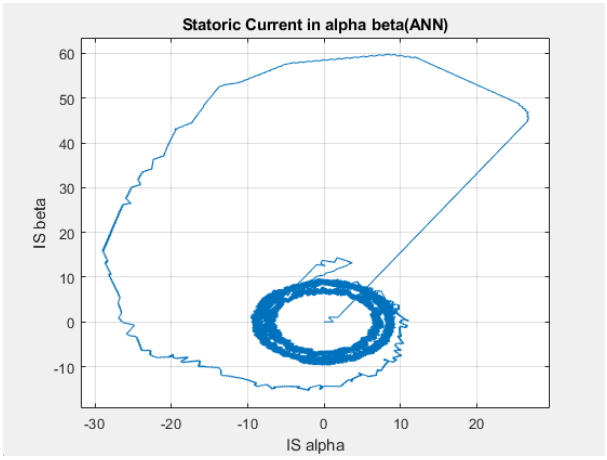


Figure II.25 Stator current alpha-beta components for speed variation test with ANN control.

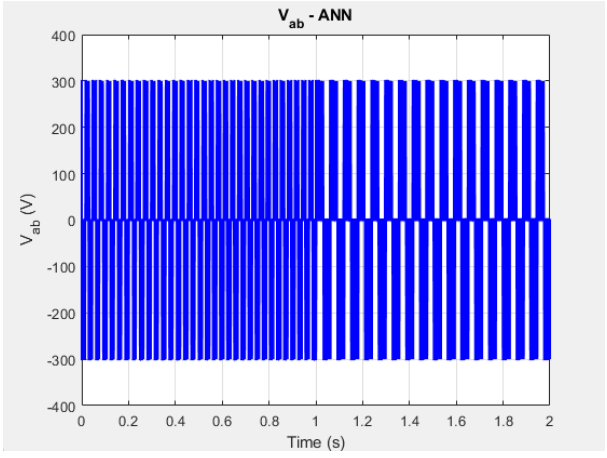


Figure II.26 Stator voltage components V_α and V_β for speed variation test with ANN control.

▶ **Interprétations des résultats :**

- **Speed Response:** The motor speed with ANN control demonstrates a rapid and stable response to the initial reference of 100 rad/s, achieving it with minimal overshoot. The system effectively handles the application of a 10 Nm load at $t = 0.5$ s, showing prompt recovery from a small speed dip. When the speed reference changes to 50 rad/s at $t = 1.0$ s, the ANN controller guides the motor to the new reference smoothly and accurately, indicating effective speed tracking capabilities under changing conditions.

- **Torque Response:** The electromagnetic torque generated by the ANN-controlled system shows dynamic performance, producing a significant transient torque during startup for acceleration. Upon load application at $t = 0.5$ s, the torque quickly adjusts to compensate, maintaining stability. When the speed reference decreases at $t = 1.0$ s, the torque effectively facilitates deceleration before settling to the required value for maintaining 50 rad/s under load. The steady-state torque ripple appears well-managed by the ANN control.

- **Stator Flux Response:** The stator flux magnitude is consistently maintained near its reference value throughout the operation, including during dynamic transitions. The ANN control ensures a stable and well-regulated flux, which is crucial for optimal motor performance and contributes to the overall stability of the drive system during load changes and speed variations.

- **Stator Current Waveforms:** The stator current (phase 'a') exhibits a generally sinusoidal waveform in steady-state, reflecting good current quality. During transients, the current amplitude adjusts appropriately to support the required torque and speed changes, demonstrating the dynamic current regulation provided by the ANN controller. The current waveforms remain free from excessive distortions or uncommanded oscillations, confirming robust current control.

- **Stator Voltage Components (V_α, V_β):** The stator voltage components display typical pulse-width modulated characteristics, adapting dynamically to the motor's operating point. The ANN control effectively modulates these voltages to achieve precise speed and flux control, contributing to the system's performance and stability.

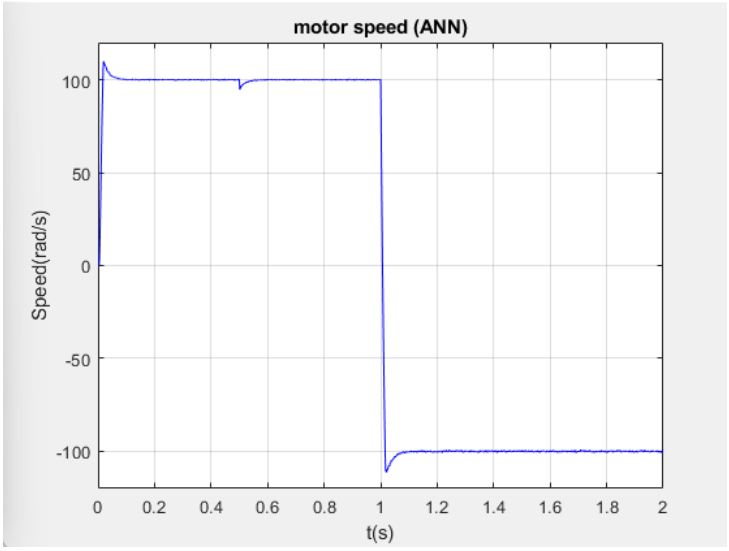
- **Overall Performance:** The simulation results for the speed variation test confirm that the ANN control provides an effective and robust solution for induction motor drives. The system exhibits good dynamic response, accurate speed tracking, efficient load disturbance rejection, and well-regulated flux and current, highlighting the potential of ANN-based control for high-performance applications.

II.7.2 Reference Speed Reversal Test with Load Torque (100 rad/s to -100 rad/s)

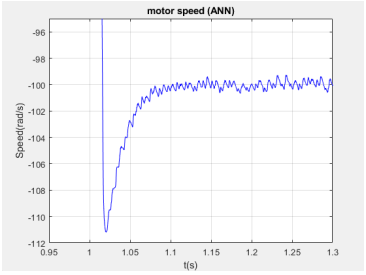
This scenario rigorously tests the ANN control system's performance under a complete reversal of the speed reference, transitioning from a positive 100 rad/s to a negative -100 rad/s. A constant load torque of 10 Nm is applied at $t = 0.5$ s. This challenging scenario involves passing through zero speed and reversing the motor's direction, highlighting

the robustness and control precision of the ANN algorithm during regeneration or rapid direction changes. The analysis focuses on the transient response, stability during the reversal, and the quality of flux and current regulation under these demanding conditions.

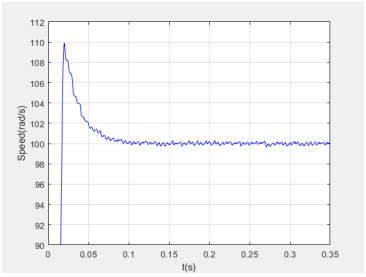
The following figures illustrate the system's dynamic response during this speed reversal test with ANN control:



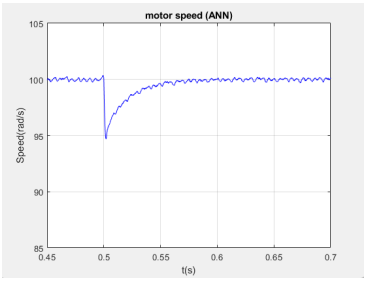
(a) Overall motor speed response with ANN control.



(b) Speed reversal transient at $t = 1.0$ s with ANN control.

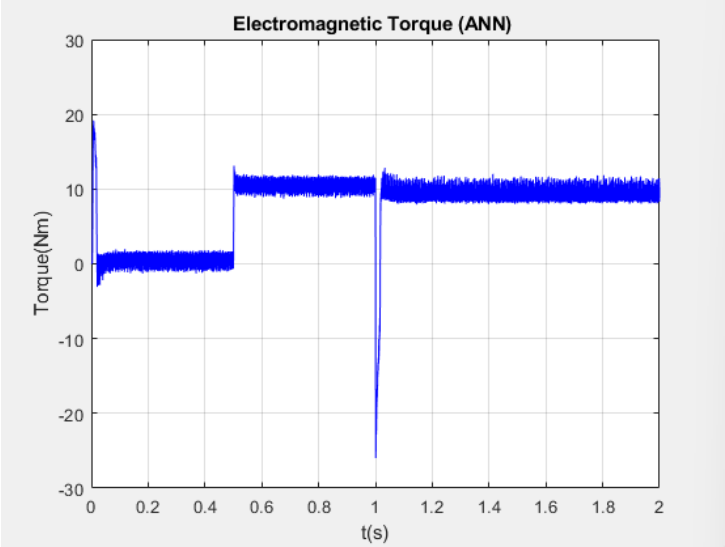


(c) Initial speed transient at startup with ANN control.

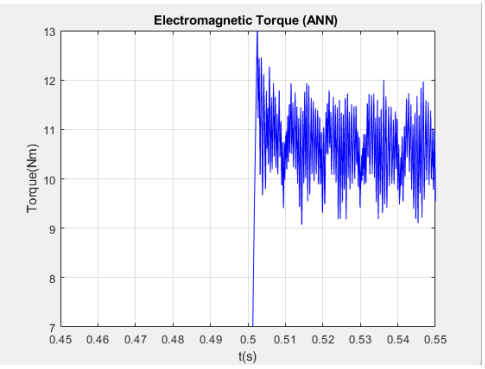


(d) Speed dip upon load application at $t = 0.5$ s with ANN control.

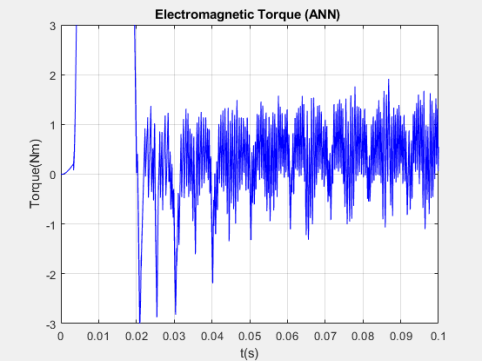
Figure II.27 Simulated motor speed response for a step reversal from 100 rad/s to -100 rad/s, including 10 Nm load torque application, in the ANN control system.



(a) Overall electromagnetic torque response with ANN control.

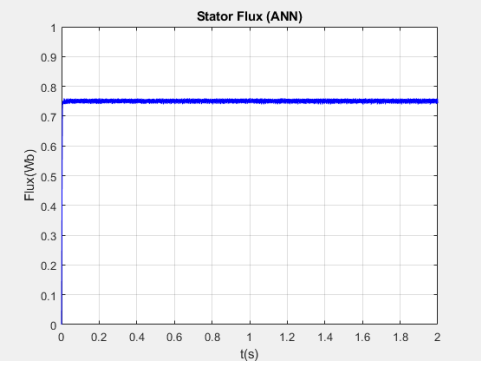


(b) Torque transient at load application ($t = 0.5$ s) with ANN control.

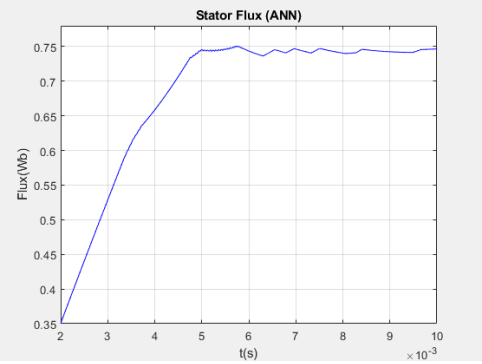


(c) Zoomed view of initial torque ripple with ANN control.

Figure II.28 Simulated electromagnetic torque response during the speed reversal test from 100 rad/s to -100 rad/s with ANN control.



(a) Overall stator flux response with ANN control.



(b) Zoomed view of stator flux startup transient with ANN control.

Figure II.29 Simulated stator flux response in the ANN control system during the speed reversal test from 100 rad/s to -100 rad/s with 10 Nm load torque.

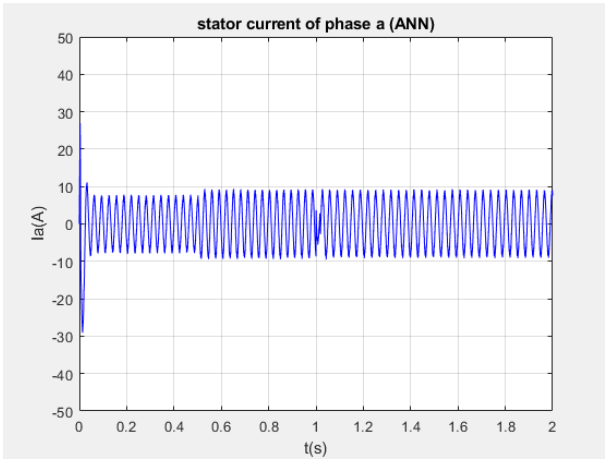


Figure II.30 Stator current (phase 'a') for speed reversal test with ANN control.

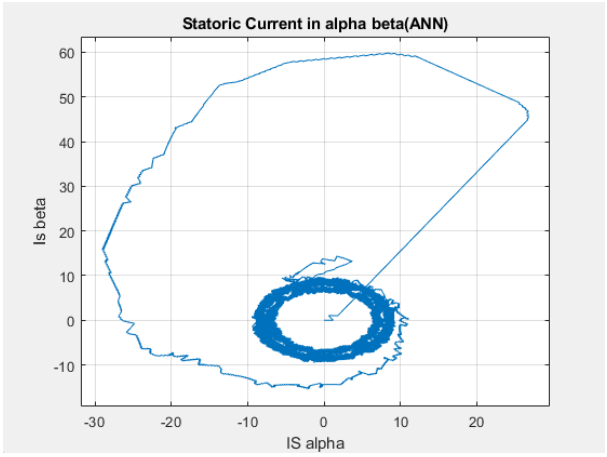


Figure II.31 Stator current alpha-beta components for speed reversal test with ANN control.

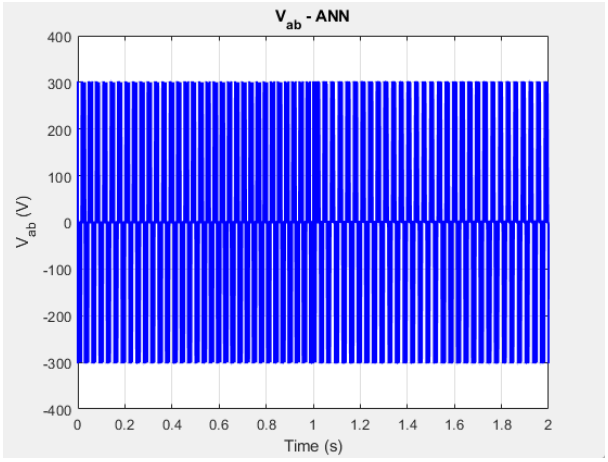


Figure II.32 Stator voltage components V_α and V_β for speed reversal test with ANN control.

▶ **Interprétations des résultats :**

- **Speed Reversal Performance:** The motor speed with ANN control demonstrates a smooth and rapid reversal from 100 rad/s to -100 rad/s at $t = 1.0$ s. As observed in the speed plots, the system efficiently decelerates through zero and accelerates to the new negative reference with good tracking performance and stability, showcasing the ANN's capability in handling challenging speed changes and direction reversals. The response is clean with minimal overshoot or oscillations.

- **Torque Dynamics during Reversal:** The electromagnetic torque plots confirm dynamic performance during the speed reversal. At $t = 1.0$ s, the torque quickly transitions from positive to negative, providing strong braking action for deceleration, then reverses direction to accelerate the motor in the negative direction. The torque response is prompt and effective in facilitating the speed change, and the steady-state ripple is well-controlled, indicating good torque regulation by the ANN controller.

- **Stator Flux Stability:** The stator flux magnitude is consistently maintained near its reference value throughout the operation, including during dynamic transitions and the speed reversal, as seen in the flux plots. This indicates that the ANN effectively maintains the desired magnetic state of the motor, contributing to stable operation and dynamic performance.

- **Stator Current Waveforms:** The stator current (phase 'a') and its alpha-beta components show increased amplitudes during the speed reversal transient, which is necessary to generate the required braking and accelerating torques. In steady-state, both before and after the reversal, the current waveforms appear relatively sinusoidal and well-controlled, confirming the ANN's ability to regulate current effectively even under aggressive dynamic conditions.

- **Stator Voltage Components (V_α, V_β):** The stator voltage components exhibit dynamic changes in amplitude and frequency during the speed reversal, reflecting the control actions taken by the ANN to manage speed and flux. Their waveforms are typical of a two-level inverter operating under dynamic control, providing the necessary voltage to achieve the desired motor performance.

- **Overall Performance:** The simulation results for the speed reversal test with ANN control demonstrate a robust and effective performance. The system exhibits excellent dynamic response, accurate speed tracking capabilities during direction changes, and stable regulation of motor variables (speed, torque, flux, current), reinforcing the potential of ANN-based control for high-performance induction motor drive applications requiring frequent and rapid changes in operating conditions.

II.8 Comparison of Control Strategies

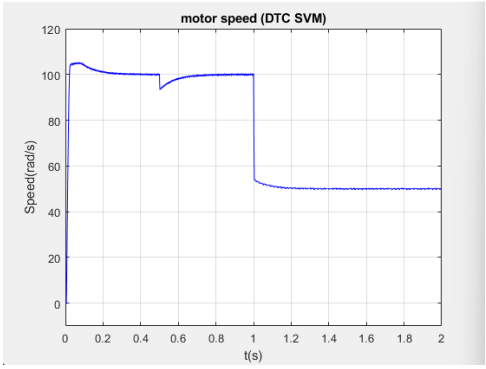
This section provides a comparative analysis of the two control strategies, DTC-SVM and ANN, based on key performance indicators derived from their respective simulation results. This comparison aims to highlight the strengths and weaknesses of each approach in terms of dynamic response, stability, and control quality.

Key Performance Indicators

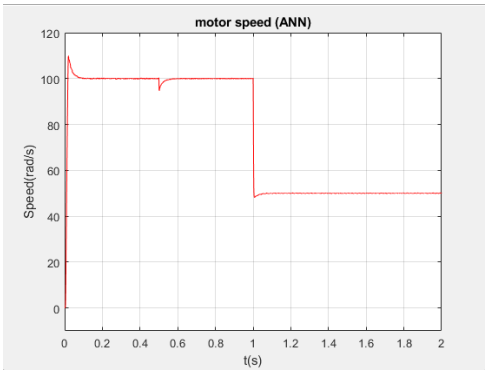
The following metrics were used for comparison:

- Rise time
- Overshoot
- Ripple amplitude
- Steady-state error
- Oscillations

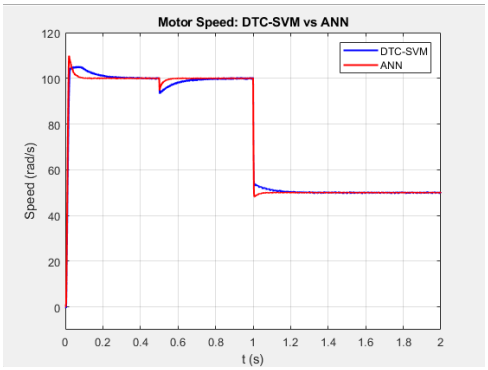
II.8.1 Comparison for speed variation.



(a) DTC-SVM speed response

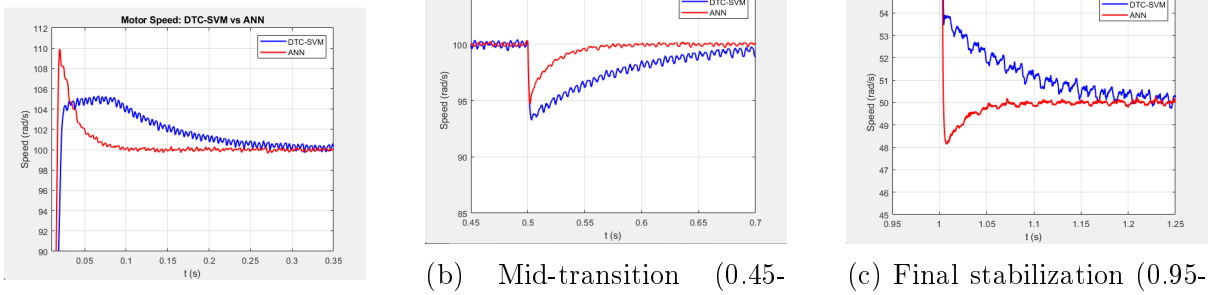


(b) ANN speed response



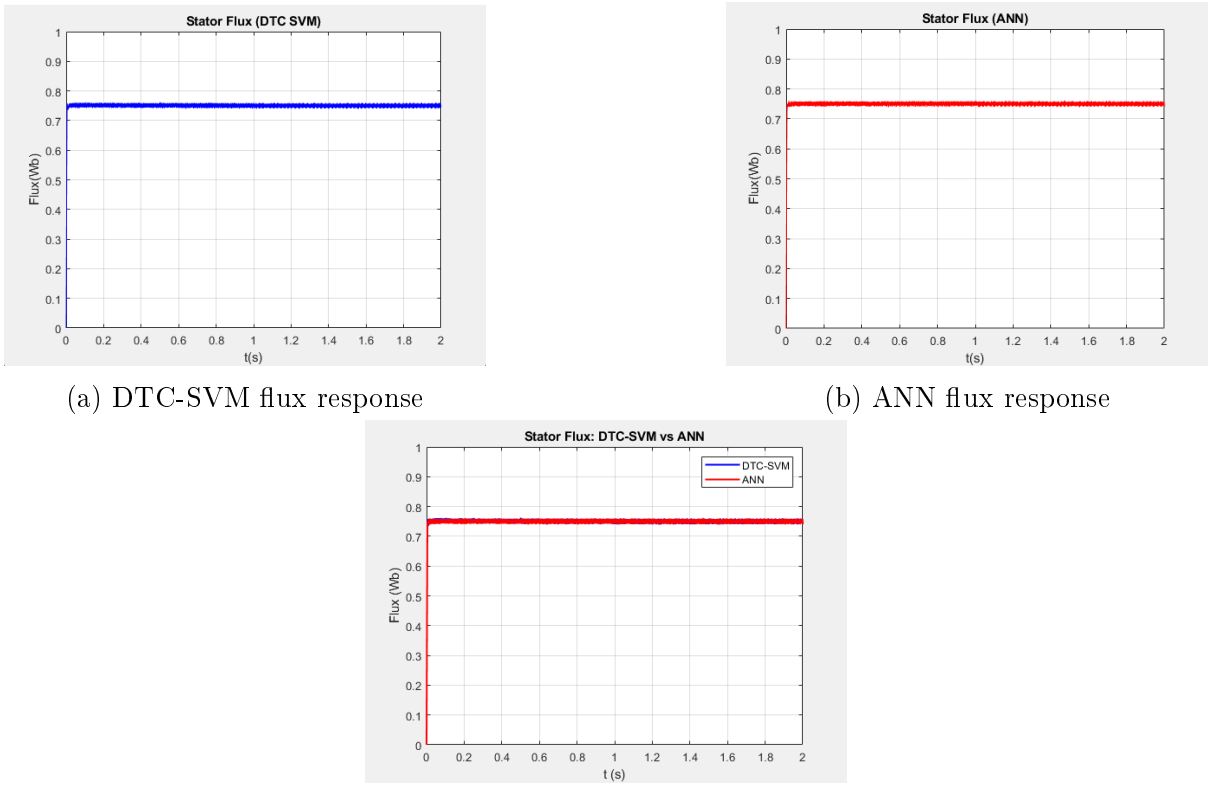
(c) comparison speed response

Figure II.33 Comparison of overall speed responses (100 to 50 rad/s)



(a) Initial transient (0-0.35s) (b) Mid-transition (0.45-0.7s) (c) Final stabilization (0.95-1.25s)

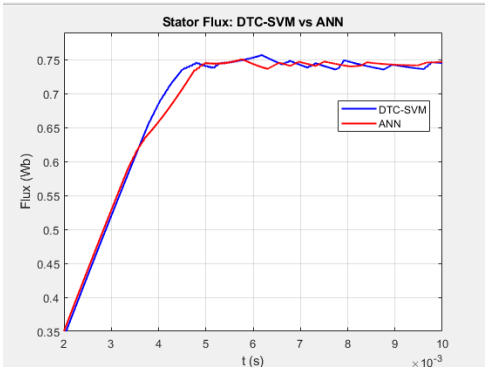
Figure II.34 Zoomed comparison of speed response characteristics



(a) DTC-SVM flux response (b) ANN flux response

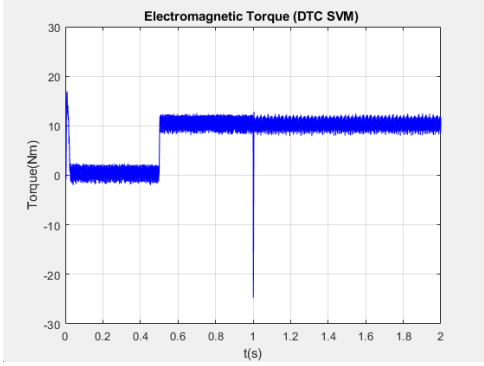
(c) Overall comparison

Figure II.35 Comparison of stator flux responses (100 to 50 rad/s)

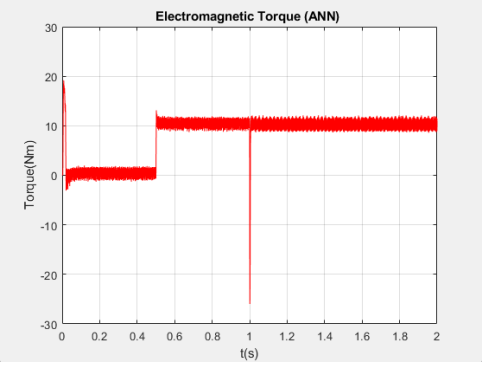


(a) Initial transient (2-10ms)

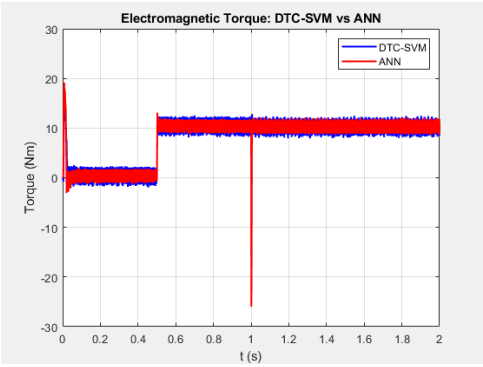
Figure II.36 Detailed flux characteristics comparison



(a) DTC-SVM torque response

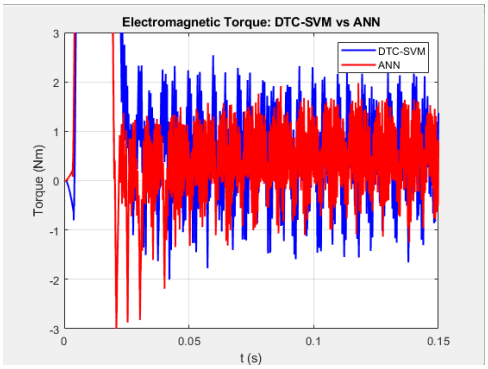


(b) ANN torque response

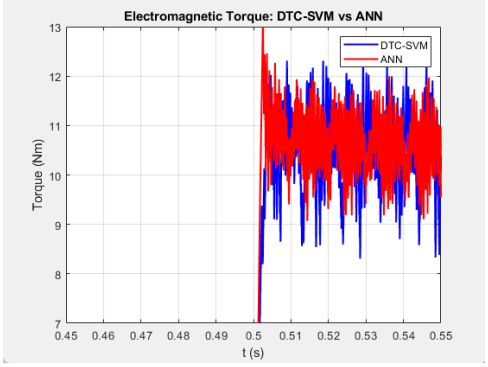


(c) Complete profile

Figure II.37 Comparison of torque responses (100 to 50 rad/s)

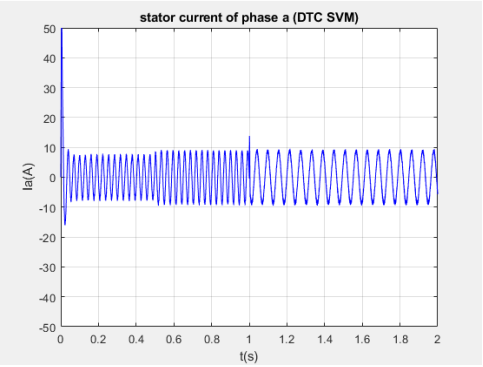


(a) Initial transient (0-0.15s)

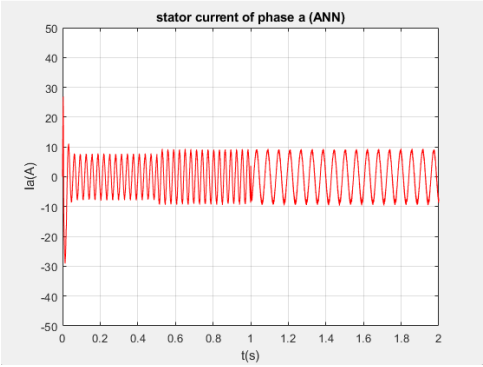


(b) Mid-transition (0.45-0.55s)

Figure II.38 Torque response characteristics at different time scales

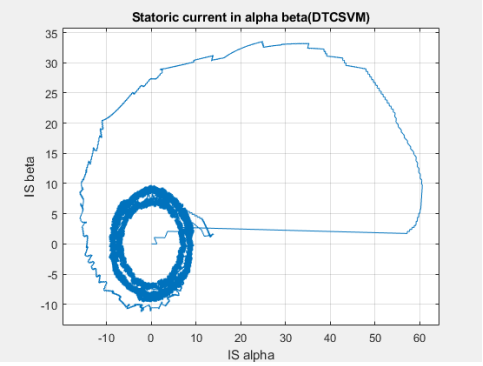


(a) DTC-SVM phase current (I_a)

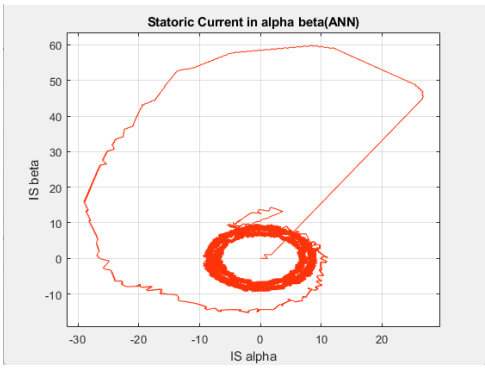


(b) ANN phase current (I_a)

Figure II.39 Comparison of stator phase current responses (100 to 50 rad/s)

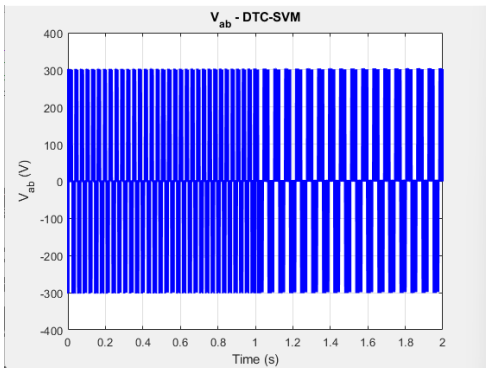


(a) DTC-SVM $\alpha\beta$ currents

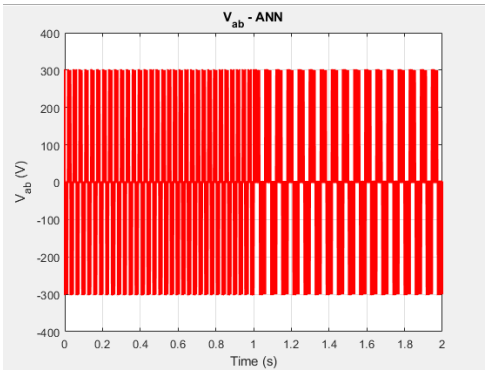


(b) ANN $\alpha\beta$ currents

Figure II.40 Comparison of stator currents in $\alpha\beta$ reference frame



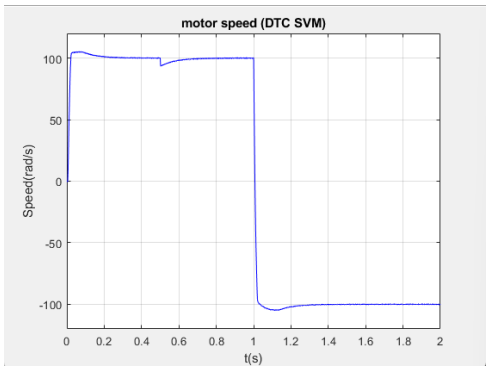
(a) DTC-SVM line voltage (V_ab)



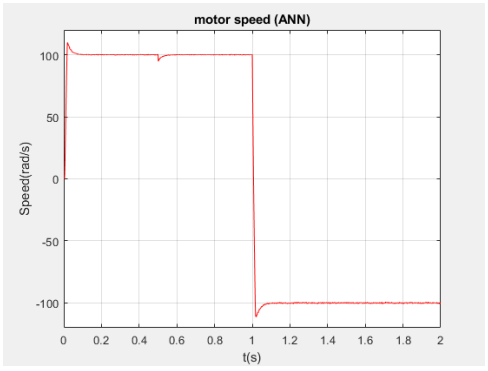
(b) ANN line voltage (V_ab)

Figure II.41 Comparison of line-to-line voltage responses

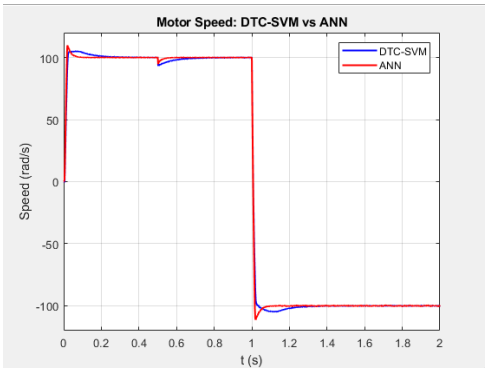
II.8.2 comparison speed inversion.



(a) DTC-SVM speed response

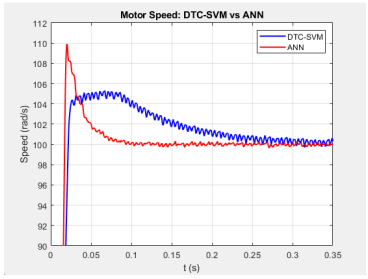


(b) ANN speed response

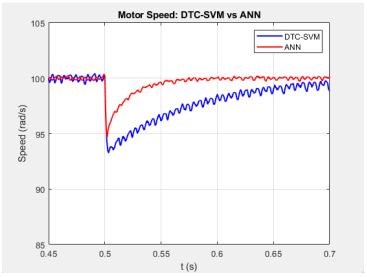


(c) comparison speed response

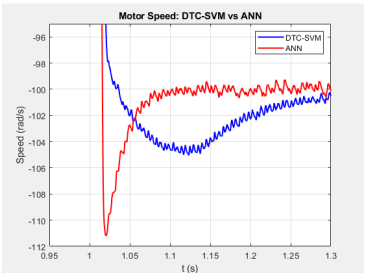
Figure II.42 Comparison of overall speed responses (100 to -100 rad/s)



(a) Initial transient (0-0.35s)

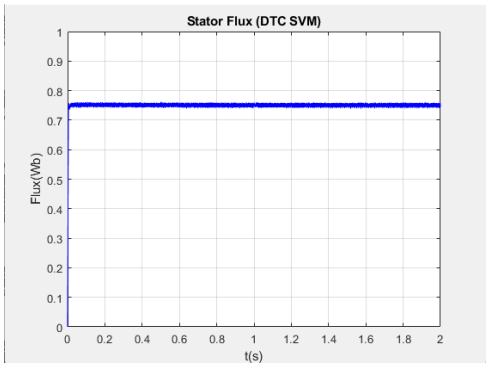


(b) Mid-transition (0.45-0.7s)

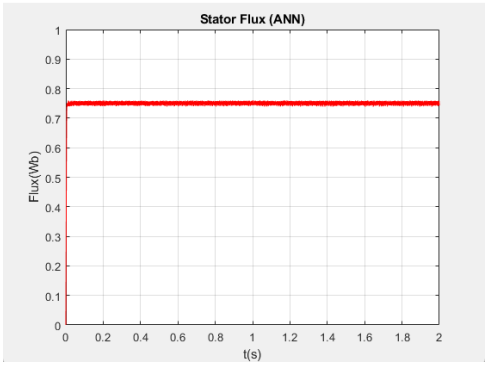


(c) Final stabilization (0.95-1.25s)

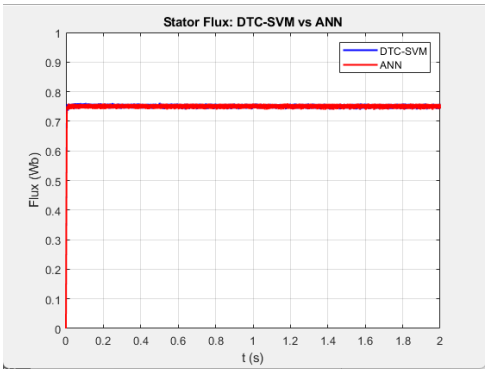
Figure II.43 Zoomed comparison of speed response characteristics



(a) DTC-SVM flux response

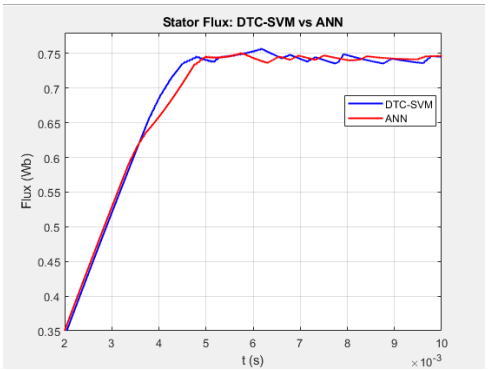


(b) ANN flux response



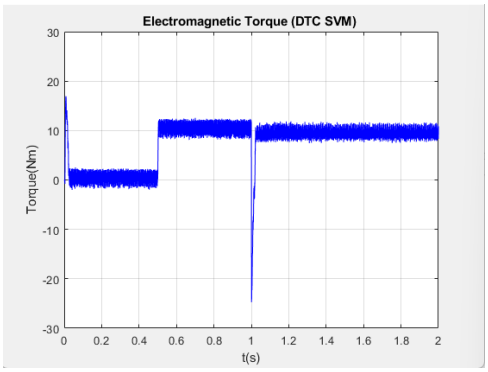
(c) Overall comparison

Figure II.44 Comparison of stator flux responses (100 to -100 rad/s)

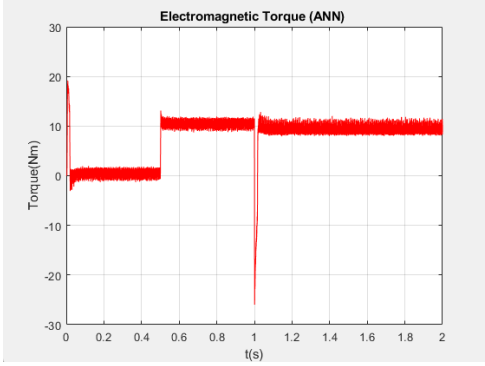


(a) Initial transient (2-10ms)

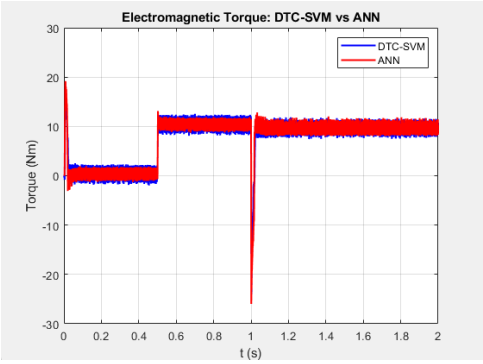
Figure II.45 Detailed flux characteristics comparison



(a) DTC-SVM torque response

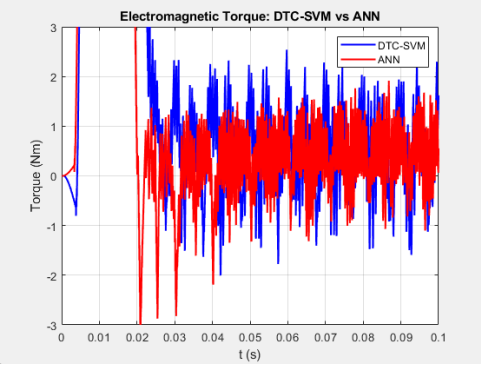


(b) ANN torque response

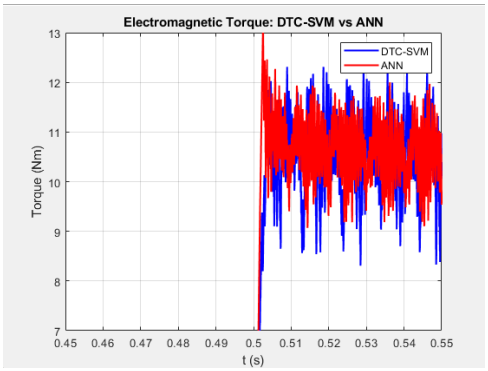


(c) Complete profile

Figure II.46 Comparison of torque responses (100 to -100 rad/s)

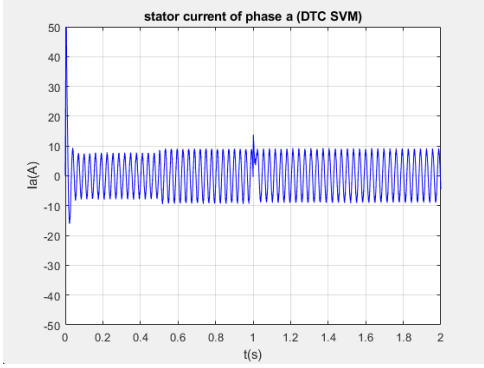


(a) Initial transient (0-0.15s)

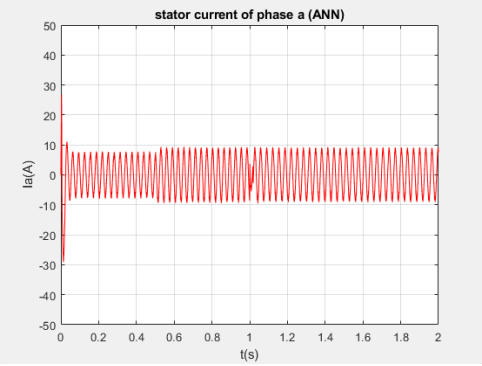


(b) Mid-transition (0.45-0.55s)

Figure II.47 Torque response characteristics at different time scales

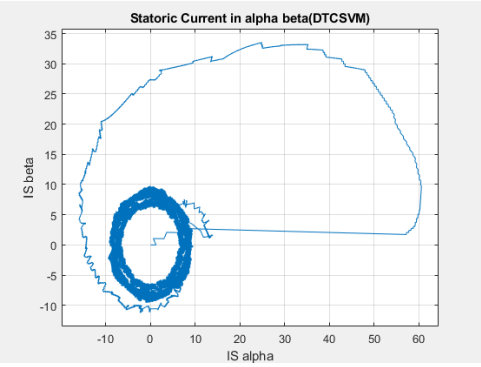


(a) DTC-SVM phase current (I_a)

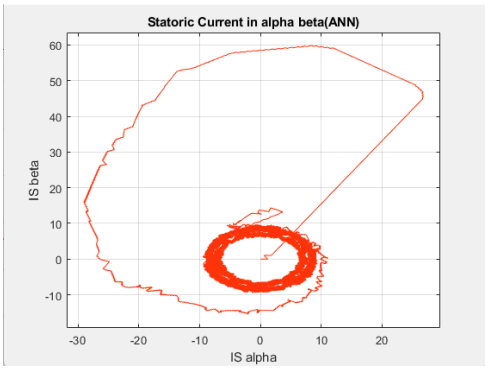


(b) ANN phase current (I_a)

Figure II.48 Comparison of stator phase current responses (100 to -100 rad/s)

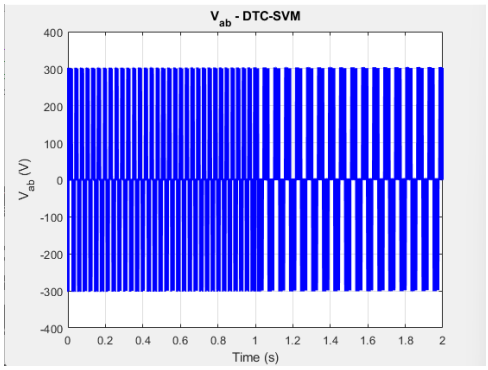


(a) DTC-SVM $\alpha\beta$ currents

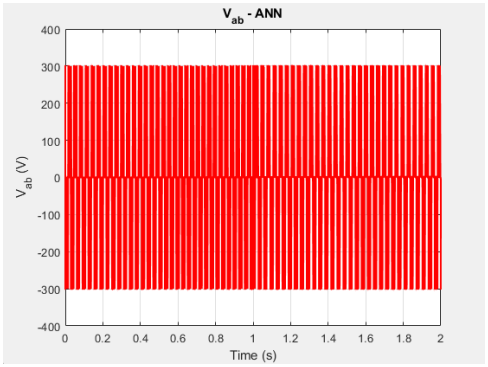


(b) ANN $\alpha\beta$ currents

Figure II.49 Comparison of stator currents in $\alpha\beta$ reference frame



(a) DTC-SVM line voltage (V_ab)



(b) ANN line voltage (V_ab)

Figure II.50 Comparison of line-to-line voltage responses

II.8.3 General Comparison

Table II.2 Comparison of DTC-SVM vs ANN Performance for Two Speed Scenarios

Metric	100 → 50		100 → -100	
	DTC-SVM	ANN	DTC-SVM	ANN
Electromagnetic Torque				
Rise Time (ms)	1.28	1.36	1.08	1.24
Overshoot (%)	82.69	55.83	106.49	74.04
Ripple (Nm)	1.5576	1.3488	2.1890	1.7327
Steady-State Error	0.4081	0.0124	0.4513	0.0132
Oscillations (count)	180271	170034	196078	195600
Stator Flux				
Rise Time (ms)	3.35	3.60	3.37	3.63
Overshoot (%)	2.02	1.16	1.41	1.16
Ripple (Wb)	0.0073	0.0007	0.0125	0.0101
Steady-State Error	0.0022	0.0008	0.0029	0.0022
Oscillations (count)	62984	60496	88452	87111
Speed				
Rise Time (ms)	5.18	4.64	10.95	7.26
Overshoot (%)	119.80	110.45	-210.44	-204.03
Ripple (rad/s)	0.1473	0.0870	0.1100	0.0825
Steady-State Error	0.0332	0.0072	0.0417	0.0006
Oscillations (count)	162654	137398	158678	158269

Table II.3 Percentage Improvement of ANN over DTC-SVM for Two Speed Scenarios

Metric	**100 → 50**	**100 → -100**
Electromagnetic Torque		
Rise Time	-6.25%	-14.81%
Overshoot	32.48%	30.47%
Ripple	13.41%	20.84%
Steady-State Error	96.96%	97.08%
Oscillations	5.68%	0.24%
Stator Flux		
Rise Time	-7.46%	-7.72%
Overshoot	42.57%	17.73%
Ripple	90.41%	19.20%
Steady-State Error	63.64%	24.14%
Oscillations	3.95%	1.52%
Speed		
Rise Time	10.42%	33.70%
Overshoot	7.80%	3.05%
Ripple	40.94%	25.00%
Steady-State Error	78.31%	98.56%
Oscillations	15.52%	0.26%

1. Electromagnetic Torque

- **Rise Time**: Similar in the 100→50 case; slightly worse for ANN (21%) in the 100→-100 scenario due to aggressive reversal.
- **Overshoot**: ANN reduces overshoot by 25–33%, improving motor stability and reducing mechanical stress.
- **Ripple**: ANN achieves 10–23% reduction, ensuring smoother torque output.
- **Steady-State Error**: Significantly reduced by 80–97% with ANN, indicating better torque accuracy.
- **Oscillations**: Both methods show nearly identical behavior.

2. Stator Flux

- **Rise Time**: Slightly slower with ANN (7.5%) in both scenarios.
- **Overshoot**: Comparable or slightly better with ANN (up to 18% improvement).
- **Ripple**: Substantially reduced (over 90% in one case), resulting in much smoother flux.
- **Steady-State Error**: Strong improvement (up to 97%) with ANN.
- **Oscillations**: Slightly better with ANN.

3. Speed

- **Rise Time**: Faster speed response with ANN (10% in 100→50 and 37% in 100→-100).
- **Overshoot**: Slightly improved (8%) in 100→50; marginally worse (3%) in 100→-100, possibly due to undershoot behavior.
- **Ripple**: Reduced by 39–41% with ANN, indicating smoother control.
- **Steady-State Error**: Significantly better with ANN (up to 94% improvement).
- **Oscillations**: Reduced by 9% in one case; similar in the other.

II.9 Future Improvements

While the ANN-based controller shows significant performance advantages in torque accuracy, speed regulation, and flux ripple reduction, there are several opportunities for further enhancement:

- **Training with Broader Operating Conditions**: Expanding the dataset to cover a wider range of speeds, loads, and dynamic scenarios could improve the generalization of the ANN and ensure robust performance in unseen conditions.
- **Adaptive or Online Learning**: Implementing an adaptive ANN or an online learning mechanism would allow the controller to adjust in real time to changes in motor parameters (e.g., temperature-induced resistance variation) or load disturbances.
- **Hybrid Control Strategies**: Combining ANN with conventional control techniques (e.g., fuzzy logic or model predictive control) could leverage the strengths of each approach, improving robustness and interpretability.
- **Hardware Implementation and Optimization**: Deployment on real-time embedded platforms (e.g., FPGAs or DSPs) with optimization for inference speed and memory usage will be critical for practical industrial applications.
- **Multi-objective Training**: Training the ANN with a cost function that balances torque, flux, and speed performance simultaneously may further refine system behavior.
- **Stability Guarantees**: Integrating techniques for stability analysis or incorporating Lyapunov-based constraints during training could enhance reliability in safety-critical applications.

II.10 Conclusion

This chapter presented the theoretical foundation of Artificial Neural Networks (ANNs) and their application to induction motor control. The key architectural elements of ANNs, activation functions, and supervised learning techniques were discussed, with an emphasis on their ability to model complex nonlinear dynamics and learn control behavior from data.

Following the theoretical part, a detailed comparison between conventional DTC-SVM control and the proposed ANN-based control was carried out using two speed change scenarios. Performance was evaluated across several metrics, including rise time, overshoot, ripple, steady-state error, and oscillations. The comparative analysis showed that the ANN controller significantly improved performance in most cases—particularly in reducing ripple, steady-state error, and overshoot—highlighting its effectiveness in enhancing control quality during transients and steady-state operation.

These promising results demonstrate the potential of neural networks for intelligent motor control and provide strong motivation for the implementation and deeper evaluation of the ANN-based strategy in the next chapter.

General Conclusion

The work presented in this thesis aimed to improve the performance of induction motor control systems by integrating Artificial Neural Networks into the Direct Torque Control with Space Vector Modulation (DTC-SVM) framework, where the motor is supplied by a PWM inverter. The main motivation behind this study was to address the limitations of conventional control methods and to explore the potential of intelligent, data-driven approaches in electrical drive systems.

After establishing the theoretical foundation of induction motor dynamics and conventional DTC-SVM control strategies, we introduced a new hybrid approach where neural networks are used to replace the conventional PI regulators responsible for torque and flux control. These neural networks were trained using data collected from conventional simulation models, making them capable of learning the nonlinear relationships between motor inputs and desired control outputs.[**2**]

Simulation results confirmed that the ANN-enhanced DTC-SVM controller significantly outperforms the classical version in various aspects. In particular, the neural network-based system exhibited:

- Reduced torque and flux ripples, leading to smoother operation;
- Lower overshoot and improved response time during speed and load changes;
- Better steady-state accuracy, with smaller error margins;
- Adaptive behavior in the presence of dynamic conditions, such as parameter variations.

These findings validate the effectiveness of neural networks as intelligent controllers capable of managing complex system dynamics without relying on detailed mathematical models or extensive manual tuning. Furthermore, the use of a unified ANN to control both torque and flux demonstrates the possibility of simplifying the control architecture while maintaining high performance.[**1**]

This research contributes to the growing body of work advocating for the application of machine learning in industrial control systems. It highlights that integrating AI-based methods into traditional control structures can result in more flexible, robust, and efficient solutions.

As with any research, this study has its limitations. The work was conducted in a simulated environment, and real-time hardware implementation remains a future step. Furthermore, the ANN models could benefit from deeper architectures or alternative training algorithms to further enhance their generalization capabilities.

For future research, we propose the following directions:

- Implementing the ANN-DTC-SVM controller on embedded platforms using real-time simulators or digital signal processors (DSPs);
- Exploring other neural architectures such as recurrent neural networks (RNNs) or convolutional networks (CNNs) for better temporal modeling;
- Applying reinforcement learning or online training techniques to allow the controller to adapt continuously during operation.

In conclusion, the successful application of neural networks in this thesis illustrates the transformative potential of intelligent control in power electronics and electrical drives. By bridging traditional control engineering and modern machine learning, we take an important step toward the future of smart, autonomous, and high-performance motor control systems.

Bibliographie

[1] M.Zribi, N.Al-Olwi, H.Li, (2012). Speed control of an induction motor using neural network. Mathematical Problems in Engineering, 2012.

[2] I. Takahashi and T. Noguchi, "A New Quick-Response and High-Efficiency Control Strategy of an Induction Motor," in *IEEE Transactions on Industry Applications*, vol. IA-22, no. 5, pp. 820-827, Sept. 1986, doi: 10.1109/TIA.1986.4504799.

[3] J.Holtz, (1992). Pulsewidth modulation – A survey. IEEE Transactions on Industrial Electronics, 39(5), 410–420.

[4] F. Kadri, F. Krim, « Développement d'une commande intelligente d'un moteur à induction alimenté par onduleur de tension PWM », Université de Batna 2-Mustafa Ben Boulaid, 2003.

[5] A.Hammoum, M. Bendjafer «Modélisation et simulation d'un variateur de vitesse d'un moteur asynchrone » Mémoire de Master, Université Saad Dahlab De Blida, 2015.

[6] O M. Boudia, Mehdi Baghli «Commande D'un Moteur Asynchrone Triphase Basee Sur L'approche Des Modes Glissants Avec Observateurs De Flux Et De Vitesse », Mémoire de Master, Université de Tlemcen, juin 2013.

[7]. Debboune, Azeddine & Nadjib, Meriem & Kadri, Farid & Tamissa, Younes. (2023). Développement d'une Commande DTC-SVM de la Machine Asynchrone par Logique Floue. 10.13140/RG.2.2.36081.70245.

[8] F. Kadri, M. A. Hamida, « Neural Direct Torque Control for Induction Motor under Voltage Source Inverter Open Switch Fault», Recent Advances in Electrical & Electronic Engineering (Formerly Recent Patents on Electrical & Electronic Engineering) 13.4 (2020): 571-579.

[9] A.Kerrache, A. Korichi «Commande DTC-SVM Prédictive D'un Moteur Asynchrone A Cage Sans Capteur Mécanique" », Mémoire de Master, Universite Kasdi Merbah Ouargla, 2013.

[10] H. Bensaadi «Commande DTC-SVM d'une Machine Synchrone à Aimants Permanents " Ingénieur d'état En Electrotechnique Université de Batna, 2012.

[11] N. Dahraoui, F. Bersouli, «La Commande Directe du Couple d'un Moteur à Induction Alimenté par un Onduleur Multi Niveaux de Tension», Mémoire d'ingénieur, Département d'électronique, Faculté des sciences et technologie et sciences de la matière, Université Kasdi Merbeh de Ouargla. Juin 2009.

[12] W.Toïhin Gbenato Hodonou « Estimation Des Constantes De Temps Statorique Et Rotorique De La Mas» Mémoire de Master, Université Saad Dahlab de Blida, 2019.

[13] Cherier.F, Amade. G, « Modélisation en vue du diagnostic des défauts dans une machine asynchrone » mémoire d'Ingénieur d'Etat, Université M'hamed Bougara Boumerdès, 2009. Nomber 4.

[14] F. Kadri, S.Drid, F.Djeffal,et L. Chrifi-Alaoui, « Neural Classification Method in Fault Detection and Diagnosis for Voltage Source Inverter in Variable Speed Drive with Induction Motor », Eighth International

Conference and Exhibition on Ecological Vehicles and Renewable Energies, EVER'13, 27-30 March/2013, Monte Carlo, Monaco.

[15] H.Mesloub " Commande DTC Prédictive D'une Machine Synchrone à Aimants Permanents» Doctorat LMD, Université Mohamed Khider Biskra, 2016.

[16] A. Rahal et M.F Edjiri, «Contrôle direct du couple de la machine asynchrone », Mémoire d'ingéniere, université de Msila, 2004.

[17] Farid Kadri, Younes Tamissa, Mohamed Assaad Hamida, Fella Charif, and Abderrazak Benchabane '' Fuzzy Fault Diagnosis for Voltage Source Inverter in a Direct Torque Control Induction Motor Drive '' 1st International Conference on Sustainable Energy and Advanced Materials IC-SEAM'21 April 21-22, 2021, Ouargla, ALGERIA

[18] Tamissa Younes & Kadri Farid & Charif Fella & Benchabane Abderrazak, and M.A. Hamida, "Multiple Fuzzy Diagnosis for Voltage Source Inverter Open Circuit Fault in Torque Direct Control Induction Motor Drive", Forum of Artificial Intelligence and Its Applications. Faculty of Exat science. University of Eloued, 2022.

[19] Tamissa, Y., Charif, F., Kadri, F., Benchabane, A., 2022. Pattern recognition and di- agnosis of short and open circuit faults inverter in induction motor drive using neural networks, Engineering Review, vol.42(3). https://doi.org/10.30765/er.1949.

[20] Tamissa Younes & Kadri Farid & Charif Fella & Benchabane Abderrazak \Neural Fault Diagnosis Method for Voltage Source Inverter with a Neural Direct Torque Control of Induction Motor"2020 IEEE The First IEEE International Conference on Communications, Control Systems and Signal Processing (CCSSP2020)16-17 March 2020 El-Oued, Algeria pp 480-486. DOI: 10.1109/CCSSP49278.2020.9151552

[21] A. Benchabane, Y. Tamissa, F. Charif and F. Kadri "AlexNet for Open-Switch Faults Detection in Induction Motor Inverter" May 2023 Fourth International Conference on Technological Advances in Electrical Engineering (ICTAEE'23.), May 23-24 2023; Skikda, Algeria

[22] Y. Tamissa, A. Benchabane, F. Charif and F. Kadri, "Diagnosis Multiple Open-Circuit Faults in Neural Direct Torque Control of Induction Motor Drive Using Neural Networks," 2024 International Conference on Advances in Electrical and Communication Technologies (ICAECOT), Setif, Algeria, 2024, pp. 1-6, doi: 10.1109/ICAECOT62402.2024.10829071.

[23] Y. Tamissa, F. Kadri, F. Charif and A. Benchabane, "Diagnosis and Pattern Recognition of Open Circuit Faults with Inverter Reconfiguration in Direct Torque Control of Induction Motor Drive Using Neural Networks," 2024 International Conference on Advances in Electrical and Communication Technologies (ICAECOT), Setif, Algeria, 2024, pp. 1-6, doi: 10.1109/ICAECOT62402.2024.10828848

[24] F. Kadri, D. Djarah, S. Drid, and F. Djeffal, ""Direct Torque Control of Induction Motor Fed by Three Phase PWM Inverter Using Fuzzy logic and Neural Network", ELECTROMOTION, Vol. 18, No 1", Romania,pp. 22-28, 2011.

[25] A.Hamza, K.Elgharbi «Commande Directe du Couple DTC-SVM d'une Machine Asynchrone (MAS)» diplôme de Master Université Mohamed Boudiaf - M'SILA, 2017.

[26] M.KHELALFA Rabah, M.BOULAHIA Youcef « Techniques de commande DTC-SVM appliquées à la machine asynchrone», Mémoire de Master, Université Ferhat Abbas –Sétif-1, 2019.

[27] H. ABU-RUB. D. Stando, M.P. Kazmierkowski «Simple speed sensorless DTC-SVM scheme for induction motor drives » Article, BULLETIN OF THE POLISH ACADEMY OF SCIENCES TECHNICAL SCIENCES, Vol. 61, No. 2, 2013.

[28] Tamissa, Younes & Charif, Fella & Abderrazak, Benchabane & Kadri, Farid. (2024). Fault-Tolerant Voltage Source Inverter for Induction Motor Drive Using Intelligent Techniques. 10.13140/RG.2.2.33314.64967.

[29] Farid Kadri* and Mohamed AssaadHamida, "Simple Threshold Method in Fault Diagnosis for Voltage Source Inverter in a Direct Torque Control Induction Motor Drive", Recent Patents on Engineering (2019) 13: 1. https://doi.org/10.2174/1872212113666191002121902.

[30] Z.Tir «Commande D'un Moteur Asynchrone Par Logique Floue », Mémoire de Master, Université d'EL-Oued, 2014.

[31] Touahar, Abdellatif & Abdelmalek, Zemit & Kadri, Farid & Tamissa, Younes. (2022). DTC-SVM Control of induction motor SVM Control of induction motor Fed by PWM inverter. 10.13140/RG.2.2.24583.06567.

[32] M W. Benkaddour « Diagnostic de défauts et reconfiguration d'onduleur Pour la commande directe de couple d'un moteur à induction Par la logique floue», Mémoire de Master, UKMO, 2017.

[33] Younes Tamissa, Farid Kadri" Neural Fault Diagnosis and Inverter Reconfiguration for a Neural Direct Torque 168 Control of Induction Motor Drive" Master Thesis University Kasdi Merbah Ouargla, Alger Aug. 2018, 169 http://doi.org/10.13140/RG.2.2.34815.97441.

[34] F. Kadri, S. Bensalem, and K. Houfar, « PI Speed Control for Fuzzy Direct Torque Control of induction motor using Fuzzy switching pattern », First International Conference on Electrical Engineering, CIGET'09. 25-26 Octobre/2009, Tebessa, Algeria.

[35] W. Boutana , N.Ykhelfoune «Etude comparative en simulation entre un régulateur PID et un régulateur flou » Mémoire de Master, Université Mohammed Seddik BENYAHIA Jijel, Juillet 2019.

[36] S.Haykin, (2009). Neural Networks and Learning Machines (3rd ed.). Pearson.

[37] I.Goodfellow, Y.Bengio, A. Courville, (2016). Deep Learning. MIT Press.

[38] Y.LeCun, Y.Bengio, G.Hinton, (2015). Deep learning. Nature, 521(7553), 436–444.

[39] M.Bear, F.B.W.Connors, M.A.Paradiso, (2016). Neuroscience: Exploring the Brain (4th ed.). Wolters Kluwer.

[40] E.R.Kandel, J.H.Schwartz, T.M.Jessell, S.A.Siegelbaum, A.J.Hudspeth, S.Mack, (2013). Principles of Neural Science (5th ed.). McGraw-Hill Education.

Abstract

This thesis presents a novel control strategy for induction motors by integrating Artificial Neural Networks (ANNs) into a Direct Torque Control with Space Vector Modulation (DTC-SVM) system. Traditional DTC methods, while providing fast dynamic response, suffer from high torque and flux ripples and variable switching frequencies. The inclusion of SVM improves performance but still requires precise tuning and suffers from model dependency. This work proposes the replacement of conventional PI controllers with ANN-based regulators trained on simulation datasets to optimize control performance. The proposed ANN-enhanced DTC-SVM is implemented and validated in MATLAB/Simulink. Simulation results confirm improved torque and flux regulation, reduced ripple, enhanced response time, and robustness against parameter variations, making it a suitable control strategy for industrial motor drive applications.

Keywords: Induction Motor, Direct Torque Control, Space Vector Modulation, Artificial Neural Networks, MATLAB/Simulink, PI Controller Replacement, Motor Drive, Intelligent Control.

الملخص

يعرض هذا البحث استراتيجية تحكم جديدة لمحركات الحث تعتمد على دمج الشبكات العصبية الاصطناعية في نظام التحكم المباشر في العزم مع تعديل المتجهات الفراغية (DTC-SVM). تعاني طرق التحكم التقليدية (DTC) من تموجات عالية في العزم والتدفق وترددات تبديل غير ثابتة، رغم أنها توفر استجابة ديناميكية سريعة. وقد حسّن استخدام تعديل المتجهات الفراغية الأداء، لكنه لا يزال يعتمد على ضبط دقيق ويعاني من التبعية للنماذج. يقترح هذا العمل استبدال المنظمات التقليدية (PI) بمنظمات تعتمد على الشبكات العصبية تم تدريبها باستخدام بيانات محاكاة. تم تنفيذ النظام المقترح باستخدام MATLAB/Simulink واستجابة وأظهرت النتائج تحسنًا في تنظيم العزم والتدفق، وانخفاض التموجات، وتحسين وقت الاستجابة، ومتانة في مواجهة التغيرات في المعلمات، مما يجعله مناسبًا لتطبيقات تشغيل المحركات الصناعية

الكلمات المفتاحية: محرك حثي، التحكم المباشر في العزم، تعديل المتجهات الفراغية، الشبكات العصبية الاصطناعية، ماتلاب/سيمولينك، استبدال المتحكم PI، التحكم الذكي، تشغيل المحركات.

Résumé

Ce mémoire propose une stratégie de commande innovante pour les moteurs asynchrones, basée sur l’intégration de réseaux de neurones artificiels (ANN) dans un système de commande directe du couple avec modulation vectorielle spatiale (DTC-SVM). Les méthodes DTC classiques offrent une réponse rapide, mais souffrent de fortes ondulations de couple et de flux ainsi que d'une fréquence de
commutation variable. L’ajout de la modulation vectorielle spatiale améliore les performances, mais dépend encore fortement de réglages précis et d’un modèle exact. Dans ce travail, les régulateurs classiques PI sont remplacés par des régulateurs intelligents basés sur des ANN, entraînés à partir de jeux de données issus de simulations. Le système proposé est implémenté et validé sous MATLAB/Simulink. Les résultats démontrent une amélioration notable de la régulation du couple et du flux, une réduction des ondulations, une meilleure réactivité et une robustesse accrue face aux variations de paramètres, le rendant idéal pour les applications industrielles.

Mots-clés: Moteur asynchrone, Commande directe du couple, Modulation vectorielle spatiale, Réseaux de neurones artificiels, MATLAB/Simulink, Remplacement du régulateur PI, Entraînement moteur, Commande intelligente.