Modeling a Permanent Magnet Synchronous Motor using Finite Element Method in FEniCS

March 2025

MA5990: Project Report

Under the guidance of

Dr. Vetrivel V
Dr. Phil Weir
Ms. Kanika Miglani

Submitted by

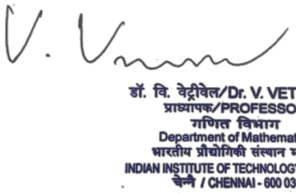
Abhinav T K
MA23M002

Department of Mathematics
Indian Institute of Technology Madras

THESIS CERTIFICATE

This is to certify that the thesis titled ***Modeling a Permanent Magnet Synchronous Motor using Finite Element Method in FEniCS*** submitted by **Abhinav T K** to the Indian Institute of Technology Madras for the **M.Tech Project Thesis**, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Research Guides



Chennai, India

Dr. Vetrivel V
(Professor, Department of Mathematics, IIT Madras)

Belfast, Northern Ireland

Dr. Phil Weir
(Director, Flax & Teal Ltd.)

Bangalore, India

Ms. Kanika Miglani
(Data Scientist, Flax & Teal Ltd.)

ACKNOWLEDGEMENTS

I thank **Dr. Vetrivel V**, Professor, Department of Mathematics, IIT Madras, for being my project guide and providing continuous support and guidance throughout this project. I would also like to thank the Department of Mathematics, IIT Madras, for providing the necessary facilities to carry out my work on campus.

I thank **Dr. Phil Weir**, Founder and Director of Flax & Teal Ltd., for his constant guidance, motivation, and encouragement throughout this project. I am grateful to him for giving me the opportunity to intern at Flax & Teal Ltd. I also appreciate his one-on-one mentorship and the belief he placed in my work. Additionally, I thank Flax & Teal Ltd. for providing financial support during the course of this project.

I thank **Ms. Kanika Miglani**, Data Scientist at Flax & Teal Ltd., for being my project co-guide and for managing our project efficiently. Her constant mentorship helped me achieve various milestones, and I appreciate the time she dedicated to guiding me through this project and clearing any roadblocks.

I thank **Dr. Jørgen S. Dokken**, Senior Research Engineer at Simula Research Laboratory, for his valuable time in addressing my technical queries while working on this project in FEniCS. I also extend my gratitude to the developers of the FEniCS project, the open-source computing platform used for my work. Jørgen, a collaborator of the FEniCS project, provided insightful guidance, and I especially thank him for sharing the TEAM 30 codes, which ensured a seamless start to this project for me.

I thank **Dr. Ellery Ames**, Scientific Software Engineer at Flax & Teal Ltd., for his technical guidance, support, and encouragement throughout this project. Additionally, I thank all my colleagues at Flax & Teal Ltd. for making this internship a great experience.

I would also like to thank **Partha** and **Snehal**, fellow interns, for supporting me throughout this project journey. Working together, sharing knowledge, and attending meetings together was a great experience.

Lastly, I am deeply thankful to my parents, family, and friends for their unwavering support, understanding, and love. A special thanks to my brother, Vyshnav, for his motivation and encouragement throughout the project.

Abhinav T K

ABSTRACT

Keywords: Finite Element Method, FEniCS, Open-source software, Permanent Magnet Synchronous Motor (PMSM), Maxwell's equations.

This project focuses on implementing a computational model for simulating the electromagnetic behavior of Permanent Magnet Synchronous Motors (PMSMs) using Finite Element Method (FEM) in FEniCS, a popular open-source computing platform for solving partial differential equations (PDEs). PMSMs are considered for this modeling project as they are the most common choice for electric heavy vehicles and jet engine applications, offering high efficiency, superior torque characteristics, and high power density compared to other motor types. The study emphasizes on developing efficient workflows for both 2D and 3D models. FEniCSx, the latest iteration of FEniCS, is used to define the variational form of Maxwell's equations and solve them. Insights from TEAM Problem 30 - Induction motor analyses was used as part of the literature review to understand FEM implementation steps in FEniCS. An efficient FEM-based model for PMSM is developed, showcasing the feasibility of using the open-source platform FEniCS for real-world electric machine simulations. The primary results of the 2D model include the magnetic vector potential and magnetic flux density, analyzed using ParaView. Animations were generated to visualize the rotating magnetic field. Additionally, torque was calculated to assess the industrial implications of the model. We compared the FEniCSx model results with existing model implementations results. Finally, we shifted to the three dimensional implementation of the model and learned about the key considerations and challenges involved in transitioning from 2D to 3D.

TABLE OF CONTENTS

LIST OF NOTATIONS

Symbol	Quantity	Unit
$\mathbf{B}$	magnetic flux density	T or Wb/m^2
$\mathbf{H}$	magnetic field intensity	A/m
$\mathbf{E}$	electric field intensity	V/m
$\mathbf{D}$	electric displacement field	C/m^2
$\mathbf{J}$	eddy current density	A/m^2
$\mathbf{J_0}$	source current density	A/m^2
μ	magnetic permeability	N/A^2
μ_0	permeability of free space ($4\pi \times 10^{-7}$)	N/A^2
μ_r	relative permeability (material-specific) ($\mu = \mu_0\mu_r$)	dimensionless
ν	reluctivity ($\nu = 1/\mu$)	A^2/N
ε	electric permittivity	F/m
σ	electrical conductivity	S/m
ρ	resistivity ($\rho = 1/\sigma$)	$\Omega \cdot m$
$\omega_\mathbf{r}$	angular velocity of the rotor	rad/s
$\mathbf{u}$	velocity of the rotor	m/s
$\mathbf{A}$	magnetic vector potential	Wb/m
V	electric scalar potential	V
Γ_B	external boundary	
Γ_{nc}	interface b/w conducting & non-conducting region	
$\mathbf{n}$	unit normal vector	
∇	del operator	
∇V	gradient of V	
$\nabla \cdot \mathbf{A}$	divergence of A	
$\nabla \times \mathbf{A}$	curl of A	

LIST OF FIGURES & TABLES

List of Figures

List of Tables

1 INTRODUCTION

Permanent Magnet Synchronous Motors (PMSMs) are among the top choices for electric jet engines and heavy-duty vehicles because of their high efficiency, power density, and superior torque characteristics compared to other motor types [1]. However, the permanent magnets in these motors contain rare earth elements, which are not economical, making motor optimization essential for developing electric vehicles. Numerical simulations, such as the finite element method (FEM), can be employed to design and analyze motor performance. FEM enables precise computation of electromagnetic fields while handling complex geometries, material non linearities, and spatial variations in flux density — critical factors for optimizing motor performance. Additionally, such numerical simulations facilitate parametric studies and design optimization by evaluating different rotor/stator configurations, magnet placements, and winding arrangements to enhance efficiency, torque density, and overall performance.

There are many commercial and open-source software tools available for finite element analysis. In this study, we use FEniCS [2, 3], a popular open-source computing platform that allows users to efficiently translate scientific models into finite element code. FEniCS is a well-maintained, actively developed platform with strong community support and comprehensive documentation.

1.1 Permanent Magnet Synchronous Motor (PMSM)

A Permanent Magnet Synchronous Motor (PMSM) is a type of AC motor that uses permanent magnets embedded in or attached to the rotor to create a constant magnetic field [4]. This makes it different from induction motors, which rely on induced currents to generate magnetism. PMSMs are widely used in applications that require high efficiency and torque density, such as electric vehicles, aviation, and industrial machinery. Their compact size and energy-efficient operation make them a preferred choice in scenarios where space and power are constrained. They offer high efficiency and high torque density within a small packaging volume. Advances in PMSM simulation directly contribute to the development of sustainable and energy-efficient technologies within heavy-duty electric vehicles and electric aircrafts. [1]

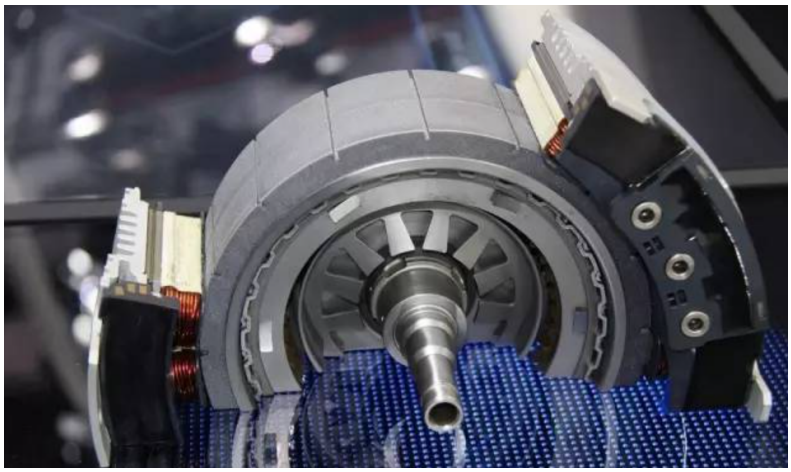


Figure 1: Rare Earth Permanent Magnet Motors

[1] Image source: https://www.stanfordmagnets.com/permanent-magnetic-synchronous-motors.html

Working of a PMSM:

A PMSM operates by combining the magnetic field created by permanent magnets on the rotor with the alternating magnetic field produced by three-phase stator coils. When the stator coils are excited with alternating current (AC) supply, they generate a rotating electromagnetic field around the stator. This rotating field interacts with the constant magnetic poles of the permanent magnets on the rotor, causing the rotor to synchronize and rotate at the same speed as the rotating stator field (hence the name "synchronous"). Permanent magnets provide a continuous source of flux, eliminating the need for external rotor excitation, while the AC supply of the stator ensures the rotating field that pulls or pushes the magnet poles around. [4]

1.2 Finite Element Method (FEM)

Finite Element Method is a numerical technique used to solve a variety of physical problems governed by PDEs [5]. It involves breaking down a complex problem domain into smaller, simpler parts called "finite elements" which are interconnected at the nodes. The most important advantage of FEM is that it is well suited for problems with physical domains having complex geometries.

Steps in FEM [5–7]:

1. **Discretization of the Domain:**
   Divide the domain into finite elements (triangular, quadrilateral, etc.), [8] connected at nodes. Finer meshes improve accuracy but increase computation.

2. **Weak Formulation:**
   Convert PDEs into an integral form, making them solvable using approximate solutions in function spaces such as Sobolev spaces.

3. **Approximation of the Solution:**
   Use basis (shape) functions to approximate the unknown solution within each element.

4. **Assembly of the System of Equations:**
   Construct the global system of equations by assembling element contributions into a stiffness matrix.

5. **Applying Boundary Conditions:**
   Apply physical constraints like fixed displacements or forces to ensure a meaningful solution.

6. **Solving the Linear System:**
   Solve the system using numerical methods like LU decomposition, with optimizations for large models.

7. **Post-Processing:**
   Visualize results (e.g., stress, temperature) and derive secondary quantities. Advanced visualization techniques, such as contour plots and vector fields, help interpret the solution and validate the results against experimental or analytical solutions.

1.3 FEniCS

FEniCS is a popular open-source computing platform for solving partial differential equations (PDEs) with the finite element method (FEM) [2, 3]. FEniCS enables users to quickly translate scientific models into efficient finite element code. FEniCS comes with high-level Python and C++ interfaces. More about FEniCS can be found in Appendix A. [2]

[2]The code repository for this project can be found in GitHub: https://github.com/abhinavtk7/pmsm-fenics

2 LITERATURE REVIEW

2.1 Modeling a permanent magnet synchronous motor in FEniCSx for parallel high-performance simulations

The main motivation of this project was based on the paper - modelling a permanent magnet synchronous motor in FEniCSx for parallel high-performance simulations [14] which addresses the challenge of modeling PMSM for high-performance applications, particularly in electric vehicles and aviation. The authors developed a three-dimensional PMSM model using FEniCSx optimized for parallel computing. Their primary goal was to enhance the scalability of PMSM simulations to handle extreme-scale problems efficiently. The paper extensively evaluates the parallel performance of the model on a high-performance computing (HPC) cluster. The study concludes that using iterative solvers and parallel computing significantly accelerates PMSM simulations, making them feasible for large-scale multiphysics analysis. The two-dimensional model we develop is based on the numerical formulation of this model.

2.2 TEAM Problem No. 30 a- Induction Motor Analyses

The literature review began by studying the implementation of TEAM 30 - Induction Motor Analyses [15] problem which simulates an induction motor in 2D and calculates the magnetic flux density (**B**) using the A-V formulation.

Problem No. 30 a - Induction Motor Analyses [15] describes an induction motor problem in which the eddy currents in the rotor are induced both by time harmonic currents on the stator and by the rotation of the rotor. It is a problem in two-dimension where angular velocities of rotor and stator, source current density, conductivity, relative permeability of each material are given. In the problem Maxwell's equations are solved using A-V formulation to find the magnetic vector potential (A) using the Finite Element Method. Quantities such as Magnetic flux density (B), Electromagnetic torque and Induced voltage in the phase coil A are calculated by postprocessing. [3]

2.3 Results of TEAM 30 FEniCS code

2.3.1 Induction Motor Mesh

The analysis was done on a three-phase AC induction motor 2D mesh. Meshes were generated using GMSH [18]. [4]

[3] A FeniCS implementation of the problem can be found at Github: Wells-Group/TEAM30 [17], which is the starting point for the 2D implementation of PMSM model.

[4] GMSH is an open-source mesh generator widely used for creating geometries and generating meshes for FEM in both 2D and 3D.

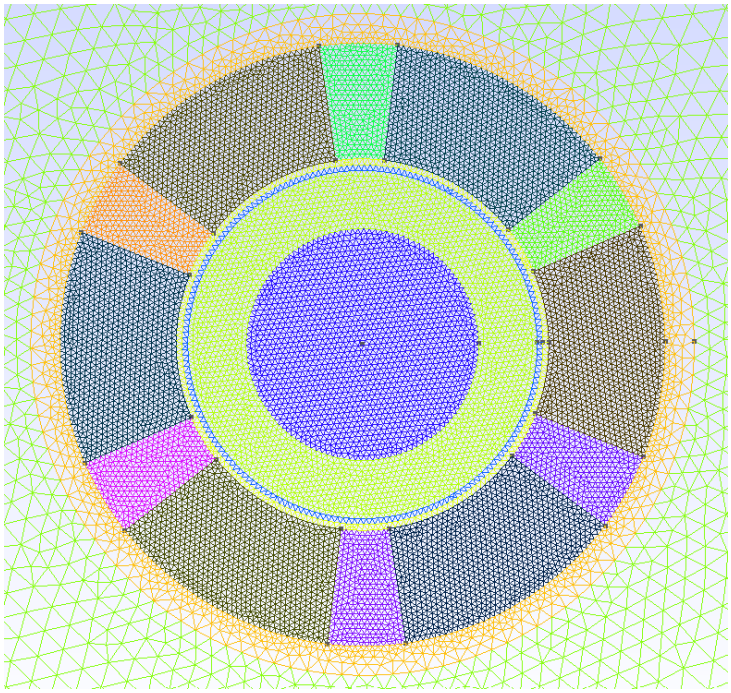
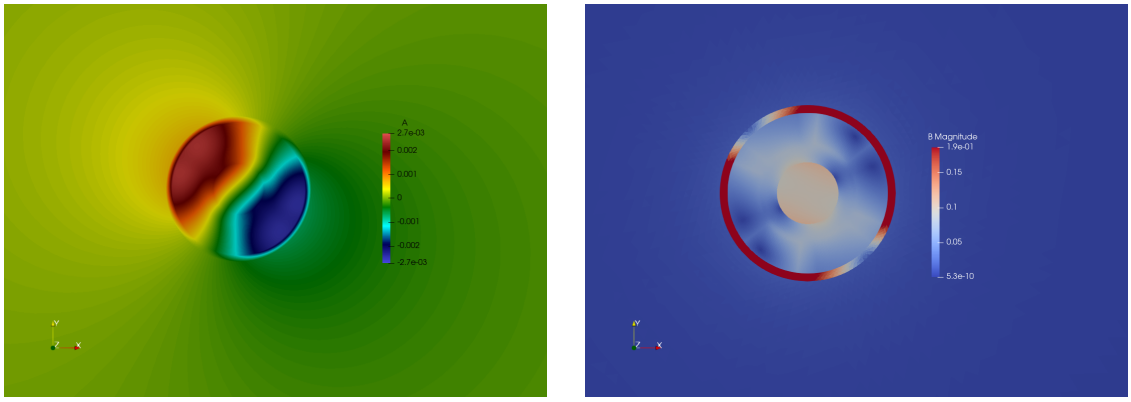


Figure 2: Induction Motor: 2D mesh

2.3.2 Induction Motor Outputs

Figure 3 showing magnetic vector potential (A) and magnetic flux density (B) are generated during the simulation. They are stored in .bp files and visualized in Paraview [19].

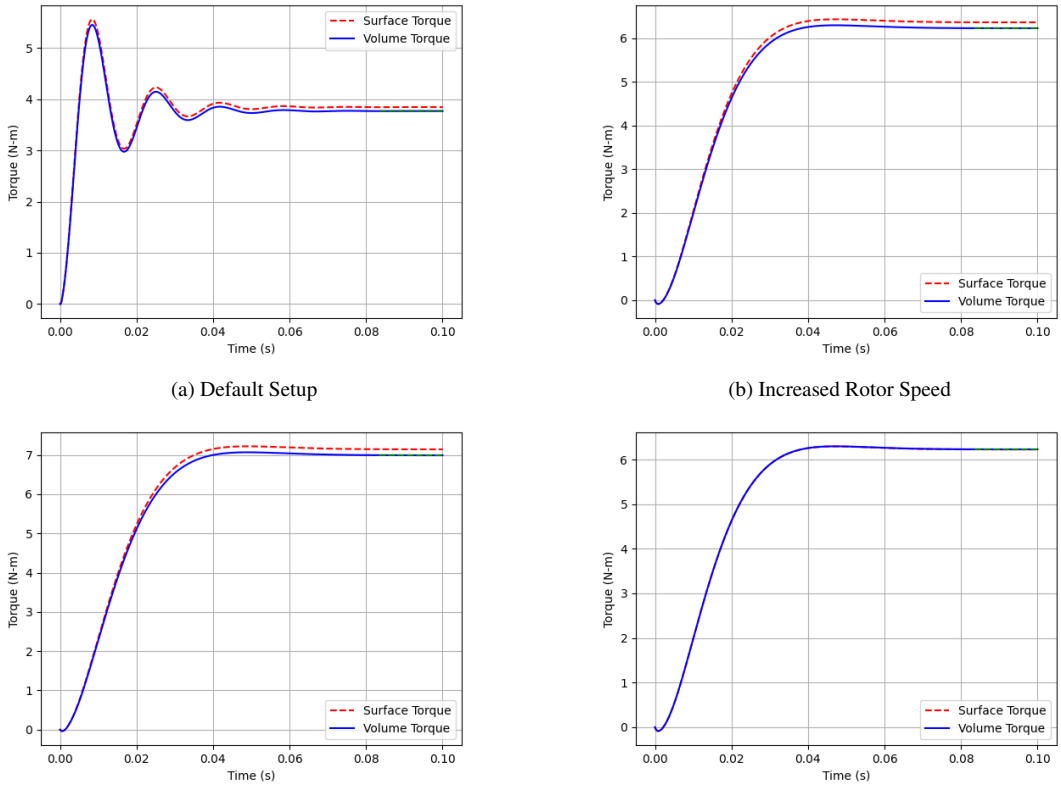


(a) Magnetic Vector Potential (A)

(b) Magnetic Flux Density (B)

Figure 3: Induction Motor: Results

Electromagnetic torque is the main quantity of interest. Torque calculation was done in two ways: Average surface torque calculated using Maxwell's stress tensor [36] and average volume torque using Arkkio's method [37].
A parametric study was done to understand the effect of model parameters such as angular rotor speed, polynomial degree for the magnetic vector potential, number of time steps per phase and number of phases to run on the torque calculation. The induction motor is excited with the AC three-phase currents, after which we get the steady-state torque values when we vary different parameters. Torque v/s time plots are shown below when the AC currents are turned on.



(a) Default Setup

(b) Increased Rotor Speed

(c) Increased Time Resolution

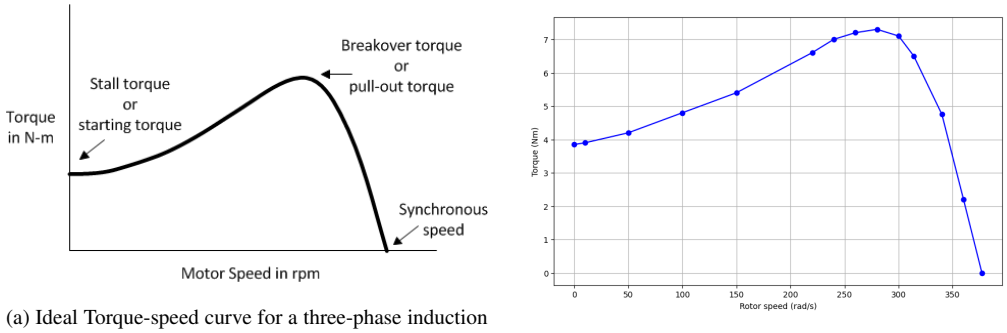
(d) Higher Polynomial Degree

Figure 4: Induction Motor: Torque calculations for three-phase induction motor

Figure	Angular Speed (rad/s)	Degree of polynomial	Number of time steps
(a) Default Setup	0	1	100
(b) Increased Rotor Speed	314.16	1	100
(c) Increased Time Resolution	314.16	1	300
(d) Higher Polynomial Degree	314.16	2	100

Table 1: Induction Motor: Parametric study for Torque Calculation

The torque-speed characteristics of a 3-phase induction motor [20] is defined as the curve plotted between torque developed and rotational speed of the motor. The comparison of the ideal torque-speed curve with the TEAM 30 model's results provides initial validation of the model's accuracy and suggests numerical stability.



(a) Ideal Torque-speed curve for a three-phase induction motor

(b) Torque-speed curve: Result of TEAM 30 code

Figure 5: Induction Motor: Torque-speed characteristics curve

3 METHODOLOGY

3.1 Numerical formulation

3.1.1 Maxwell's equations

We start the formulation with the Maxwell's equations which are the fundamental laws governing electromagnetism. They describe how electric and magnetic fields interact with charges and currents. In differential form, the set of Maxwell's equations can be written as follows:

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \tag{1a}$$

$$\nabla \times \mathbf{H} = \frac{\partial \mathbf{D}}{\partial t} + \mathbf{J} \tag{1b}$$

$$\nabla \cdot \mathbf{B} = 0 \tag{1c}$$

$$\nabla \cdot \mathbf{D} = \rho \tag{1d}$$

where $\mathbf{E}$ is the electric field intensity, $\mathbf{H}$ is the magnetic field intensity, $\mathbf{B}$ is the magnetic flux density, $\mathbf{J}$ is the current density, and $\mathbf{D}$ is the electric displacement field.
Equation 1a is known as Faraday's Law, or law of magnetic induction, and describes the effect of a changing magnetic field on the electric field. Equation 1b is the Ampere's circuital law generalized by Maxwell, which details how electrical currents can generate a magnetic and changing electric fields (displacement currents). Equation 1c states that the magnetic field is divergence-free and Equation 1d describes the relationship between the electric field and the electric charges that cause it.
Eddy Current Regime: The Eddy Current model is obtained when the displacement current effects can be neglected [29]. Hence the problem is simplified by omitting Equation 1d and asserting $\mathbf{D} = 0$ in Equation 1b to get $\nabla \times \mathbf{H} = \mathbf{J}$.

3.1.2 Constitutive relations

The constutive relations are as follows,

$$\mathbf{B} = \mu \mathbf{H} \tag{2a}$$

$$\mathbf{J} = \sigma \mathbf{E} \tag{2b}$$

$$\mathbf{D} = \varepsilon \mathbf{E} \tag{2c}$$

Here μ is the permeability, σ is the conductivity and ε is the permittivity of the material. Moreover $\nu = \frac{1}{\mu}$ and $\rho = \frac{1}{\sigma}$ where ν is the reluctivity and ρ is the resistivity of the material.

3.1.3 A-V formulation

In the A-V formulation [21, 22, 24, 25] we introduce the magnetic vector potential (A) through the following definition:

$$\mathbf{B} = \nabla \times \mathbf{A} \tag{3}$$

To ensure the uniqueness of the magnetic vector potential, the divergence of it can be selected according to Coulomb gauge, i.e. $\nabla \cdot \mathbf{A} = 0$ [28] It is substituted into Faraday's law (1a) to

define the electric scalar potential (V)

$$\mathbf{E} = -\frac{\partial \mathbf{A}}{\partial t} - \nabla V \tag{4}$$

In **2D problems**, Coulomb gauge $\nabla \cdot \mathbf{A} = 0$ is satisfied automatically. If the source current density has only z component, the magnetic field intensity vector (**H**) and the magnetic flux density vector (**B**) have x and y components, i.e.

$$\vec{J_0} = J_0(x,y)\vec{e}_z,$$

$$\vec{H} = H_x(x,y)\vec{e}_x + H_y(x,y)\vec{e}_y,$$

$$\vec{B} = B_x(x,y)\vec{e}_x + B_y(x,y)\vec{e}_y.$$

The magnetic vector potential has only the z component,

$$\vec{A} = A_z(x,y)\vec{e}_z,$$

because $(A_x = 0, A_y = 0 \text{ and } A_z = A_z(x,y))$

$$\vec{B} = \nabla \times \vec{A} = \begin{vmatrix} \vec{e}_x & \vec{e}_y & \vec{e}_z \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & A_z \end{vmatrix} = \vec{e}_x \frac{\partial A_z}{\partial y} - \vec{e}_y \frac{\partial A_z}{\partial x},$$

i.e. $B_x(x,y) = \frac{\partial A_z}{\partial y}$ and $B_y(x,y) = -\frac{\partial A_z}{\partial x}$. The divergence of this one component vector potential is equal to zero, because

$$\nabla \cdot \vec{A} = \frac{\partial A_z(x,y)}{\partial z} = 0. \tag{5}$$

3.1.4 Modeling permanent magnets

The main change in PMSM when compared to induction motors is the introduction of permanent magnets embedded or surface-mounted on the rotor which allows higher torque and power density since the rotor's magnetic field is established by permanent magnets. The permanent magnets (PMs) are modelled as Neodymium iron boron (NdFeB), a type of rare-earth permanent magnet. A linear model is approximated for the PMs where the magnets act like a source of flux [26]. Each magnet wedge is assigned a fixed direction and a constant magnitude. PM subdomains (10 magnet wedges) are marked by domain markers and we assign the Magnetization(M) values to them.
Magnetic flux density is given by:

$$\mathbf{B} = \mu \mathbf{H} + \mathbf{B_r}$$

$\mathbf{B_r}$ is the remanent magnetic flux density of the ferromagnet.
The magnetization **M** is calculated by:

$$\mathbf{M} = \nu_0 \mathbf{B_r}$$

Hence the behavior of the permanent magnets is described by the equation:

$$\mathbf{B} = \mu \mathbf{H} + \mu_0 \mathbf{M} \tag{6}$$

3.1.5 Modeling rotation

The rotation of the motor is modelled by using the motion voltage term $\mathbf{u} \times \mathbf{B}$ [21, 27]

$$\mathbf{E}' = \mathbf{E} + \mathbf{u} \times \mathbf{B}$$
$$\mathbf{J} = \sigma \left(-\frac{\partial \mathbf{A}}{\partial t} - \nabla V + \mathbf{u} \times \mathbf{B} \right)$$

$$\mathbf{J} = \sigma \left(-\frac{\partial \mathbf{A}}{\partial t} - \nabla V + \omega_r r \times \mathbf{B} \right) \quad \text{in } \Omega_r \cup \Omega_{pm} \tag{7}$$

3.1.6 Governing equations

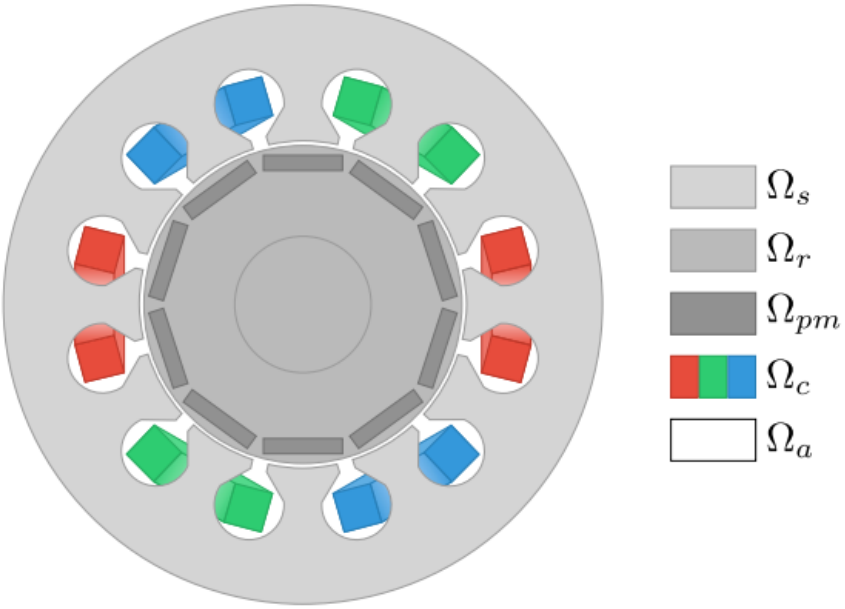


Figure 6: Cross-section of the modeled pmsm motor

Figure 6 shows the cross-section of the modeled pmsm motor. Ω_s denotes the stator, Ω_r the rotor, Ω_{pm} the permanent magnets, Ω_c the coil windings, and Ω_a the surrounding air. The coil windings are colored to signify each phase; red for phase A, green for phase B, and blue for phase C. [14]

The Coulomb gauge is defined as $\nabla \cdot A = 0$
For a unique solution to the magnetic vector potential, the Coulomb gauge must be satisfied.

Strong form:
The solution of the magnetic vector potential will be calculated using

$$\nu \nabla^2 \mathbf{A} = \sigma \frac{\partial \mathbf{A}}{\partial t} + \nabla V - \sigma \omega_v r \times (\nabla \times \mathbf{A}) \quad \text{in } \Omega_r \tag{8a}$$

$$\nu \nabla^2 \mathbf{A} = \sigma \frac{\partial \mathbf{A}}{\partial t} + \nabla V \quad \text{in } \Omega_s \tag{8b}$$

$$\nu\nabla^2\mathbf{A} = -\mathbf{J}_s \quad \text{in } \Omega_c \tag{8c}$$

$$\nu\nabla^2\mathbf{A} = -\nabla\times(\nu\mu_0\mathbf{M}) - \sigma\omega_v r\times(\nabla\times\mathbf{A}) \quad \text{in } \Omega_{pm} \tag{8d}$$

The electric scalar potential will be calculated using

$$\sigma\nabla^2 V = \nabla\cdot\sigma\left(-\frac{\partial\mathbf{A}}{\partial t} + \omega_v r\times(\nabla\times\mathbf{A})\right) \quad \text{in } \Omega_r \tag{9a}$$

$$\sigma\nabla^2 V = \nabla\cdot\left(-\sigma\frac{\partial\mathbf{A}}{\partial t}\right) \quad \text{in } \Omega_s \tag{9b}$$

$$\sigma\nabla^2 V = \nabla\cdot(\sigma\omega_v r\times(\nabla\times\mathbf{A})) \quad \text{in } \Omega_{pm} \tag{9c}$$

Homogeneous Dirichlet boundary condition

$$\mathbf{A} = 0 \quad \text{in } \partial\Omega \tag{10}$$

is imposed on the outer boundary by assuming that the outer boundary is located far from the motor, where the magnetic fields are negligible.

3.1.7 Temporal discretization

Backward Euler method was employed and the time derivative in the strong formulation is discretized as below:

$$\frac{\partial\mathbf{A}}{\partial t} = \frac{A_{n+1}-A_n}{\Delta t_n} \tag{11}$$

where Δt_n is the time-step. $t_{n+1} = t_n + \Delta t_n$

3.2 Weak formulation

Transition from the strong form to the weak formulation involves integrating the governing strong form equations against a set of test functions and applying integration by parts to reduce the order of derivatives. The unknown variables to be solved are represented by a trial function.

The weak formulation of the problem reads: given A_n at time t_n, and $J_{s,n+1}$ and M_{n+1} at time t_{n+1}, find $A_{n+1}\in[Q_h]^3$ and $V_{n+1}\in Q_h$ such that

$$\begin{aligned} F_A(A_{n+1};\mathbf{v}) := &\int_\Omega \nu\nabla A_{n+1}\cdot\nabla\mathbf{v}\,dx + \int_{\Omega_{r,s}}\sigma\frac{A_{n+1}-A_n}{\Delta t}\cdot\mathbf{v}\,dx \\ &+ \int_{\Omega_{r,s}}\sigma V_{n+1}\cdot\mathbf{v}\,dx + \int_{\Omega_{r,pm}}\sigma\omega_v r\times(\nabla\times A_{n+1})\cdot\mathbf{v}\,dx \\ &- \int_\Omega J_{s,n+1}\cdot\mathbf{v}\,dx - \int_{\Omega_{pm}}\nu\mu_0 M_{n+1}\cdot(\nabla\times\mathbf{v})\,dx = 0, \quad \forall\mathbf{v}\in[Q_h]^3 \end{aligned} \tag{12}$$

$$F_V(V_{n+1};q) := \int_\Omega \sigma \nabla V_{n+1} \cdot \nabla q \, dx - \int_{\Omega_{r,s}} \sigma \frac{A_{n+1} - A_n}{\Delta t} \cdot \nabla q \, dx$$
$$+ \int_{\Omega_{r,pm}} \sigma \omega_v r \times (\nabla \times A_{n+1}) \cdot \nabla q \, dx = 0, \quad \forall q \in Q_h \tag{13}$$

where Q_h is a finite element space and $Q_h \subset H^1(\Omega)$.

3.2.1 Weak formulation in 2D

In 2D implementation, our input current $\mathbf{J_0} = (0,0,J_0)$, we get $\mathbf{A} = (0,0,A_z)$ and we can solve a simple scalar equation for the z component of the magnetic potential.
We can write

$$\nabla \times A = \frac{\partial A_z}{\partial y}\hat{i} - \frac{\partial A_z}{\partial x}\hat{j}$$

We also assume that $\frac{\partial V}{\partial z} = 0$.
Hence, we can solve a simple scalar equation for the z component of the magnetic potential.

Coupled system after simplifying to 2D:

$$\begin{aligned}
&\int_{\Omega_c \cup \Omega_n} \mu^{-1} \nabla A_z \cdot \nabla v_z \, \mathrm{d}x + \int_{\Omega_c} \sigma \frac{\partial A_z}{\partial t} v_z \, \mathrm{d}x \\
&+ \int_{\Omega_c} \sigma \frac{\partial V}{\partial z} v_x \, \mathrm{d}x + \int_{\Omega_c} \sigma \left(\frac{\partial V}{\partial x}\frac{\partial q}{\partial x} + \frac{\partial V}{\partial y}\frac{\partial q}{\partial y} \right) \mathrm{d}x \\
&+ \int_{\Omega_c} \sigma (\mathbf{u} \cdot \nabla A_z) v_z \, \mathrm{d}x = \int_{\Omega_n} J_{0,z} v_z \, \mathrm{d}x + \int_{\Omega_{pm}} \mu^{-1}\mu_0 \left(M_x \frac{\partial v_z}{\partial y} - M_y \frac{\partial v_z}{\partial x} \right) \mathrm{d}x
\end{aligned} \tag{14}$$

2D Weak Formulation: Comparison with the Induction Motor problem

The key differences between the weak formulations of the induction motor problem we studied and the PMSM problem we implemented are as follows:

1. Permanent Magnet (PM) Region Ω_{pm} is defined.
2. Added PM region to the conducting region. $\Omega^{\text{new}} = \Omega^{\text{old}} \cup \Omega_{pm}$
3. Magnetization source term: $\int_{\Omega_{pm}} \mu^{-1}\mu_0 \left(M_x \frac{\partial v_z}{\partial y} - M_y \frac{\partial v_z}{\partial x} \right) \mathrm{d}x$
4. Modeled rotation for PM.
5. Remaining regions are kept the same and the same boundary conditions are applied.

3.3 Finite Element Method

The Finite Element Method (FEM) is the most popular and the most flexible numerical technique to determine the approximate solution of the partial differential equations in engineering [21, 22, 25]. The main steps of simulation with FEM is shown in Figure 7 [21]

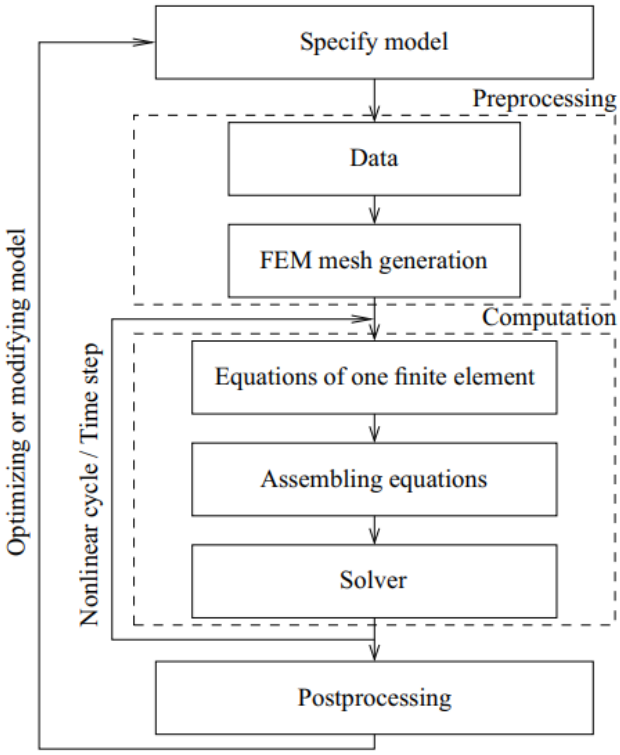


Figure 7: Steps of simulation by FEM

3.3.1 Shape functions

A shape function in FEM is an interpolation function that defines how a field variable varies within an element. The potential functions in the formulation, V is scalar valued that can be approximated by the nodal shape function. **A** is vector potential function which can be approximated by either nodal shape functions or vector shape functions, also called edge shape functions [21, 22, 25]. See Appendix B to know more about shape functions.

3.4 PMSM Mesh Generation

We start our implementation by generation of three-phase PMSM mesh in 2D. The mesh is generated using GMSH [18][5].

The following considerations were given while creating the mesh:

- **Unstructured Triangular Mesh:** The mesh consists of triangular elements, which are well-suited for FEM analysis.
- **Three-phase AC motor** with 6 copper coils and 10 permanent magnets. Rectangular magnets are approximated as segments of a circular ring.

[5]Mesh is created in msh format and converted to XDMF format for use with FEniCS.

- **Sub domains:** Rotor steel, Rotor Aluminium, Permanent magnets, Air Gap, Copper coils, Stator, Air. Each subdomain will be assigned different material properties (permeability, conductivity, etc.). Boundary facets (outer boundary, air-gap boundary) are marked for applying boundary conditions.
- **Coarser Mesh in Outer Regions:** The external space has a coarser mesh to reduce computational cost while still maintaining boundary constraints.

The parameters, mesh resolution (default:res = 1 mm) and size of surround box with air (default: L = 1 m) can be adjusted.[6]

Distance of Boundary from Origin	
Domain	**Radius (mm)**
Rotor steel	17
Permanent Magnets	36 - 38
Rotor Aluminium	40
Air Gap	42
Copper coils	62
Stator	75

No. of cells	41402
No. of DOFs	41564

Table 2: PMSM 2D Mesh: Default setup

3.5 Model Parameters

Parameter	Value	Unit
Rotational speed	600	RPM
Supply current frequency	50	Hz
Time-step	1	ms
Final time	100	ms

Table 3: Simulation parameters

From the equations defined in the weak formulation (14), we have

$$\Omega_c = \Omega_{\text{rotor}} \cup \Omega_{\text{pm}} \tag{15}$$

$$\Omega_n = \Omega_{\text{air}} \cup \Omega_{\text{copper}} \cup \Omega_{\text{stator}} \tag{16}$$

Material properties for each domains:

[6]Implementation of the 2D mesh: click here

	Ω_{rotor}	Ω_{pm}	Ω_{air}	Ω_{copper}	Ω_{stator}
μ_r	100	1.04457	1	0.999991	100
$\sigma[\Omega/m]$	$2 \cdot 10^6$	$6.25 \cdot 10^5$	-	-	0
$\rho[kg/m^3]$	7850	7500	-	-	7850

Table 4: Material properties of PMSM model

Permeability of free space, μ_0 [H/m] $= 1.257$
In each copper winding we have a current density $J = 1.41 \times 10^6 A/m^2$.
We can write $\mathbf{J_0}(t) = (0, 0, \alpha J \cos(\omega t + \beta))$ where the segment with center at angle θ has the following parameters

θ	α	β
0	1	0
$\frac{\pi}{3}$	-1	$\frac{2\pi}{3}$
$\frac{2\pi}{3}$	1	$\frac{4\pi}{3}$
π	-1	0
$\frac{4\pi}{3}$	1	$\frac{2\pi}{3}$
$\frac{5\pi}{3}$	-1	$\frac{4\pi}{3}$

Table 5: Three phase: Parameters for current excitation

Magnetization part: [7]

- 10 equally spaced magnets which are approximated as segments of a circular ring with an arc angle of 30° and a gap of 6° between magnets.
- Each magnet wedge has a constant magnitude (8.38×10^5 A/m (coercivity)) and an orientation given by angle which is the polar coordinate for the magnet's direction in the (x, y) plane.
- Magnet polarity is toggled for every other wedge (using `inout` variable), ensuring N–S–N–S arrangement.

3.6 FEniCS implementation for 2D model

3.6.1 Installation and setup

Implementation of the simulation is done using FEniCSx [2] the latest iteration of FEniCS. It provides support for a wide range of cell types and elements, memory parallelisation, and complex number support. 2D implementation is based on the TEAM30 project [15, 17] and is configured to run inside a Docker container with dolfinx.
Installation guide of this project can be found in the Appendix C.

3.6.2 Code overview

Majority of FEM workflow such as reading the mesh, defining trial and test functions, assembling the system and saving the solution is handled by **dolfinx** library. Implementation of weak formulation is done using **UFL**. The linear system is solved using **PETSc**, which ensures

[7] Magnetization code: click here

scalability and efficient numerical computation. [8]
Code snippet of trial and test function expression and weak formulation definition using ufl can be found in Appendix D.

3.6.3 Solver configuration

PETSc KSP solver is configured to solve the finite element system in the PMSM model. The solver is assigned the system matrix **A** and given a unique prefix to avoid conflicts.
Although a detailed solver tuning is not done in this project, the importance of solver settings is explored. Using ksp_type = preonly with pc_type = lu ensures direct solving, which is robust and accurate, but not scalable for large problems due to memory and computational cost. Exploring iterative solvers (like cg, gmres) with appropriate preconditioners (e.g., ilu, hypre, gamg) could significantly improve performance for large-scale or 3D problems.

3.7 Torque calculation

Torque is calculated in two ways:

1. **Surface torque (Maxwell's stress tensor)** [36]: This approach integrates the Maxwell stress over a boundary surface in the air gap, effectively capturing the tangential force that the electromagnetic field exerts on the rotor. The resulting torque is then the integral of $\mathbf{r} \times d\mathbf{F}$ around that boundary.

2. **Volume torque (Arkkio's method)**: Proposed by Arkkio [37], this method computes torque by integrating the radial and tangential field components across surface in the air gap. It is often cited as being more robust against small local variations in the mesh or boundary field, making it numerically stable.

You can find the derivation of the torque calculation equations in Appendix E.

3.8 2D to 3D Implementation

Moving from 2D to 3D in FEM for PMSM modeling brings significantly more complexity, but the core physics remains the same. Key considerations we followed for the implementation are stated below.

Mesh Generation: [9]

- Tetrahedral elements are the default choice for 3D FEM.
- Gmsh is used to generate the mesh and imported it into FEniCS. We transitioned the motor model from 2D to 3D by extruding the existing 2D geometry in Gmsh.
- Generating a quality mesh is more difficult due to the need for well-shaped tetrahedron to avoid numerical instabilities.
- Adaptive meshing may be necessary for better accuracy in regions with high field variations.

[8] GitHub link of 2D code: https://github.com/abhinavtk7/pmsm-fenics/blob/main/src/pmsm.py
[9] 3D Mesh generation code: https://github.com/abhinavtk7/pmsm-fenics/blob/main/src/generate_pmsm_3D.py

Weak Formulation:

- Magnetic vector potential (A): Now a 3D vector field, leading to more degrees of freedom (DOFs).
- The formulation involves curl-curl operators and divergence-free constraints.
- Divergence-free constraints can be enforced by applying the Coulomb gauge condition, $\nabla \cdot A = 0$, when using nodal elements. Using mixed finite elements (e.g., Nédélec elements) ensures compatibility between **A** and **B** thereby preserving the divergence-free condition.
- The variational formulation remains similar but involves 3D integrations, which increases computational cost.

FEniCS Implementation [10]

- Use VectorFunctionSpace() to define the magnetic vector potential **A**.
- Mixed function spaces, like (Ax, Ay, Az, V) can be used to satisfy additional constraints.
- Computational cost is significantly higher; parallelization may be necessary for efficiency.

Boundary Conditions

- Homogeneous boundary condition A = 0 applied on external boundaries such as top/bottom surfaces and the outer cylindrical boundary.

[10] GitHub Link for 3D code: https://github.com/abhinavtk7/pmsm-fenics/blob/main/src/pmsm_3D.py

4 RESULTS & DISCUSSION

4.1 2D Results

4.1.1 PMSM 2D Mesh

- Figure 8 shows the PMSM mesh in 2D, generated using Gmsh [18]. We can see concentric circular arcs for the rotor, stator and air gap, plus different wedge-shaped regions corresponding to magnets and coils.
- This finite element mesh containing triangular elements represents a three-phase AC motor.
- Each region has a different color to indicate a separate subdomain, to which we will assign unique material properties (conductivity, permeability, density, etc.).
- It has 6 copper coils and 10 permanent magnets. Rectangular magnets are approximated as segments of a circular ring.
- Boundary facets (outer boundary) are marked for applying boundary conditions.
- An air-box surrounds the motor region.
- **Sub domains:** Air, Stator, Rotor steel, Rotor Aluminium, Copper coils (6x), Permanent magnets (10x), Air Gap.

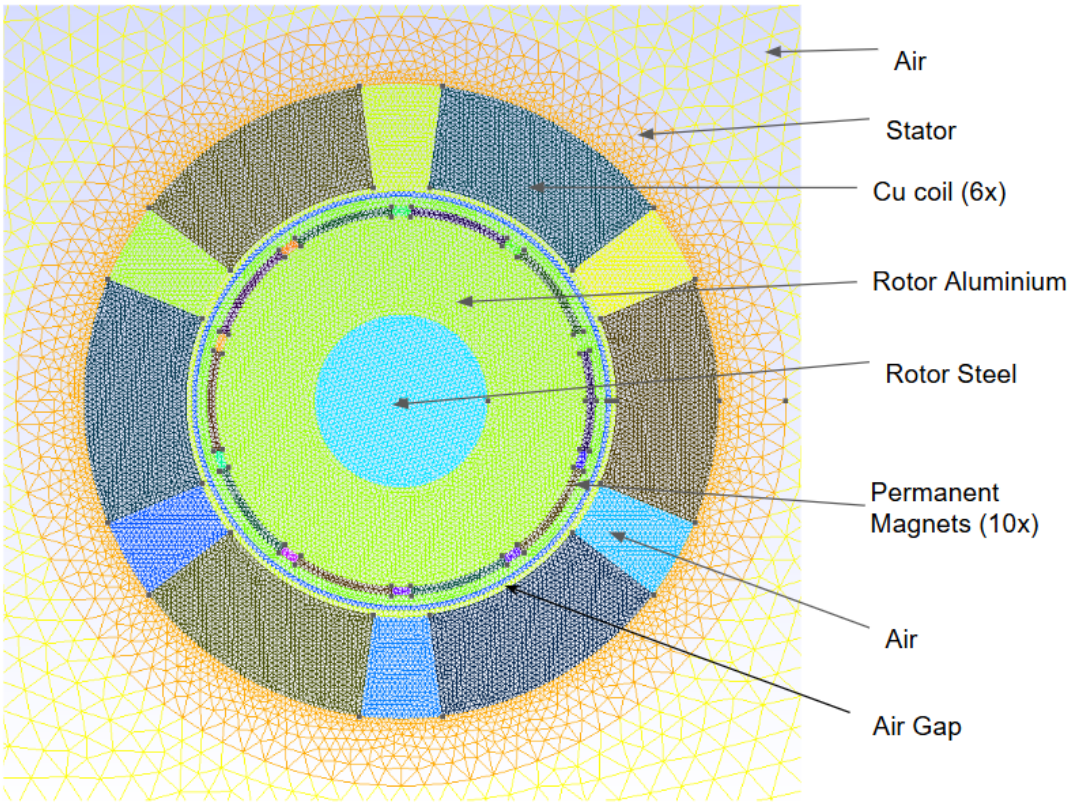


Figure 8: PMSM 2D Mesh with sub-domains

4.1.2 Magnetic vector potential (A) and Magnetic flux density (B)

The primary results of the 2D model are magnetic vector potential (A) and magnetic flux density (B). Figure 9 shows the magnetic vector potential (A) distribution at the 100th time step for the default model configuration, visualized in ParaView [19]. The solution is stored in .bp files, a format that efficiently handles large simulation data. ParaView can directly read and animate .bp outputs, enabling interactive exploration of the rotating field.

Figure 10 shows the magnetic flux density (B) magnitude at the 100th time step. As expected, the strongest B values appear near the poles of the permanent magnets, where the magnetization is highest. The color scale indicates flux density from near zero (blue) up to around 6.7 T (red).

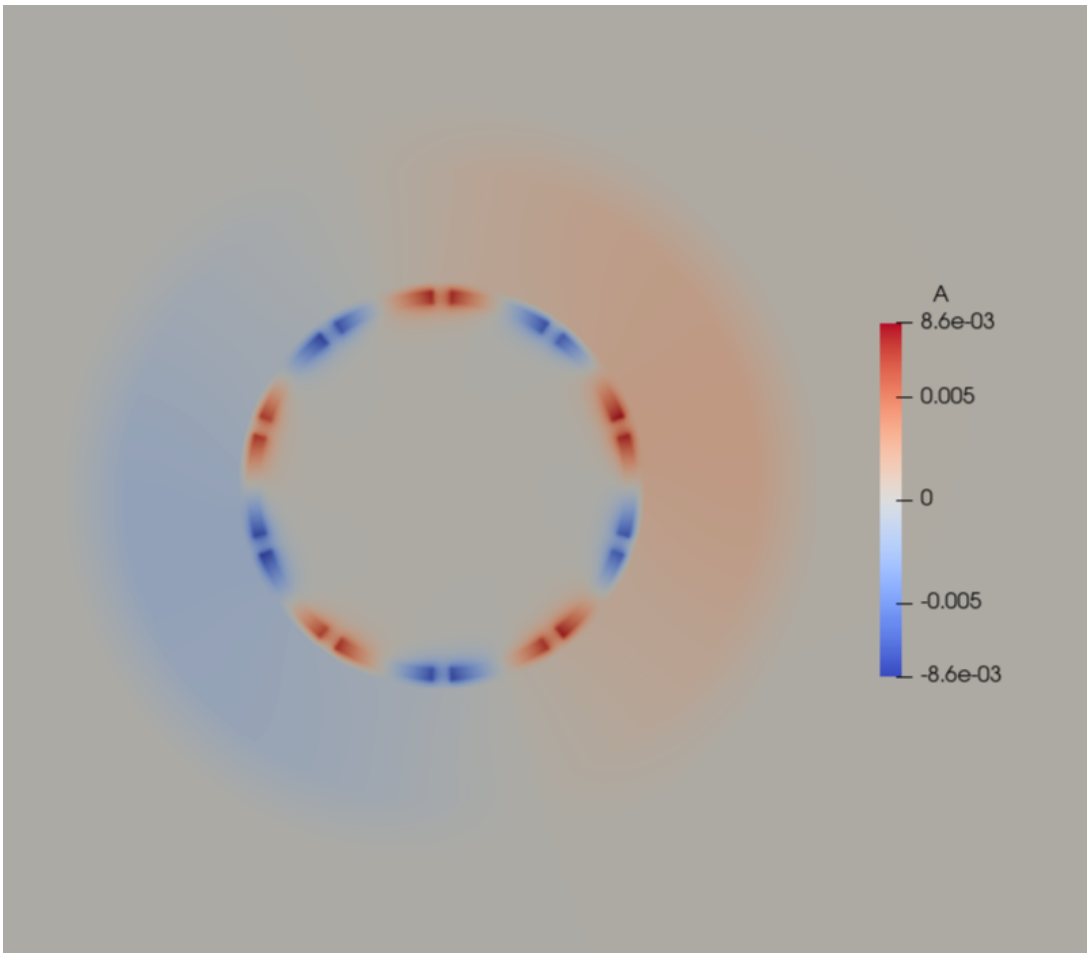


Figure 9: PMSM 2D Results: Magnetic Vector Potential (A) in Wb/m

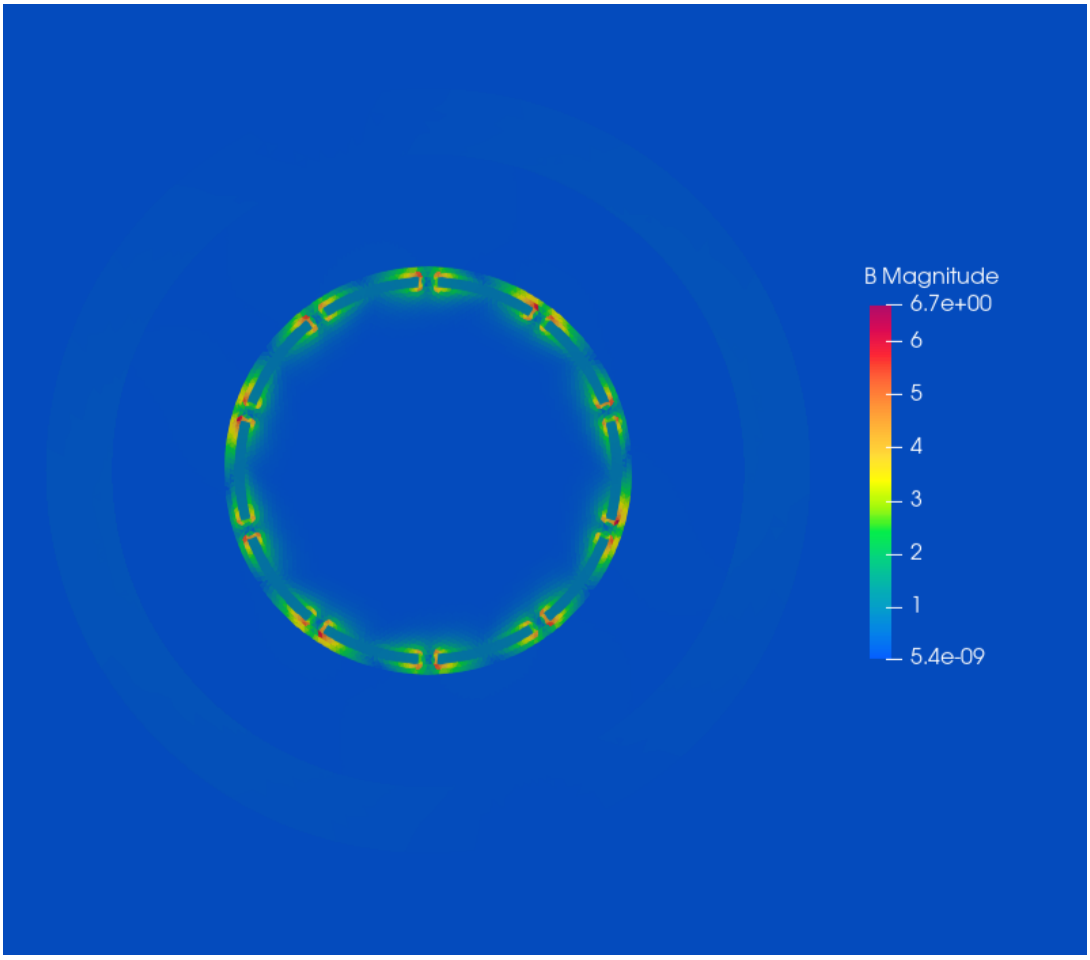


Figure 10: PMSM 2D Results: Magnetic Flux Density (B) in T

To better visualize the rotating field over 100 time steps, short animation clips were generated. [11] These animations show how both stator and rotor fields evolve and rotate at a uniform rate.

Key Observations:

- **Rotating Magnetic Field:** The results confirm a rotating field typical of a synchronous motor operation. Each magnet pole contributes to a sinusoidal (or near-sinusoidal) variation of B around the air gap.
- **Uniform Rotation:** Rotor and stator's magnetic fields rotate together at the same speed. The rotor's permanent magnets align with the stator's rotating field, ensuring synchronous operation.
- **High B at Magnet Poles:** The magnetic flux density (B) is strongest at the surface of the permanent magnet poles. The peak value of the flux density here is 6.7 Tesla.
- **Low Field:** Inner rotor region and the outer air domain beyond the stator show negligible field.

[11] **Watch the animation:**
Magnetic vector potential: click here
Magnetic flux density: click here

- **Time Evolution:** Over the 100 time steps, the color maps and animations indicate a smooth progression of the field rotation, demonstrating no numerical instabilities or abrupt changes—pointing to a well-conditioned finite-element formulation and stable time integration.

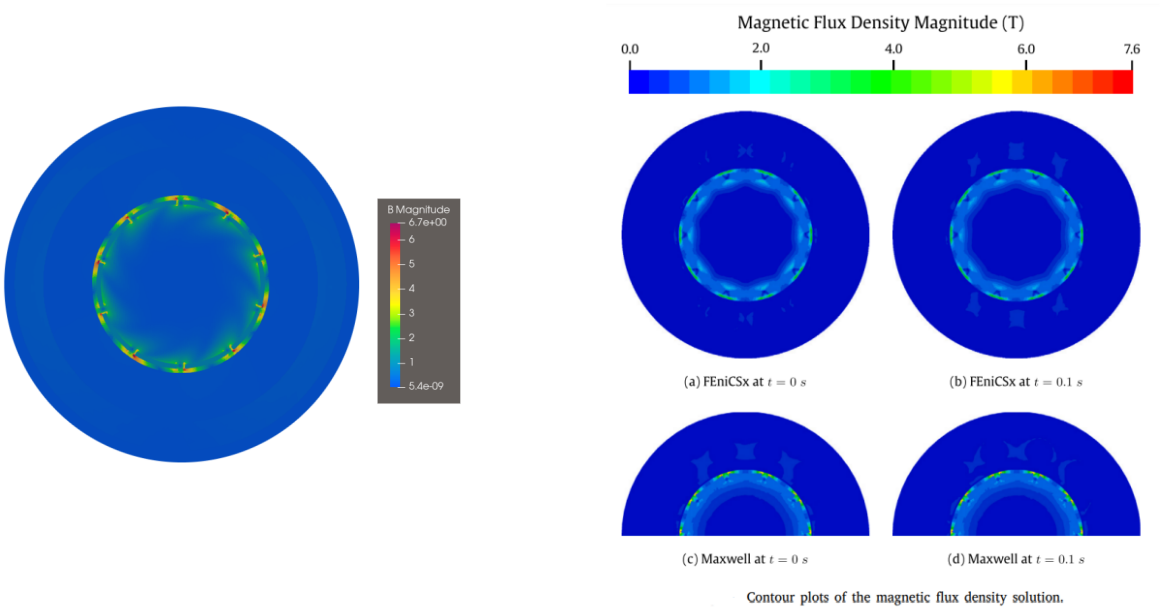
These outcomes validate the overall 2D electromagnetic model for the PMSM. The strong alignment of B around magnet poles and the near-uniform rotation of the vector potential confirm that:

- Material assignments and boundary conditions are producing physically consistent fields.
- Time-stepping is capturing the fundamental synchronous rotation as the magnets "carry" the rotor flux around the air gap.
- Mesh Resolution in the air-gap region is sufficient to capture peak flux densities without spurious oscillations or undue numerical diffusion.

Going forward, these fields can be used to calculate torque to understand the motor's performance.

4.1.3 Comparison with Reference Paper Results:

Figure 11a shows the magnetic flux density distribution obtained from our 2D PMSM model at t = 0.1 s. Figure 11b (taken from [14]) illustrates flux density results from a FEniCSx model and an Ansys Maxwell model [31] at t = 0 and t = 0.1 s. While each solver and geometry might have subtle differences the overall flux patterns are consistent. [32–35]



(a) Results from our model

(b) Results from reference paper

Figure 11: PMSM 2D: Magnetic Flux Density comparison with reference paper

Key Observations:

- **General Flux Shape:** In both our model and the reference solutions, the strongest flux density is concentrated near the permanent magnet poles, forming a ring of high $|\mathbf{B}|$ around the rotor edge. The center region (rotor shaft) and the outer air domain exhibit much lower values.

- **Amplitude Agreement:** The color scales indicate a similar peak flux on the order of 6 - 7 T in the magnets, which matches well with the reference paper's FEniCSx and Ansys Maxwell runs. Small discrepancies can arise from mesh resolution or material parameter assumptions.
- **Minor Differences:** Some directional asymmetry or sharper "shark-fin" flux arcs appear in our model. This often stems from how magnets are approximated or slight boundary condition differences. Another reason for this unsymmetric behavior may be due to the rotation of the magnetic field. Since the rotor is rotating in one direction the nature of the magnetic field tends to be not symmetrical.

Overall, our computed flux density distribution aligns well with the published FEniCSx and Ansys Maxwell results, both in magnitude and in shape (strong radial flux from magnet surfaces). The small deviations in color patterns and boundary transitions likely reflect modeling or meshing nuances rather than fundamental errors. Hence, this comparison demonstrates that our 2D PMSM finite-element model is producing physically realistic fields consistent with recognized solvers. It further validates the material assignments, magnetization approach, and boundary conditions used in our simulation pipeline.

4.1.4 Torque calculation

Figure 12 plots the torque vs. time at 600 RPM (62.83 rad/s). Torque is calculated in both ways: Surface torque & Volume torque. The red dashed line (surface torque) and blue solid line (volume torque) closely overlap, with nearly identical peak and average values. The average torque is approximately 1.0 Nm (surface) vs. 0.99Nm (volume), indicating less than 1% difference between the two calculations. Overlapping of both calculations indicate that the numerical computations are consistent across formulations.

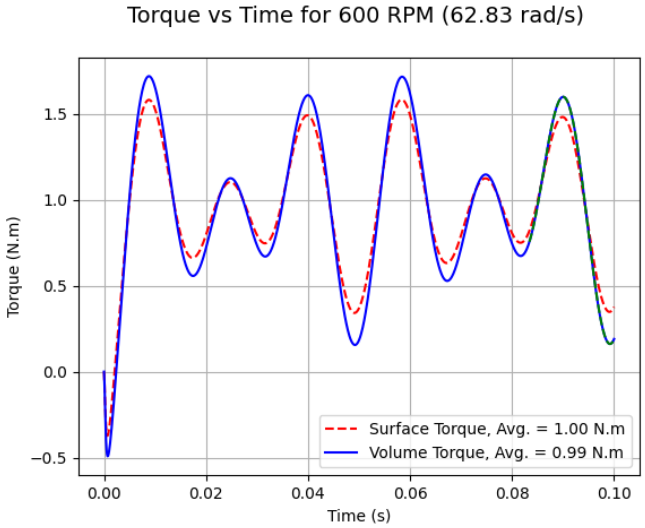


Figure 12: PMSM 2D: Torque calculation

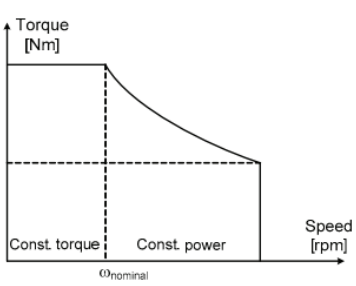
Surface-based integrals can be more sensitive to mesh quality right at the boundary, whereas the volume (Arkkio) method distributes the integration through the air gap field, often proving more stable. The nearly identical results here confirm both the meshing and PDE solution are robust.

4.1.5 Comparing the torque-speed curve of PMSM motors

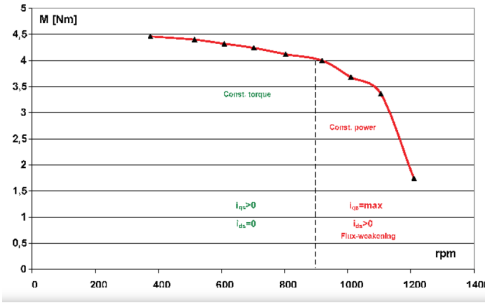
The torque-speed curve of a motor represents the relationship between the torque generated by the motor and its speed (RPM). This curve varies depending on the type of motor. Figure 13 shows torque–speed characteristics of:

1. Figure 13a: An ideal theoretical curve for a standard three-phase PMSM, which exhibits roughly constant torque up to its base speed, then transitions to a constant-power region (declining torque as speed increases further).

2. Figure 13b: An experimental result (from [20]) for a real motor test, used here mainly for the shape comparison—highlighting how real machines approximate that ideal form, but with some practical deviations.



(a) Ideal torque-speed characteristic for the three-phase PMSM motor



(b) Experimental result of Torque-speed curve from a reference paper [20]

Figure 13: Torque-speed curve comparison

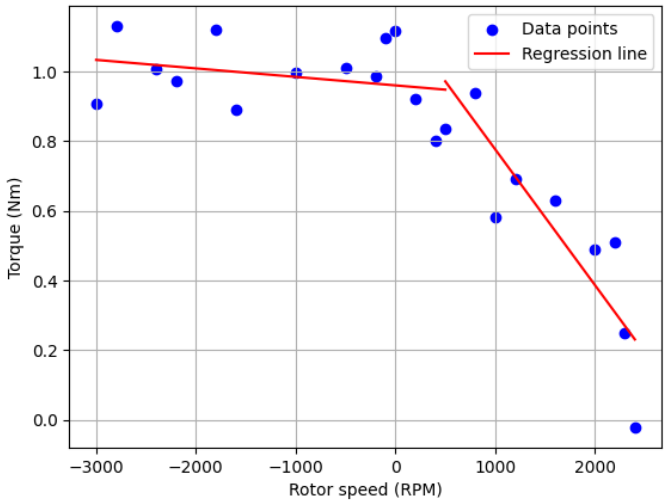


Figure 14: Torque-speed characteristics curve

Figure 14 shows our torque–speed data, obtained by varying the rotor speed in the finite-element model and plotting the average surface torque at each speed. The blue data points represent raw simulation results, and the red line is a piecewise linear regression fit to better visualize the trend.

Piecewise Linear Regression of Data: Two separate linear regression lines have been fitted as it allows for a better representation of the data structure compared to a single linear fit across

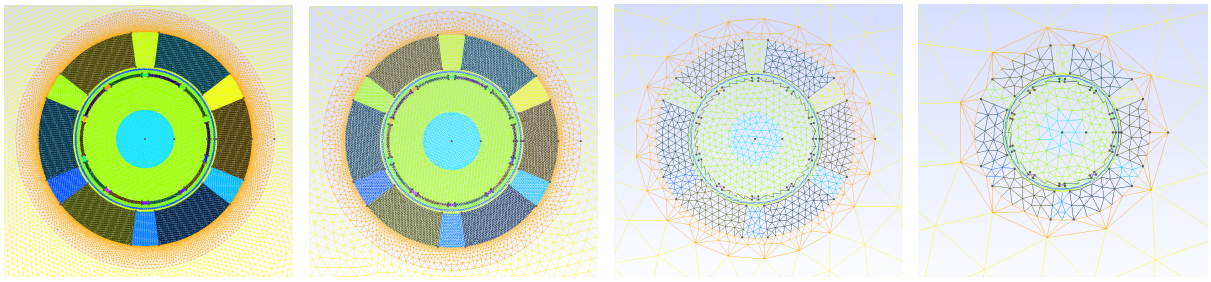
the entire range.
Region 1: In the left side of the plot, the torque remains nearly constant, with a slight downward trend.
Region 2: The right side of the plot shows a sharp decline in torque as the rotor speed increases.

Interpretations:

- **Shape Consistency:** Both the ideal and experimental curves show a flat torque region transitioning to a steep drop at high speeds. Our simulation matches with this expected performance of PMSM where the motor maintains a nearly constant torque at lower speeds before the torque starts dropping due to voltage and flux limitations at high speeds.
- **Piecewise Linear Fit:** Splitting the torque–speed data into two linear segments helps illustrate the near-constant torque at lower speeds (Region 1) and the rapid drop in Region 2. A single linear fit would obscure the distinct operational zones. The break point around 500 RPM suggests the transition between the constant torque region and the field-weakening region.
- **Comparisons to Literature:** The reference paper's result [20] demonstrates a similar torque drop beyond base speed.

4.1.6 Mesh Refinement

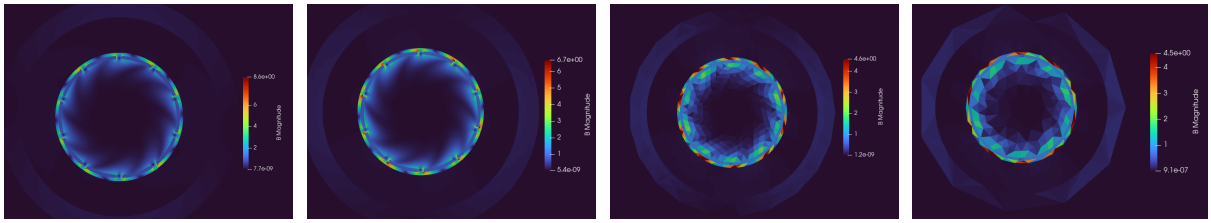
To analyze the impact of mesh resolution on computational efficiency and accuracy of finite element simulations, different mesh sizes were tested. As shown in Figure 15, meshes were generated with element sizes of 0.5 mm, 1 mm, 5 mm, and 10 mm. The finer meshes exhibit higher element density which are critical for electromagnetic field computations, particularly in regions of interest such as the stator and rotor interfaces.



(a) 0.5 mm (b) 1 mm (c) 5 mm (d) 10 mm

Figure 15: Meshes with different resolutions

The impact of mesh refinement on the computed magnetic field density is illustrated in Figure 16. A finer mesh provides a more detailed and accurate representation of the magnetic field distribution. In contrast, coarser meshes had noticeable distortions and loss of accuracy, particularly at the air-gap.



(a) 0.5 mm (b) 1 mm (c) 5 mm (d) 10 mm

Figure 16: Magnetic field densities for different resolutions

Mesh Resolution (mm)	# elements	# nodes	# DOFs	Execution time (s)	Avg. surface torque (Nm)	Avg. volume torque (Nm)	% diff
0.5	1,63,686	82,004	1,64,008	1365.8959	0.9081	0.9021	0.67%
1	41,564	20,782	41,564	77.8646	0.9023	0.9037	-0.15%
5	2,434	1,217	2,434	2.4745	1.0086	0.9158	10.13%
10	1,076	538	1,076	1.7006	2.3139	0.972	138.06%

Table 6: PMSM 2D: Mesh Refinement Summary

The effect of mesh refinement on computational efficiency and accuracy was analyzed by evaluating different mesh resolutions. Table 6 summarizes key metrics such as the number of elements, degrees of freedom (DOFs), execution time, computed torque values and the percentage difference between torque values for each mesh resolution.

Computational Cost vs. Accuracy

As expected, finer meshes (0.5 mm and 1 mm) have significantly higher element counts and degrees of freedom (DOFs), leading to increased execution times. The 0.5 mm mesh took approximately 1366 seconds, whereas the 1 mm mesh reduced execution time drastically to 77 seconds while maintaining accuracy. The coarser meshes (5 mm and 10 mm) reduced execution times further (2.47 s and 1.70 s, respectively) but resulted in substantial deviations in computed torque values.

Accuracy of Computed Torque Values

From the table it is clear that Avg. volume torque remains almost consistent across mesh sizes because of the way we calculate it. The average surface and volume torque values remain consistent for the finer meshes (0.5 mm and 1 mm), suggesting numerical stability. The percentage difference in volume torque between these two is only 0.67% and -0.15%, respectively. However, for the 5 mm and 10 mm meshes, the percentage difference rises significantly (10.13% and 138.06%, respectively). This suggests that these coarse meshes introduce numerical errors, leading to inaccurate torque predictions.

Optimal Mesh Selection

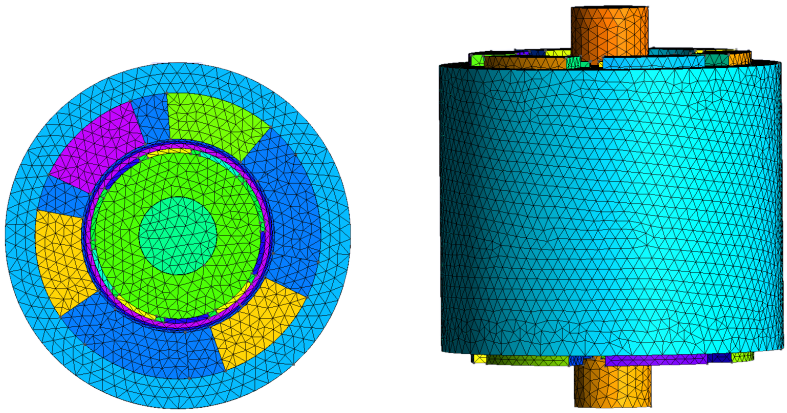
The 0.5 mm mesh, while the most accurate, is computationally expensive. The 1 mm mesh provides a near-optimal balance between computational efficiency and accuracy, achieving a small deviation in results while being 17 times faster than the finest mesh. The 5 mm and 10 mm meshes show significant deviations in results and should be avoided for high-accuracy simulations.

Using **adaptive meshing** (e.g., 0.5 mm mesh only in regions with steep gradients) can significantly improve performance without sacrificing much accuracy. Instead of applying a dense mesh across the entire domain, refining only the critical areas allows us to retain high accuracy where it is needed and reduce the total number of elements and computation time.

4.2 3D Results

4.2.1 PMSM 3D Mesh

Figures 17 and 18 show the 3D finite-element mesh for the PMSM model, generated by extruding the earlier 2D mesh in both axial directions. The rotor, stator, copper windings, and permanent magnet wedges from the 2D layout are preserved, while a cuboid air box is added around the cylinder to represent ambient space in 3D. Here, tetrahedral elements are the default choice for 3D meshing in Gmsh. Top, bottom and cylindrical surface of the motor are assigned suitable boundary conditions ($\mathbf{A} = 0$). This mesh can become significantly larger than in 2D, which may require more robust solvers or parallel computing resources.



(a) Top view (b) Side view

Figure 17: PMSM 3D: Mesh

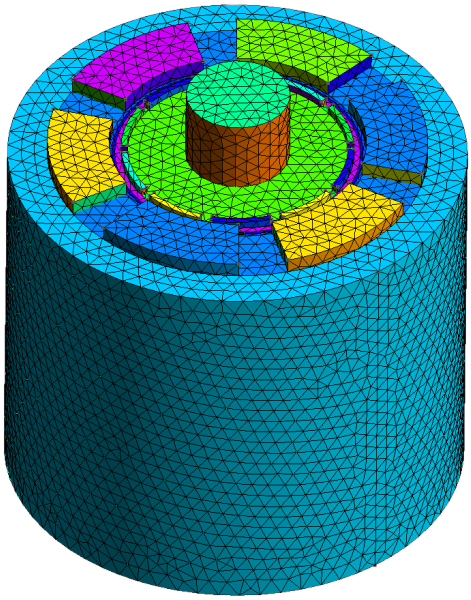
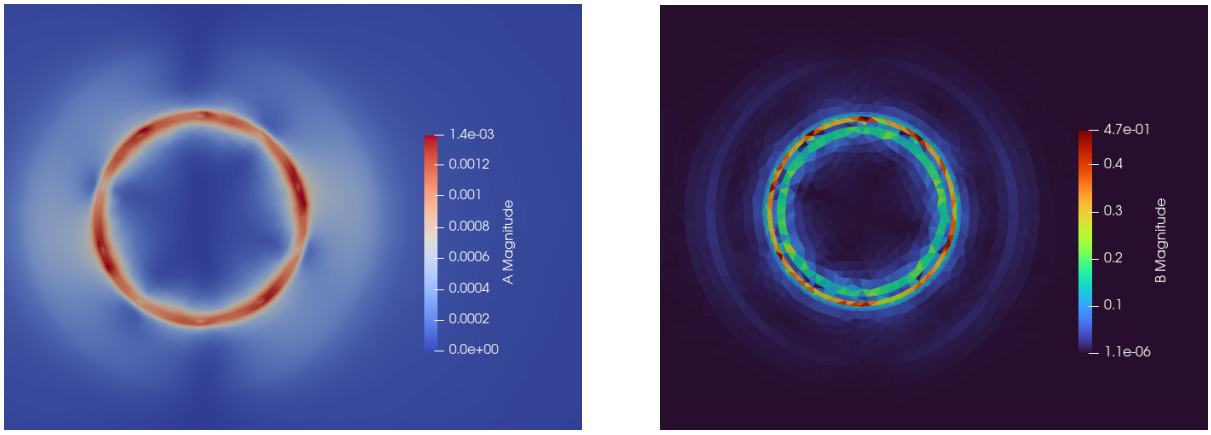


Figure 18: PMSM 3D: Mesh Isometric view

4.2.2 Magnetic vector potential (A) and Magnetic flux density (B)

Similar to 2D model, the primary 3D outputs are magnetic vector potential (A) and magnetic flux density (B)[12]. Figures 19 and 20 show cross-sections at the 100th time step for two planes:

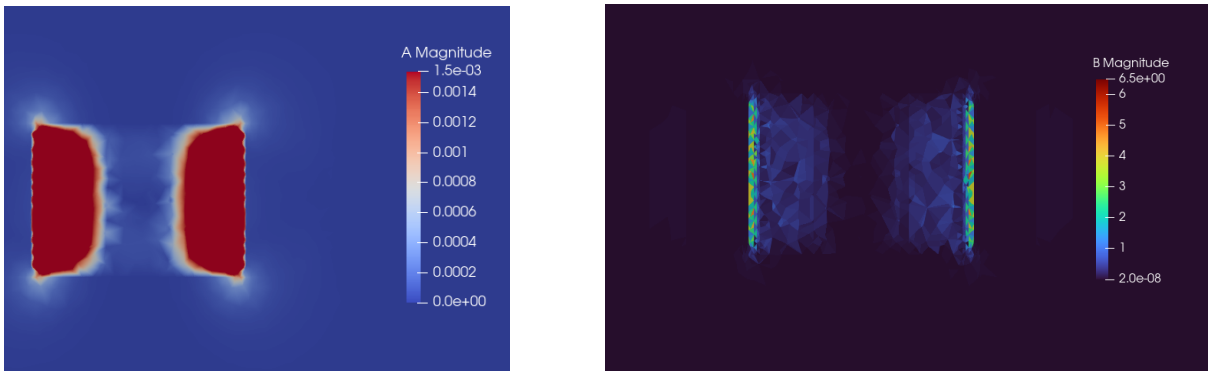
1. **XY Plane cross-section** (Figure 19): This horizontal slice passes through the center of the motor, effectively mirroring the 2D layout but within the 3D domain.
2. **YZ Plane cross-section** (Figure 20): A vertical slice, revealing how A and B vary along the axis of the machine and around the rotor/stator.



(a) Magnetic Vector Potential (A)

(b) Magnetic Flux Density (B)

Figure 19: PMSM 3D: Results - XY plane cross-section



(a) Magnetic Vector Potential (A)

(b) Magnetic Flux Density (B)

Figure 20: PMSM 3D: Results - YZ plane cross-section

Magnetic Vector Potential (A)

- **XY Plane:** Figure 19(a) shows a ring of higher magnitude A near the rotor's outer radius, correlating with the placement of permanent magnets. The field smoothly decreases toward the outer air domain, indicating minimal flux extension far beyond the stator.

[12] All simulation outputs are stored in .bp format (ADIOS2), and then post-processed in ParaView for visualization.

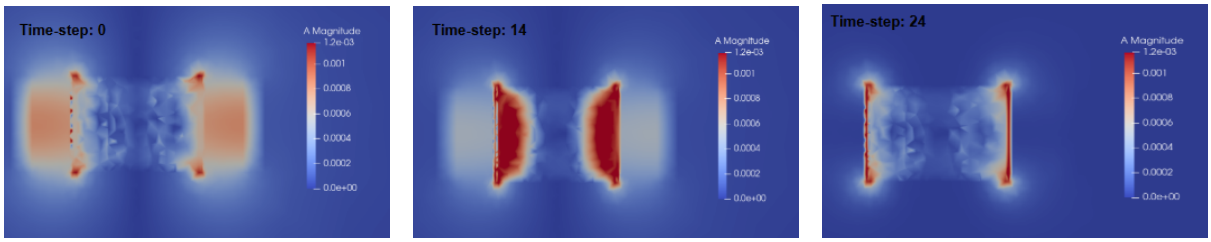
- **YZ Plane:** Figure 20(a) highlights how A extends vertically along the machine axis. Near the magnet surfaces, the magnitude peaks, then declines in the stator and external air.

Magnetic Flux Density (B)

- **XY Plane:** Figure 19(b) displays a circular band of high B around the rotor where the magnets reside, with flux crossing the air gap into the stator. Concentrations of flux near magnet edges are visible, similar to the 2D results, but now we can see flux smoothly distributed around the entire rotor circumference.
- **YZ Plane:** Figure 20(b) shows the axial variation of B. The strongest flux density occurs at the rotor rim (where permanent magnets contact the air gap) and diminishes along the radial direction. The dark areas at the top/bottom reflect the air region or rotor shaft, where flux is weaker.

To better visualize the rotating field over 100 time steps, short animation clips were generated. [13]These animations show how both stator and rotor fields evolve and rotate at a uniform rate.

Magnetic Vector Potential (A) at Different Time Steps:



(a) Figure 1 (b) Figure 2 (c) Figure 3

Figure 21: PMSM 3D: Magnetic vector potential (A) at different time steps

Figure 21 depicts the magnetic vector potential $\mathbf{A}$ in the XZ plane at three time steps: $t = 0$ (initial), $t = 14$, and $t = 24$. Each subfigure visualizes the magnitude of $\mathbf{A}$ with a color scale spanning from low (blue) to high (red). The figure provides a clear visualization of the rotating magnetic field in 3D. Each snapshot's $\mathbf{A}$ distribution corresponds to a different rotor angle in the time-stepping simulation, emphasizing the dynamic nature of PMSM operation.

[13]**Watch the rotation video for XY plane cross-section:**
Magnetic vector potential: click here
Magnetic flux density: click here

Watch the rotation video for YZ plane cross-section:
Magnetic vector potential: click here
Magnetic flux density: click here

5 CONCLUSION & FUTURE SCOPE

This project demonstrates the FEM simulation of a PMSM using the FEniCS platform. We start from the literature review of TEAM 30 problem understanding the FEniCS implementation of the induction motor model. For numerical formulation we started from the Maxwell's equation and derived the A-V formulation along with modeling permanent magnets and rotation. Then we derived the weak formulation and finally solving it in FEniCSx.

The two-dimensional model implementation started with generating the 2D mesh using GMSH. The mesh was then exported to xdmf to be compatible with FEniCSx. Variational form was converted to code using UFL library. PETSc was used to solve the linear system. The main result of the 2D model was the magnetic vector potential (A). Using A we calculated magnetic flux density (B) using Eq 3 and then the torque values. We compared the B result with the result of the reference paper and found similar. The torque-speed curve of PMSM motor was plotted and compared with the ideal curve and an experimental result. The curves were similar. Mesh refinement was carried out to find how the model behaves with different resolutions starting from finer (0.5 mm) to coarser (10 mm). Different metrics were analyzed and summarized which showed expected accuracy and computational cost.

For the three-dimensional implementation we started by focusing on the mathematical representation fron 2D to 3D and then understanding the key challenges and considerations to be taken. Shifting to 3D is essentially the same physics, but with more degrees of freedom, more complicated geometry, and bigger solver challenges. The 3D mesh was built from the 2D mesh by extruding the model and using tetrahedral elements. The weak formulation and its FEniCS implementation was carried accordingly and the results were generated in 3D. The primary results A and B were visualized in 2D by taking the cross section of XY plane and YZ plane. Animations were created to visualize the rotating magnetic field for 100 time steps.

There is significant scope for the next steps in this project. We could use Nédélec edge elements in place of nodal elements and compare the results. Nédélec elements are specifically designed to enforce the continuity of tangential components of vector fields, making them more suitable for solving Maxwell's equations. Also Nodal elements struggle to properly approximate curl-based formulations, whereas Nédélec elements are designed to directly work with H(curl) function spaces, leading to more accurate results. Another opportunity is to conduct a parameter study of the model to achieve an optimal balance between torque, efficiency, and stability, as PMSM motors are widely used in commercial applications that requires high performance, and energy efficiency.

Overall, this work studies the advancements in the electrification of jet engines and heavy vehicles in the transportation sector, paving the way for more efficient and sustainable solutions.

References

[1] Henke, M., Narjes, G., Hoffmann, J., Wohlers, C., Urbanek, S., Heister, & Ponick, B. (2018). Challenges and opportunities of very light high-performance electric drives for aviation. Energies, 11(2), 344. Dr. Cüneyt Sert, *ME 582, Finite Element Analysis in Thermofluids*

[2] Alnæs, M., Blechta, J., Hake, J., Johansson, A., Kehlet, B., Logg, A., ... & Wells, G. N. (2015). The FEniCS project version 1.5. Archive of numerical software, 3(100).

[3] Logg, A., Mardal, K. A., & Wells, G. (Eds.). (2012). Automated solution of differential equations by the finite element method: The FEniCS book (Vol. 84). Springer Science & Business Media.

[4] Pillay, P., & Krishnan, R. (1989). Modeling, simulation, and analysis of permanent-magnet motor drives. I. The permanent-magnet synchronous motor drive. IEEE Transactions on industry applications, 25(2), 265-273.

[5] Zienkiewicz, O. C., Taylor, R. L., & Zhu, J. Z. (2005). The finite element method: its basis and fundamentals. Elsevier.

[6] Reddy, J. N. (1993). An introduction to the finite element method. New York, 27(14).

[7] Fish, J., & Belytschko, T. (2007). A first course in finite elements (Vol. 1). New York: Wiley.

[8] Periodic Table of the Finite Elements - https://www-users.cse.umn.edu/ arnold/femtable/index.html

[9] Baratta, I. A., Dean, J. P., Dokken, J. S., Habera, M., HALE, J., Richardson, C. N. & Wells, G. N. (2023). DOLFINx: the next generation FEniCS problem solving environment.

[10] Alnæs, M. S., Logg, A., Ølgaard, K. B., Rognes, M. E., & Wells, G. N. (2014). Unified form language: A domain-specific language for weak formulations of partial differential equations. ACM Transactions on Mathematical Software (TOMS), 40(2), 1-37.

[11] Kirby, R. C., & Logg, A. (2006). A compiler for variational forms. ACM Transactions on Mathematical Software (TOMS), 32(3), 417-444.

[12] Scroggs, M. W., Dokken, J. S., Richardson, C. N., & Wells, G. N. (2022). Construction of arbitrary order finite element degree-of-freedom maps on polygonal and polyhedral cell meshes. ACM Transactions on Mathematical Software (TOMS), 48(2), 1-23.

[13] Loeliger, J., & McCullough, M. (2012). Version Control with Git: Powerful tools and techniques for collaborative software development. " O'Reilly Media, Inc.".

[14] McDonagh, J., Palumbo, N., Cherukunnath, N., Dimov, N., & Yousif, N. (2022). Modelling a permanent magnet synchronous motor in FEniCSx for parallel high-performance simulations. Finite Elements in Analysis and Design, 204, 103755.

[15] TEAM Benchmark Problems, Problem No. 30a - Induction Motor Analyses, - https://www.compumag.org/jsite/images/stories/TEAM/problem30a.pdf

[16] Testing Electromagnetic Analysis Methods (T.E.A.M.) - https://www.compumag.org/wp/team/

[17] Github: Wells-Group/TEAM30 - https://github.com/Wells-Group/TEAM30/

[18] Geuzaine, C., & Remacle, J. F. (2009). Gmsh: A 3-D finite element mesh generator with built-in pre-and post-processing facilities. International journal for numerical methods in engineering, 79(11), 1309-1331.

[19] Ayachit, U. (2015). The paraview guide: a parallel visualization application. Kitware, Inc..

[20] Aiso, K., & Akatsu, K. (2022). Performance comparison of high-speed motors for electric vehicle. World Electric Vehicle Journal, 13(4), 57.

[21] Kuczmann, M., & Iványi, A. (2008). The finite element method in magnetics. Akadémiai Kiadó.

[22] Jin, J. M. (2015). The finite element method in electromagnetics. John Wiley & Sons.

[23] Nédélec, J. C. (1980). Mixed finite elements in R3. Numerische Mathematik, 35(3), 315-341.

[24] Biro, O., Preis, K., & Richter, K. R. (2002). On the use of the magnetic vector potential in the nodal and edge finite element analysis of 3D magnetostatic problems. IEEE Transactions on magnetics, 32(3), 651-654.

[25] Bastos, J. P. A., & Sadowski, N. (2003). Electromagnetic modeling by finite element methods. CRC press.

[26] Zhang, Q., & Cen, S. (Eds.). (2015). Multiphysics Modeling: Numerical Methods and Engineering Applications: Tsinghua University Press Computational Mechanics Series. Elsevier.

[27] Lubin, T., & Rezzoug, A. (2015). 3-D analytical model for axial-flux eddy-current couplings and brakes under steady-state conditions. IEEE Transactions on Magnetics, 51(10), 1-12.

[28] Bastos, J. P. A., & Sadowski, N. (2013). Magnetic materials and 3D finite element modeling. CRC press.

[29] Song, H., & Ida, N. (1991). An eddy current constraint formulation for 3d electromagnetic field calculations. IEEE transactions on magnetics, 27(5), 4012-4015.

[30] Hale, J. S., Li, L., Richardson, C. N., & Wells, G. N. (2017). Containers for portable, productive, and performant scientific computing. Computing in Science & Engineering, 19(6), 40-50.

[31] Cendes, Z. J., Smetak, E. C., & Shenton, D. (1988). Mesh generation for modelling in magnetics. Mathematical and Computer Modelling, 11, 192-196.

[32] G.-J. Park, Y.-J. Kim, S.-Y. Jung, Design of IPMSM applying V-Shape skew considering axial force distribution and performance characteristics according to the rotating direction, IEEE Trans. Appl. Supercond. 26 (4) (2016) 1–5, http://dx.doi.org/10.1109/TASC.2016.2543267.

[33] Sizov, Gennadi Y., Dan M. Ionel, and Nabeel AO Demerdash. ”Modeling and parametric design of permanent-magnet AC machines using computationally efficient finite-element analysis.” IEEE Transactions on Industrial Electronics 59.6 (2011): 2403-2413.

[34] Zhu, Z. Q., Jewell, G. W., & Howe, D. (2000, January). Finite element analysis in the design of permanent magnet machines. In IEE seminar on current trends in the use of finite elements (FE) in electromechanical design and analysis (ref. no. 2000/013) (pp. 1-1). IET.

[35] Khoshkar, H. F., & Doroudi, A. (2014). Finite element analysis of permanent magnet synchronous motors subjected to symmetrical voltage sags. Iranian Journal of Electrical and Electronic Engineering, 4

[36] Wang, X., Wang, X. B., & Gascoyne, P. R. (1997). General expressions for dielectrophoretic force and electrorotational torque derived using the Maxwell stress tensor method. Journal of electrostatics, 39(4), 277-295.

[37] Arkkio, A. (1987). Analysis of induction motors based on the numerical solution of the magnetic field and circuit equations. Helsinki University of Technology.

Appendix A

The latest version of the FEniCS project, **FEniCSx**, consists of several building blocks, namely DOLFINx, UFL, FFCx, and Basix.

- **DOLFINx** is the high performance C++ backend of FEniCSx, where structures such as meshes, trial and test functions are implemented [9]. It also contains compute intensive finite element assembly and mesh refinement algorithms and provides an interface to linear algebra solvers and data-structures, such as PETSc.
- **UFL** is a high-level form language for describing variational formulations with a high-level mathematical syntax [10].
- **FFCx** is the form compiler of FEniCSx; given variational formulations written with UFL, it generates efficient C code [11].
- **Basix** is the finite element backend of FEniCSx, responsible for generating finite element basis functions [12].

Why Use FEniCS?

1. **Easy Discretization** – Converts differential equations to discrete form in a few lines, automating matrix assembly and integration.
2. **Python + C++** – Combines Python's simplicity with C++'s speed for efficient coding.
3. **Open-source** – FEniCS is a leading open-source software for solving PDEs using FEM, making it a powerful tool for scientific computing. It is actively developed, well-documented, and supported by a strong community.
4. **Parallel Computing** – FEniCS comes with built-in parallelization that helps in significant reduction of run time of simulations.
5. **HPC-Optimized** – Uses PeTSc and XDMF for scalable, efficient computations on clusters. We can code our implementation on our laptop and that same code would work on computational clusters that supports HPC.

Git

Git is a version control system that helps to track changes in our code, collaborate with others, and maintain different versions of our project. It is widely used in software development, especially for open-source projects. [13] For this project Git is used in the following ways:

- Track changes in Python scripts, mesh files, and simulation parameters.
- Allowing others to review, and improve the code.
- Creating separate branches for experimenting different features and adding them to the main code using pull request.
- Maintain a **README.md** file to explain the project, installation steps, usage and showcasing results.
- Store simulation outputs, plots, or reports in a structured manner.

Appendix B

Nodal shape functions: Scalar potential functions can be represented by a linear combination of shape functions associated with nodes of the finite element mesh [21, 22]. Within a finite element, a scalar potential function V is approximated by

$$V \approx \sum_{i=1}^{m} N_i V_i$$

where N_i and V_i are the nodal shape functions and the value of potential function corresponding to the ith node, respectively. The number of degrees of freedom is m = 3 in a 2D problem using triangular FEM mesh, and the shape functions are linear, and the numer of degree of freedom is m = 6, when the shape functions are second order. The nodal shape functions can be defined by the relation

$$N_i = \begin{cases} 1, & \text{at the node } i, \\ 0, & \text{at other nodes.} \end{cases}$$

Edge shape functions: Vector potentials can be represented by either nodal shape functions or edge shape functions. Edge shape functions are also called vector shape functions[21, 22].Edge shape functions are higher-order interpolation functions used in FEM to define variations along element edges. Unlike nodal shape functions, which are associated with element vertices, edge shape functions are linked to element edges and help capture rotational effects in elements. Nédélec shape functions are a type of edge shape functions specifically designed for solving problems involving vector fields that require curl-conformity, like electromagnetic problems governed by Maxwell's equations. Nédélec elements ensures tangential continuity of vector fields across elements [23].

In our models we use nodal shape function for approximating **A**. There is scope for using Nédélec shape functions and analyzing the solutions as a next step.

Appendix C

The code repository for this project on GitHub: https://github.com/abhinavtk7/pmsm-fenics This repository is based on the `TEAM30` project Github: Wells-Group/TEAM30 and is configured to run inside a Docker container with `dolfinx`. This guide provides step-by-step instructions to set up and run the project.

Prerequisites

Ensure you have the following installed on your system:

- Docker (Required for running the containerized environment)
- Git (Required for cloning the repository)
- VS Code (Recommended for editing and debugging the project)

Setup

Step 1: Clone the Repository

```
git clone git@github.com:abhinavtk7/pmsm-fenics.git
cd pmsm-fenics
```

Step 2: Start the Docker Container

Run the following command from the project root to start the container:

```
docker run -ti -v $(pwd):/root/shared -w /root/shared/ --shm-size=512m
--name=pmsm ghcr.io/fenics/dolfinx/dolfinx:v0.7.0
```

This command:

- Mounts the **current directory** to `/root/shared/` inside the container.
- Sets the working directory to `/root/shared/`
- Allocates shared memory (`--shm-size=512m`) to prevent memory issues.
- Names the container `pmsm`

Step 3: Install Dependencies

Inside the container, install required Python packages:

```
python3 -m pip install tqdm pandas
```

Step 4: Verify Installation

Run the Python script `pmsm_msh.py` with a resolution of 0.01 mm inside the container to verify the setup:

```
python3 scripts/pmsm_msh.py --res 3 --progress
```

If the code runs successfully, the setup is complete.

Usage

Starting the Docker Container Again

If you exit the container, restart it using:

```
docker start -ai pmsm
```

If the container was removed, recreate it using the command from Step 2.

Stopping the Docker Container

Exit the container by running:

```
exit
```

Or stop it from another terminal:

```
docker stop pmsm
```

Removing the Docker Container

To remove the container completely:

```
docker rm pmsm
```

Using VS Code for Editing and Debugging

If you want to edit or debug the code, you can do it using VS Code.

Open VS Code and Attach to the Running Container

1. Launch VS Code.
2. If you haven't installed the **Remote - Containers** extension, install it.
   - Open Extensions (`Ctrl + Shift + X`).
   - Search for `Remote - Containers` and click **Install**.
3. Press `Ctrl + Shift + P` to open the command palette.
4. Type and select **Attach to Running Container**.
5. Choose the container named `pmsm` (Make sure you've started the container).
6. Once attached, you will be inside the `pmsm` container.

Appendix D

Code snippet of Trial and test functions expression and weak formulation definition using ufl.

```
cell = mesh.ufl_cell()    # 'triangle'
FE = ufl.FiniteElement("Lagrange", cell, degree)    # degree = 1
ME = ufl.MixedElement([FE, FE])
VQ = fem.FunctionSpace(mesh, ME)

# Define test, trial and functions for previous timestep
Az, V = ufl.TrialFunctions(VQ)
vz, q = ufl.TestFunctions(VQ)
AnVn = fem.Function(VQ)
An, _ = ufl.split(AnVn)  # Solution at previous time step
JOz = fem.Function(DG0)  # Current density


# Create integration sets
domains = {"Air": (1,), "AirGap": (2, 3), "Al": (4,), "Rotor": (5, ),
           "Stator": (6, ), "Cu": (7, 8, 9, 10, 11, 12),
           "PM": (13, 14, 15, 16, 17, 18, 19, 20, 21, 22)}

Omega_n = domains["Cu"] + domains["Stator"] + domains["Air"] +
                                              domains["AirGap"]
Omega_c = domains["Rotor"] + domains["Al"] + domains["PM"]
Omega_pm = domains["PM"]

# Magnetization part
coercivity = 8.38e5  # [A/m]
DG0v = fem.FunctionSpace(mesh, ("DG", 0, (2,)))
Mvec = fem.Function(DG0v)


# Create integration measures
dx = ufl.Measure("dx", domain=mesh, subdomain_data=ct)

# Define temporal and spatial parameters
dt = fem.Constant(mesh, dt_)
x = ufl.SpatialCoordinate(mesh)

omega = fem.Constant(mesh, default_scalar_type(omega_u))

# Motion voltage term
u = omega * ufl.as_vector((-x[1], x[0]))

# Magnetization term
curl_vz = ufl.as_vector((vz.dx(1), -vz.dx(0)))
mag_term =  (mu_0/mu) * ufl.inner( Mvec , curl_vz) * dx(Omega_pm)

```

```
# Define variational form
f_a =   + dt / mu * ufl.inner(ufl.grad(Az), ufl.grad(vz)) * dx(Omega_n +
                                                              Omega_c) \
        + dt / mu * vz * (n[0] * Az.dx(0) - n[1] * Az.dx(1)) * ds \
        + sigma * (Az - An) * vz * dx(Omega_c) \
        + dt * sigma * ufl.dot(u, ufl.grad(Az)) * vz * dx(Omega_c) \
        - dt * J0z * vz * dx(Omega_n) \
        - dt * mag_term

f_v =   + dt *sigma*(V.dx(0)*q.dx(0) + V.dx(1)*q.dx(1)) * dx(Omega_c)

form_av = f_a + f_v
a, L = ufl.system(form_av)
```

Appendix E

Torque calculation equations' derivation:

Surface torque

We start by defining Maxwell's stress tensor

$$\sigma_{ij} = \varepsilon E_i E_j + \frac{1}{\mu_0} B_i B_j - \frac{1}{2}\left(\varepsilon \mathbf{E}\cdot\mathbf{E} + \frac{1}{\mu_u}\mathbf{B}\cdot\mathbf{B}\right)\delta_{ij}$$

where $\mathbf{E}$ is the electric field, $\mathbf{B}$ the magnetic flux density and δ_{ij} the Kronecker-delta function. We would like to compute the traction at a surface Γ with normal n, i.e. $d\mathbf{F} = \mathbf{n}\cdot\sigma = \sigma_{ij}n_i$, which can be written on vector form as

$$\begin{aligned} \mathrm{d}\mathbf{F} &= \varepsilon_0(\mathbf{n}\cdot\mathbf{E})\mathbf{E} + (\mathbf{n}\cdot\mathbf{B})\mathbf{B} - \frac{1}{2}\left(\varepsilon_0\mathbf{E}\cdot\mathbf{E} + \frac{1}{\mu_0}\mathbf{B}\cdot\mathbf{B}\right)\mathbf{n} \\ &= \frac{1}{\mu_0}(\mathbf{n}\cdot\mathbf{B})\mathbf{B} - \frac{1}{2\mu_0}(\mathbf{B}\cdot\mathbf{B})\mathbf{n}. \end{aligned} \tag{17}$$

where the last equality comes from the fact that we are interested in the stress tensor in the air surrounding the engine, where $\mathbf{E}$ is $\mathbf{0}$.
The electromagnetic torque is obtained by the integral

$$\mathbf{T_e} = L\int_\Gamma (\mathbf{r}\times\mathrm{d}\mathbf{F})\ \mathrm{d}\Gamma = L\int_\Gamma \frac{1}{\mu_0}(\mathbf{B}\cdot\mathbf{n})\mathbf{r}\times\mathbf{B} - \frac{1}{2\mu_0}(\mathbf{B}\cdot\mathbf{B})\mathbf{r}\times\mathbf{n}\ \mathrm{d}\Gamma, \tag{18}$$

where L is the depth of the domain and $\mathbf{r}$ is the position vector from the rotation axis to a point on the surface.

2D simplification for surface torque:

In $2D$ as $\mathbf{r} = (x, y, 0)$, $\mathbf{B} = (B_x, B_y), 0)$, $\mathbf{n} = (n_x, n_y, 0)$ then

$$T_{e,z} = L\int_\Gamma \frac{1}{\mu_0}(\mathbf{B}\cdot\mathbf{n})(r_x B_y - r_y B_x) - \frac{1}{2\mu_0}(\mathbf{B}\cdot\mathbf{B}(r_x n_y - r_y n_x)\mathrm{d}\Gamma \tag{19}$$

The boundary Γ is a surface in the air-gap enclosing the rotor.

Volume torque: Arkkio's method

It has been shown by Arkkio that a volume formulation of the torque is more stable than the surface integration when using numerical approximations. The formulation used is:

$$\mathbf{T_e} = \frac{1}{\mu_0(r_o - r_i)}\int_{AirGap} rB_r B_\theta\ \ \mathrm{d}x, \tag{20}$$

where r_i, r_o is the inner and outer radius of the air gap, B_r the radial component of the magnetic field, B_θ the azimuthal direction, and $r = |\mathbf{r}|$ [37].