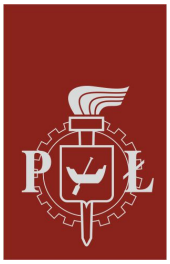

Lodz University of Technology
Faculty of Mechanical Engineering
Institute of Turbomachinery

inż. Michał Wilczek
Index no: 220430

Master Thesis in
Advanced Mechanical Engineering
Full-time studies

Finite Element Analysis of Magneto-Thermal Transient Effects in Superconducting Accelerator Magnets

Lodz University of Technology Supervisor
dr hab. inż. Artur Gutkowski

CERN Supervisor
dr inż. Michał Maciejewski

Łódź, January 2020

Abstract

In this thesis, an electro-thermal multidimensional model is constructed in ANSYS APDL to simulate self-protectability scenarios of superconducting magnets. The analysis takes the longitudinal and the turn-to-turn quench propagation into account. The developed method has been illustrated with a case of the skew quadrupole, being one of the high-order corrector magnets for the High-Luminosity LHC upgrade project.

The quench propagation in superconducting magnets is a magneto-thermal effect. This is a challenging task to simulate due to multi-physics, multi-rate, multi-domain, and multi-scale problems. Moreover, high material nonlinearities must be studied as a function of temperature and, in some cases, magnetic field strength.

The possibility of conducting the quench studies in ANSYS APDL is verified first. The models, simulating the 1D quench propagation, are prepared in ANSYS and compared with STEAM-BBQ, being a tool for the 1D quench propagation implemented in COMSOL and developed at TE-MPE-PE at CERN.

The standard solution of the heat balance equation is non-feasible due to steep temperature gradients and high nonlinearities. Therefore, to avoid the explicit solution of the quench front propagation, the ANSYS model relies on an external routine that imposes a constant quench velocity along the conductor based on simplified 1D numerical studies. The methodology is verified with standard numerical analyses performed in ANSYS. This thesis also covers algorithms and a Python code that were developed to implement the proposed methodology in ANSYS APDL. Advantages and disadvantages of the developed methodology are thoroughly discussed.

Acknowledgment

With the end of this work, I would like to sincerely thank my CERN supervisor, Dr Michał Maciejewski from the STEAM group. Our multiple discussions over the phenomena governing the simulations, the algorithms related to them, and programming in general, allowed me to develop professionally in the given field. Moreover, I would like to thank him for many comments on writing, based on which I learnt how to ask the right research questions.

I would like to express words of gratitude to my supervisor from Lodz University of Technology, dr hab. inż. Artur Gutkowski. His flexible attitude to organise meetings in a convenient time for me were very helpful, as I was preparing this thesis abroad, far from my home university.

I would also like to mention the former and current STEAM members, the STEAM leader Dr Arjan Verweij, as well as Lorenzo Bortot, Dr Matthias Mentink, and Dr Emanuele Ravaioli who participated in my presentations and gave me many comments on the possibility of improving my work.

I am grateful to the researchers from INFN, Dr Marco Prioli and Samuele Mariotto, for our collaboration concerning the analysis of the skew quadrupole. Moreover, I would also like to mention Simon McIntosh from ITER Magnet Division for the introduction to the problematics of the quench velocity-based approach.

Besides, I would like to thank my colleagues Andreas Will and Dr Thomas Cartier-Michaud for their help in Python scripting as well as the cheerful atmosphere in the office, thanks to which my daily work at CERN was a pure pleasure. Last but not least, I am grateful to my friends, Douglas and Jonathan whose numerous comments on my writing allowed me to correct this thesis to be light and easy to read.

Contents

Chapter 1

Introduction

1.1 About CERN and LHC

The European Organisation for Nuclear Research (CERN) is one of the largest organisations performing scientific research in fundamental particle physics. CERN aims at providing particle accelerator facilities for numerous experiments conducted in a strong international collaboration. The Organisation is situated in the French-Swiss border in the Geneva area. Since 2008, CERN has been operating the largest particle accelerator ever built, called the Large Hadron Collider (LHC). The LHC has become the last component in the CERN accelerator complex. The accelerator consists of a 27 km-long ring with two beam pipes in which particles travel in the opposite directions and are accelerated over multiple accelerator turns. The schematic of the LHC is illustrated in Fig. 1.1.

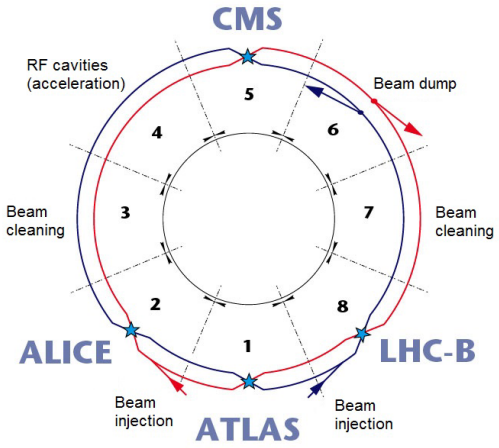

Figure 1.1: Schematic representation of the LHC [1].

In the last operation of the LHC lasting in the period of 2015-2018, called "Run 2", the LHC was accelerating protons to the energy of $E = 6.5$ TeV in each of its two beams [2]. The higher the energy that is attained, the deeper one can study the particle showers obtained during the collisions of the particle beams. They allow for unravelling the fundamentals of particles and interactions between them. There are four collision points in the LHC with different colliders installed that analyse the data from the particle showers: ATLAS[1], CMS[2], ALICE[3] and LHCb[4]. ATLAS and CMS are two large general-purpose detectors whereas ALICE is specialised in heavy-ion physics and LHCb – in matter-antimatter asymmetry. [3, p. 3-21]

As presented in Fig. 1.1, the LHC is divided into eight sectors. The interaction points, in which the detectors are installed, are situated in four sectors. The sectors two and eight serve for injecting the initially accelerated beam from other CERN accelerators. In sector four, RF cavities accelerate the beam at every turn until the particles reach the desired energy. In sector six, the LHC is equipped with a beam dump system which extracts the beam from the tunnel in case of a machine failure or if the beam quality deteriorates. [4, p. 1-4]

The LHC is currently being maintained during the so called "Long Shut Down 2" (LS2) for an upgrade of the High-Luminosity LHC which is planned to operate in the years 2021-2024 referred to as "Run 3". The project aims at increasing the frequency of particle collisions, also described as luminosity. Moreover, the goal of the HL-LHC is to accelerate the particles to the energy of $E = 7$ TeV. [2, 5]

1.2 Superconductors and Quench Problem

1.2.1 Available Superconducting Materials

At the beginning of the 20^{th} century, Heike Kamerlingh Onnes liquified helium and started using it to cool down various metals to extremely low temperatures. He discovered that the resistance of solid mercury disappeared at $T = 4.2$ K and named this phenomenon as a "superconductivity". Since that period, there have been many materials discovered, each characterised by similar physical properties corresponding to superconductivity. Using superconductors allowed the accelerator magnets to operate at higher magnetic field compared to normal conducting technologies, which consume a predominant amount of energy due to the Joule heating. There are two main types of superconductors distinguished by the temperature at which they obtain their superconducting state:

[1] ATLAS – A Toroidal LHC ApparatuS.
[2] CMS – Compact Muon Solenoid.
[3] ALICE – A Large Ion Collider Experiment.
[4] LHCb – Large Hadron Collider beauty.

1. Low-temperature superconductors (LTS)

2. High-temperature superconductors (HTS)

LTS-based cables can operate in a superconducting state in the range of $T \in (4, 20)$ K whereas HTS-based ones – at temperatures higher than $T = 20$ K. The HTS technology is very promising for future applications, because it will enable machines for an operation at higher magnetic fields while spending less energy to cool it. However, the HTS cables still remain at the stage of research and development in most of the applications concerning accelerator magnets. [3, p. 77-95]

In order to use a superconductor in engineering applications for a magnet design, it must meet four basic requirements [3, p. 77-95]:

1. The material should have appropriate basic physical properties, i.e. high critical temperature and critical magnetic field.

2. The material must be relatively common, i.e. reasonably cheap.

3. The material must be easy to form into common circular or rectangular shapes in order to use it for the fabrication of tapes or wires.

4. The material must withstand mechanical loads inside a magnet.

There are two low-temperature superconductors that are commercially available for a large-scale magnet production: Nb-Ti and Nb_3Sn. Each of these meet most of the specified technical requirements. Nb-Ti is an alloy widely used in a magnet design due to its ductility that allows for an easy cable extrusion and further winding process. On the contrary, Nb_3Sn is a brittle material sensitive to mechanical stresses. It requires a greater effort in coil production compared to Nb-Ti. In the fabrication process, it is initially formed to resemble its final geometry. The next manufacturing step consists of a multiple day high-temperature treatment during which the mixture of Nb and Sn binds in order to create a superconducting compound. The main advantage of Nb_3Sn is its higher critical parameters of temperature and magnetic field with respect to Nb-Ti. [6, p. 29-41]

1.2.2 Critical Parameters and Stability

The superconducting state depends on three critical parameters[5]: temperature, magnetic field, and current density, as illustrated in Fig. 1.2. These three variables create the so called

[5] In case of Nb_3Sn, the superconducting state also depends on the mechanical strain. However, this thesis focuses on the Nb-Ti technology in which such a phenomenon does not occur.

"critical surface" under which a material remains in the superconducting state. The transition from the superconducting to the normal conducting state is called a "quench". It occurs when an operating point of a superconductor moves outside of the critical surface. One can notice in Fig. 1.2 that Nb_3Sn is characterised by a larger volume under the critical surface with respect to Nb-Ti. Higher critical parameters of this material allow for its potential usage at stronger fields in accelerator magnets.

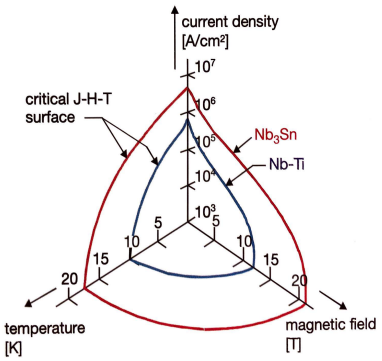

Figure 1.2: Critical surface for Nb_3Sn and Nb-Ti [3]

At operating temperatures of an LTS-based magnet, i.e. being in the range of $T \in (1.9, 4.5)$ K, even a small energy deposition in the order of 5 $\frac{\text{mJ}}{\text{cm}^3}$ may cause a quench. It is mainly due to an extremely low heat capacity of solid materials at cryogenic temperatures (see Appendix A). A superconductor is embedded in a copper matrix which [6, p. 31-33]:

1. improves the mechanical properties of a superconductor such as mechanical strength and ductility;
2. provides an alternative path for the current flow; and
3. increases the thermal stability of a superconductor by acting as a heat sink.

Starting from the left picture in Fig. 1.3, one can observe the groups of superconductor filaments. The diameter of a single filament is usually approximately equal to 10 μm. The picture in the middle represents a cross-section of a strand with multiple groups of filaments marked in grey that are embedded in a copper matrix. The diameter of a single strand usually accounts for 1 mm. The superconducting coil of a magnet can be either wound with strands or with Rutherford cables (picture in the right) that are a set of twisted strands. The Rutherford cable allows for limiting eddy currents in the copper matrix. The composite fabrication should ensure a good contact between the copper and a superconductor. The copper should also be characterised by a high residual resistivity ratio (RRR), i.e. it

ought to be highly conductive both electrically and thermally. The definition of RRR is given in Appendix A.1.1. [6, p. 31-33]

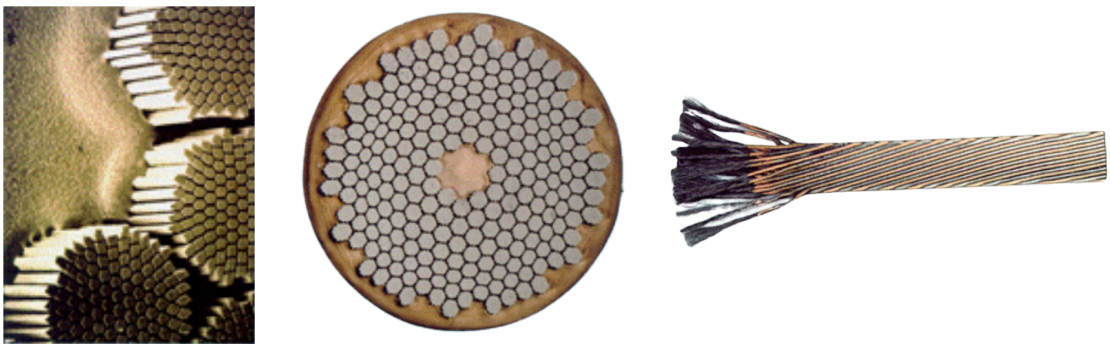

Figure 1.3: Left: groups of filaments embedded in a copper matrix, centre: strand cross-section with multiple groups of filaments, right: Rutherford cable [7].

The strand stability is also increased due to liquid helium whose main role is to cool down the magnet to its operating conditions. Liquid helium is characterised by a high heat capacity at operating temperatures of LTS magnets with respect to solids. By assuring a good thermal contact between the strand and the liquid, one can increase the system thermal stability. However, in many applications the strand or a stack of strands is fully impregnated with resin that does not enable helium to fully penetrate the cable. In case of a fully impregnated coil, the direct contact between the liquid and the strand is negligible and cooling of a quenched zone is only possible through a longitudinal heat propagation along the strand. [6, p. 122]

Taking into consideration a superconducting magnet as a whole, composed of multiple strands, it may quench because of, among many, three reasons:

1. A coil winding can move due to the action of the Lorentz forces. The frictional movement between different windings would result in the generation of heat.

2. A part of a particle beam may be deposited in the beam pipe which would lead to the creation of a particle shower and, thus, to the generation of heat as well.

3. A malfunction of a cryogenic system can also be the reason for quench of a superconducting magnet.

1.2.3 Current Sharing Phenomenon

When a composite strand operates at the critical surface of a superconductor, the current starts commuting in both elements of the composite. This phenomenon is called "current

sharing". At this point, it is assumed that the superconductor carries the current of a critical value I_c while the copper matrix bypasses the surplus of this value equal to $I - I_c$, where I is the transport current in the strand. The strand components can be represented as a parallel connection of two resistors [6, p. 119-121]. By assuming a linear dependence of I_c on temperature, the critical current can be calculated as

$$I_c \approx I \cdot \frac{T - T_{cs}}{T_c - T_{cs}}, \tag{1.1}$$

where T – local temperature of the strand. If the local temperature T is assumed to be equal to the operating bath temperature T_0, one can deduce the current sharing temperature as

$$T_{cs} = T_0 + (T_c - T_0) \cdot (1 - \frac{I}{I_c\ T_0}). \tag{1.2}$$

Therefore, the current in a copper matrix I_{Cu} can be divided into three regimes as

$$\begin{cases} I_{Cu} = 0 & \text{for } T < T_{cs}, \\ I_{Cu} = I - I_c & \text{for } T_{cs} \leqslant T < T_c, \\ I_{Cu} = I & \text{for } T_c \leqslant T. \end{cases} \tag{1.3}$$

Superconductors have relatively high resistivity in their normal conducting state compared to traditional conductors such as copper. As described in Fig. 1.4, the resistivity of tin, being a superconductor below T_c, is higher at the normal conducting state with respect to copper [6, p. 1-6]. Therefore, it is assumed that the current only flows through the copper stabiliser above T_c and only this part of the strand contributes to the Joule heating.

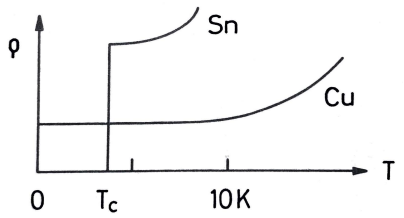

Figure 1.4: Resistivity of copper and tin at low temperatures [6].

In order to deduce the Joule effect in all operating regimes of the strand, one needs to

specify the composite fractions of the strand as

$$\begin{cases} r_{\mathrm{Cu/sc}} = \frac{f_{\mathrm{Cu}}}{f_{\mathrm{sc}}} = \frac{A_{\mathrm{Cu}}}{A_{\mathrm{sc}}} \\ \\ f_{\mathrm{Cu}} + f_{\mathrm{sc}} = 1, \end{cases} \tag{1.4}$$

where f_{Cu} – volumetric fraction of copper, f_{sc} – volumetric fraction of a superconductor, A_{Cu} – cross-sectional area of copper, A_{sc} – cross-sectional area of a superconductor. Based on (1.3) and (1.4), the heat source in the strand is formulated as

$$q_{\mathrm{Joule}} = J_{\mathrm{strand}}^2 \ \rho_{\mathrm{strand}} = J_{\mathrm{Cu}}^2 \ \rho_{\mathrm{Cu}} \ f_{\mathrm{Cu}} = \frac{I_{\mathrm{Cu}}^2}{f_{\mathrm{Cu}}^2 \ A_{\mathrm{strand}}^2} \ \rho_{\mathrm{Cu}} \ f_{\mathrm{Cu}} = \frac{I_{\mathrm{Cu}}^2}{A_{\mathrm{strand}}^2} \ \frac{\rho_{\mathrm{Cu}}}{f_{\mathrm{Cu}}}, \tag{1.5}$$

where J_{strand} – transport current density in a strand, ρ_{strand} – resistivity of a strand, J_{Cu} – current density in copper, ρ_{Cu} – resistivity of copper, I_{Cu} – current in a copper stabiliser, A_{strand} – cross-sectional area of a strand. As the unit of Equation (1.5) is [W/m^3], one can notice that the copper resistivity ρ_{Cu} should be divided by the non-superconductor fraction as

$$\rho_{\mathrm{strand}} = \frac{\rho_{\mathrm{Cu}}}{f_{\mathrm{Cu}}}. \tag{1.6}$$

This step is required if the strand is considered as a homogenised material with a heat source. Since only the copper stabiliser contributes to the Joule heating, the power density must be divided by the copper fraction f_{Cu}.

1.2.4 Minimum Propagation Zone

This section is based on [6, p. 124], and presents the analysis of a fully insulated strand in which the influence of helium in quench propagation is neglected. Furthermore, the following assumptions are made:

1. The strand is only made of a superconductor without a copper stabiliser.
2. The current density in the wire is close to its critical value.
3. The length L of a superconductor is heated from the bath temperature T_0 to the temperature T above the critical value T_{c}.

Since helium is not considered, the thermal disturbance in the strand leads to the quench propagation if the longitudinal heat evacuation is lower than the generation of heat in the quenched zone. This condition is formulated as

$$\rho J^2 A L \geqslant \frac{kA(T_{\mathrm{c}} - T_0)}{L}, \tag{1.7}$$

where ρ – resistivity of a superconductor, J – transport current density, A – wire cross-section, k – thermal conductivity of a superconductor. One can simply deduce the minimum propagating zone from the equation as [6, p. 124]

$$L_{\mathrm{MPZ}} = \sqrt{\frac{2k(T_{\mathrm{c}} - T_0)}{\rho J^2}}. \tag{1.8}$$

If the thermal disturbance is larger or equal to L_{MPZ}, the quench starts propagating along the wire. For pure Nb-Ti, the L_{MPZ} is approximately equal to $L = 1$ µm. Such a small value explains why a copper stabiliser is necessary in a composite strand. The usage of copper may increase L_{MPZ} by a factor of thousand. [6, p. 124]

1.3 Accelerator Magnets

1.3.1 Types and Functions

In the LHC, there are various accelerator magnets which can be divided into three main groups based on their function [2]:

1. Main dipole magnets. They bend the moving particles in order to keep them in the trajectory of a 27 km-long ring of the LHC. There are 1232 dipoles in the LHC.

2. Main quadrupole magnets. They constrain the width and the height of a particle beam in order to maintain it inside of the vacuum chamber within the LHC magnet. There are 392 such magnets inside the LHC.

3. Corrector magnets. One of their goals is to correct imperfections in the field quality created in the aforementioned magnets. Some of the examples of such magnets are: quadrupoles and skew quadrupoles, sextu-, octu-, deca- or dodecapoles.

Figure 1.5 presents the cross-section of the main LHC dipole. The particle beam is travelling inside of a beam pipe surrounded by a double-layer coil composed of multiple turns of a superconducting cable. The coils are clamped with pre-stressed steel collars in order to oppose the Lorentz forces acting on them during the machine operation. Since the magnets are mounted at room temperature, the pre-stress also serves for maintaining the elements position when the entire structure shrinks during the cooling process. In principle, the pre-stress maintains the contact between the coil and neighbouring elements which allows for avoiding friction and, thus, the heat generation. The collars are surrounded by a concentrically mounted iron yoke. The iron yoke increases the central magnetic field that directs the travelling beam.

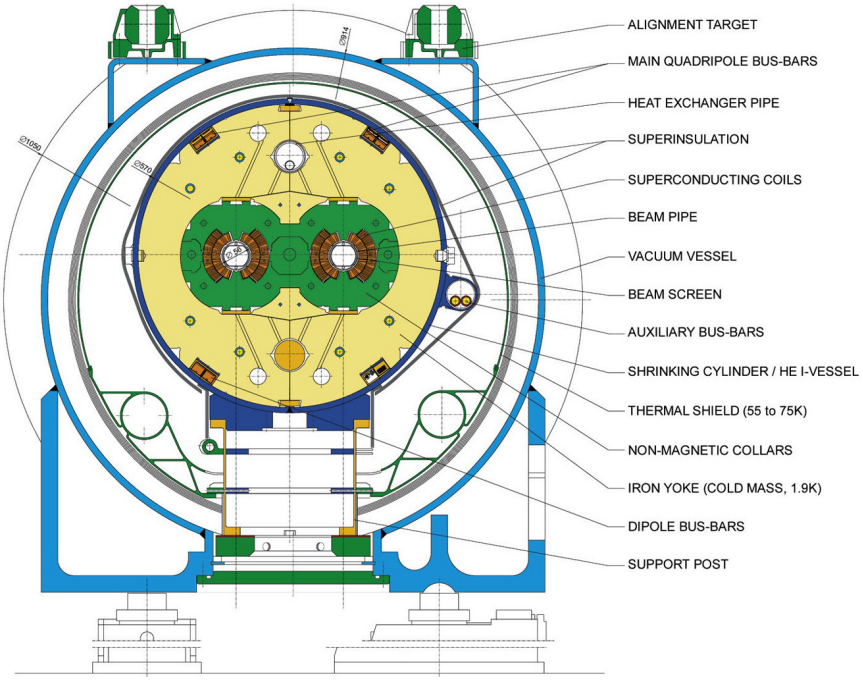

Figure 1.5: Cross-section of the LHC main dipole magnet with description of main components [8].

1.3.2 Quench Protection

When a quench occurs during the operation of a superconducting magnet, it must be detected in the shortest possible time. As soon as the quench is detected, the power converter, that supplies with current magnets connected in series, is switched off and energy, mostly magnetic, stored in the circuit is discharged. In principle, the energy should be either deposited in the coil or extracted from the magnet. The quench protection systems can be passive or active. The passive systems do not use any additional electronics which triggers the quench protection devices. An example of a passive protection system is a diode connected in parallel to a magnet. When the quench occurs, both the diode and the magnet are subjected to the same resistive voltage. If the resistive voltage reaches the forward diode voltage, the diode allows the current to bypass the magnet.

Among others, three active protection systems are described:

1. Quench heaters
2. Coupling-Loss Induced Quench (CLIQ) system
3. Energy extraction

Quench heaters are resistive strips installed along the coils in close contact with the windings. The heaters have an external power supply usually based on a charged capacitor. When a quench is detected, the heaters are fired. Due to the firing of the protection system, the quench heater spots, being in close contact with the windings, quench inside of a coil. This process supports the longitudinal quench propagation initiated at each of the quench heater spots. A larger resistive volume is created in the magnet and the current discharge occurs more quickly. The quench heaters have an external power supply independent of the superconducting coils. Therefore, they must be electrically insulated from the windings. In fact, the electrical insulation is a thermal barrier causing a delay between the moment when the heaters are fired and the time when the coil quenches. This delay is a characteristic feature of a quench heater system. The drawbacks of the quench heaters are as follows [9]:

1. On one hand, the quench heater system requires a high thermal diffusivity, i.e. the insulation between the heater and the coil should be thin enough to decrease the delay of the firing system. Meanwhile, the system should meet the requirements of an electrical insulator, which should be thick enough to prevent short-circuits. These two requirements are contradictory with respect to one another.

2. There is no possibility to replace quench heating strips of the magnet once they are installed. Therefore, in case a failure during the machine operation, the maintenance of this system is limited in a long-term.

In addition, there exists a natural effect that enables a magnet to discharge quicker, called a "quench back". According to the Faraday's law, the variation of a magnetic field in time induces eddy currents in an electrical conductor. The heat generated due to this phenomenon is referred as AC-losses. If the current drops sufficiently quickly, the AC-losses may induce a quench in the parts of a coil which remain superconducting. The CLIQ system is based on the same physical principle. It generates current oscillations in the superconducting windings to create a fast change of the magnetic field. According to [10], the CLIQ protection is more efficient with respect to the quench heaters in the operating conditions of a magnet close to the critical surface. CLIQ is implemented as a baseline for the inner triplets[6] for the HL-LHC project.

In the energy extraction system, a dump resistor is connected in series to a set of superconducting magnets in a circuit. When the quench is detected, an active switch allowing the current to bypass the dump resistor is disconnected. Thus, a part of the energy stored in the circuit is discharged in the resistor. This system allows for a faster recovery of a

[6]Inner triplets provide the final focusing of the proton beams at four collision points of the LHC [11].

magnet to its operating conditions because it requires less cooling power after the discharge process. [9]

A superconducting magnet can also be "self-protected" meaning that the energy accumulated in the circuit is discharged by a natural rise of resistivity inside of the magnet. This is the simplest and most cost-effective solution. The self-protectability is applicable in magnets characterised by relatively low current densities and, therefore, low magnetic fields. A relatively small amount of discharged energy in these magnets allows for limiting the temperature rise during the quench. In principle, the self-protectability should be the baseline for the magnet protection. However, if the energy density is too high, one should opt for an additional protection equipment.

1.3.3 Magnet Design Key Parameters

From the magnet design standpoint, the key parameters are:

1. peak voltage to ground
2. maximum hot-spot temperature

Temperature is rarely measured in experiments because there is no space to mount temperature sensors within the magnets. It is physically deduced from the resistive voltage measured across the magnet as a quench propagates. The place where a quench starts propagating is characterised by the highest temperature in the entire magnet during quench (also referred as the hot-spot temperature). The rise of resistive voltage inside the magnet must also remain within the allowable voltage-to-ground limits as well as turn-to-turn voltage limits imposed by electrical properties of the insulation. Exceeding these limits may lead to:

1. Short-circuit between different windings of the coil across the internal windings' insulation,
2. Short-circuit between the winding and the ground across the ground insulation of the magnet.

The hot-spot temperature and the voltage-to-ground value allow for defining whether the materials reach their safety limits at which they loose mechanical, thermal or electromagnetic properties leading to the magnet destruction. Therefore, there is a need to study the evolution of the key parameters in the process of a magnet design when the quench occurs in the considered system.

1.4 Numerical Analyses in Superconductors

1.4.1 Quench Modelling in Superconducting Magnets

A quench is a multi-physics problem including, among others, thermal, electric, and magnetic phenomena. In each of these phenomena, one copes with nonlinearities of material properties mainly due to a wide range of operating temperatures. The key design parameters required from a magnet are its sustainability at varying temperatures. The simulation challenges covered in this thesis are as follows:

1. Nonlinear heat capacity and thermal conductivity of magnet materials. The given material properties are a function of temperature and, in some cases, the magnetic field strength.

2. Nonlinear resistivity of copper being a function of the temperature and the magnetic field strength.

3. Nonlinear inductance of a superconducting magnet due to the influence of an iron yoke.

Apart from a multi-physics environment that requires a coupling of several simulations together, the complexity of a quench analysis is due to the following reasons:

1. Thermal and electrical time constants in transient analyses may vary by several orders of magnitude (multi-rate problem).

2. The simulation may consist of separate conceptual domains such as: circuit, magnet, etc. (multi-domain problem).

3. Analysis of components with extremely different sizes may be considered varying from a 10 µm-wide filament in a strand to a 10 m-long magnet as a whole (multi-scale problem).

As presented in Fig. 1.6, the magnet design is an iterative process in which four main studies ought to be conducted: electromagnetic, mechanical as well as those related to the circuit design and protection. The change of a design parameter in one element may entail the correction of a different one in another constituent. As an example, a study for the quench protection includes the verification to ensure the magnet does not overheat during the discharge. If the analysis concludes an error in the design, the information returns to electromagnetic studies, in which a new design of the coil has to be carried out. [12]

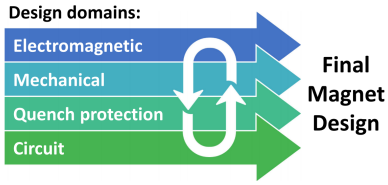

Figure 1.6: Domain considered during the design process of a superconducting magnet [12].

Figure 1.7 shows a thermal analysis workflow with respect to the problem complexity from a quench protection perspective. On one hand, the most accurate thermal map of a magnet is desired which allows for lowering the design safety coefficients. On the other hand, in order to obtain the most accurate solution, the number of degrees of freedom needs to be increased by applying a denser mesh or an additional dimension in the simulation. The higher the model completeness is, the longer the design iteration loops become. The modelling approach in a quench protection study is a compromise between the model completeness and the computation time. In fact, at an early design stage, the iteration loop should be short. However, for the final design one typically accepts a more detailed simulation.

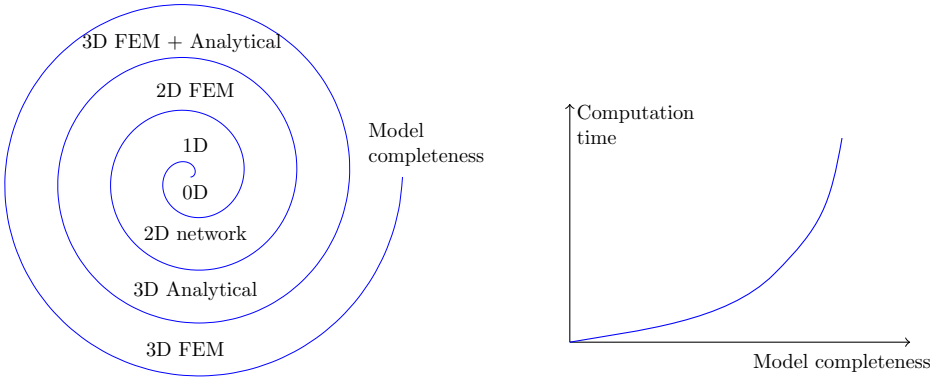

Figure 1.7: Left: thermal analysis workflow with respect to the problem complexity; Right: computation time vs. model completeness [13].

1.4.2 Overview of Numerical Methods

The availability of powerful computing machines, as well as widespread commercial and open source tools, allow for coping with engineering problems related to the magnet design numerically. Numerical models aim at solving ordinary and partial differential equations with prescribed initial and boundary conditions, by transforming them into a discrete set of algebraic equations. The solution of those algebraic equations leads to an approximate

result with respect to the initial PDE's [14]. The given process is summarised in Fig. 1.8.

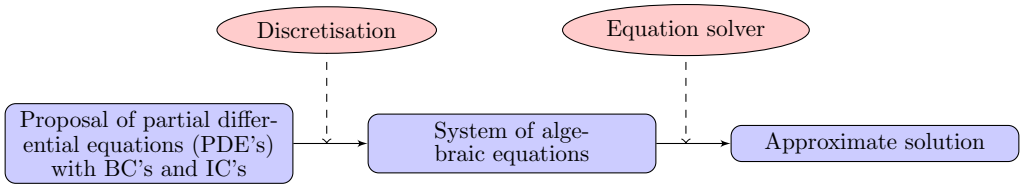

Figure 1.8: Schematic of the computational solution technique [14].

Numerical methods have several important advantages. First of all, they allow for finding an approximate solution of a real geometry, rather than an exact result of a simplified shape, being the case of analytical analyses. The coil of a magnet is composed of multiple materials including a superconductor, a stabiliser, and various layers of insulation. Taking the thermo-electric interactions between different materials in a relatively complex geometry into account makes the problem difficult or impossible to formulate analytically. Second of all, a numerical analysis serves for parameterising a given technical problem in engineering applications. Therefore, important magnet design factors, that influence operating conditions of a developed system, may be identified and improved. The numerical approach gives answers to so called "what if" questions within a fast iterative thinking process. There are various numerical methods that serve for a conversion of a PDE into a set of algebraic equations such as [14, 15]:

1. Finite difference method
2. Finite element method
3. Finite volume method
4. Spectral method

In this thesis, the finite element method (FEM) is employed. It is suitable for solving multi-physical problems, being the case of superconducting magnets in general. Many commercial numerical tools such as ANSYS or COMSOL use this method because of its capability to solve a large variety of physical problems. The finite element method related to the heat conduction problems, is discussed in Appendix B.

Chapter 2

Aim of Thesis

2.1 Research Objectives

Solving the 3D heat balance equation in a superconducting magnet, where quench is considered, is a challenging task. There are two primary reasons for that: (i) nonlinear material properties at cryogenic temperatures; (ii) high temperature gradients at the quench front. A numerical solver can handle the problem characterised as follows if the time step and mesh size are relatively small. Both implications serve for attaining the desired convergence in the solution. However, they make the simulation computationally demanding for a full-scale magnet. A good example of the level of nonlinearities at cryogenic temperatures is the thermal conductivity of copper. Its value changes from 250 to 1700 $\frac{\mathrm{W}}{\mathrm{m\,K}}$ for $T \in (1.9, 20)$ K at $B = 3$ T according to a fit provided by NIST[1] [16, p. 9-13].

To sum up, a simulation of the 3D quench propagation in a superconducting magnet requires high temporal and spatial resolution, which translates into considerable computation time. Since the 3D thermal models are computationally demanding, they are not suitable for a direct use during the the design of the magnet, which is an intrinsically iterative process. In this thesis, there are two scientific questions asked:

1. Can a multidimensional thermal analysis in superconducting accelerator magnets be efficient computationally?

2. Can a computationally efficient simulation for superconducting accelerator magnets be conducted in ANSYS?

ANSYS allows for creating multi-dimensional geometries being an important asset in magnet design. Moreover, the majority of mechanical studies in superconducting magnets

[1] NIST – National Institute of Standards and Technology.

are conducted in this software. ANSYS usage for the quench protection analyses would allow for unifying the tools in multiple design stages.

One can ask a question whether there are methods for approximating the quench problem which would allow for obtaining a quick and reliable solution. If the quench position, being a function of operating current and magnetic field strength, was known a priori, the numerical solver would solve the temperature distribution faster over the coil domain. In fact, such an approach has been already undertaken by ITER's Magnet Division to quench modelling of toroidal- and poloidal-field superconducting magnets in fusion applications [17]. In [18], ANSYS is used to solve 3D heat conduction equations simulating the thermal response of fusion magnets. The longitudinal quench propagation along the conductor is based on the analytical 1D model described in [19]. In this thesis, the following method called "quench velocity-based approach", is applied to study the quench discharge of the skew quadrupole.

2.2 Thesis Structure

The thesis is divided as follows. Chapter 1 is a general introduction to the quench analysis in superconducting magnets that allows the reader to smoothly acquaint with the problematics of the thesis. Chapter 2 depicts the main objectives of this work. Since at TE-MPE-PE section[2], COMSOL is used as the standard software for solving thermal quench problems, in Chapter 3, 1D quench propagation is solved in both ANSYS and COMSOL to cross-check the results. The analyses of a 1D strand are conducted with and without an insulation layer and resin. Chapter 4 explains in detail the quench velocity-based approach to solve multi-dimensional thermal problems in superconducting magnets. Chapter 5 describes the algorithms created to perform a multi-dimensional thermal analysis. Chapter 6 deals with how the foregoing algorithms are implemented in the simulations through Python scripts and as well as in ANSYS APDL scripts. Chapter 7 compares the quench velocity-based approach with standard analyses performed in Chapter 3. In Chapter 8, the quench velocity-based approach is applied to solve a multi-dimensional thermal problem of a skew quadrupole, being one of the high-order corrector magnets for the High-Luminosity LHC upgrade. The analysis is performed: (i) at constant current to simulate the quench propagation until the quench is detected, (ii) at varying current during the magnet discharge after the quench detection. The simulation results are compared with available measurements. Chapter 9 summarises the covered topics as well provides an answer to the research questions proposed in this thesis.

[2]TE-MPE-PE is an abbreviation for Performance Evaluation Section belonging to Machine Protection and Electrical Integrity Group being part of Technology Department at CERN.

Chapter 3

1D Quench Propagation Modelling

3.1 Motivation

At TE-MPE-PE section, the standard software to solve 1D heat propagation problems related to quench is COMSOL. In the 1st step of the quench studies, simplified ANSYS models are verified by means of STEAM-BBQ being a tool implemented in COMSOL and developed in the section [20]. The main aim of this cross-check is to verify the validity of implemented physics in ANSYS with respect to COMSOL. The presented studies are based on previously implemented 1D models in ANSYS where the influence of helium on the quench propagation is analysed [21]. A chapter of that work includes the comparison of COMSOL and ANSYS when an adiabatic 1D quench propagation is considered without an insulation layer [21] .

3.2 Assumptions

In this chapter, there are two studies conducted for a quench propagation in a 1D strand:

1. Analysis of a bare composite strand
2. Analysis of the composite strand with insulation and epoxy resin

For both cases, the mesh density study is conducted. The following assumptions are made in the simulations:

1. There is no helium cooling.
2. Temperature in the cross-section of a composite strand is uniform.

3. Turn-to-turn propagation does not occur between different windings of a coil.

4. When the insulation and epoxy resin are considered, the longitudinal heat transfer outside of the bare strand is neglected.

The second and third assumptions allow one to consider the quench simulation as a 1D longitudinal heat propagation. When the strand is analysed with an external insulation and epoxy resin, the analysis becomes a 1D+1D heat conduction problem because of the fourth assumption. It is further explained in Section 3.5. In presented thermal problems, the heat balance partial differential equation is solved as

$$\gamma c_p(T,B)\frac{\partial T}{\partial t} = \frac{\partial}{\partial \vec{r}}[k(T,B)\frac{\partial T}{\partial \vec{r}}] + q_0(T,B), \tag{3.1}$$

where T – temperature varying in time and space defined by a position vector $\vec{r}$, γ – mass density of a material, c_p – specific heat of a material, k – thermal conductivity of a material, q_v – external heat source (Joule heating). Specific heat capacity, thermal conductivity, and resistivity are a function of both temperature and magnetic field strength in the strand. The insulation of the strand and, optionally, resin are only a function of temperature. The domain is thermally anisotropic when the superconducting strand is analysed with an external insulation layer and, optionally, epoxy resin.

3.3 Reference Geometry - Skew Quadrupole

In this thesis, all the simulations are based on the geometry of a skew quadrupole which belongs to the group of high-order corrector magnets designed for the High-Luminosity LHC upgrade. The skew quadrupole is developed by the LASA laboratory of INFN-Milano. Its geometry is presented in Fig. 3.1. Each coil of the skew quadrupole (marked in red in the left picture) is fully impregnated and positioned in two mechanical supports (marked in grey). The entire magnet is surrounded by an iron yoke (marked in blue). One coil of a magnet out of four in total is presented in the right picture.

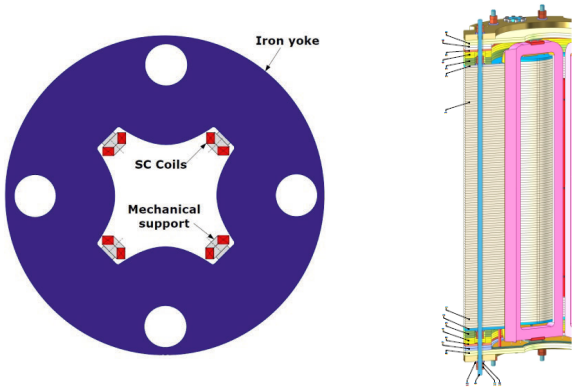

Figure 3.1: Left: cross-section of the skew quadrupole [22, p. 103-105]; right: one coil of the skew quadrupole [23].

As presented in Fig. 3.2, the coil consists of a composite strand marked in yellow made of Nb-Ti and a copper stabiliser. The strand is fully insulated with an S2-glass material marked in red. The cable is wound 754 times to form a coil with the total length of 812 m [24]. After the winding process, the coil is impregnated with an epoxy resin CTD-101K marked in blue in Fig. 3.2. The Rutherford cable is not applied in high-order corrector magnets. In the remainder of the thesis, the term "cable" is restricted to the composite strand made of a superconductor and a copper stabiliser including an external insulation layer and, optionally, a resin filler. The bare cable and winding are referred to as "strand". The one-dimensional coordinate system $\bar{x}$ in Fig. 3.2 represents the longitudinal direction of the strand. The 1D geometry is based on geometrical parameters of a single strand of the skew quadrupole whose simulations are further described in the next section.

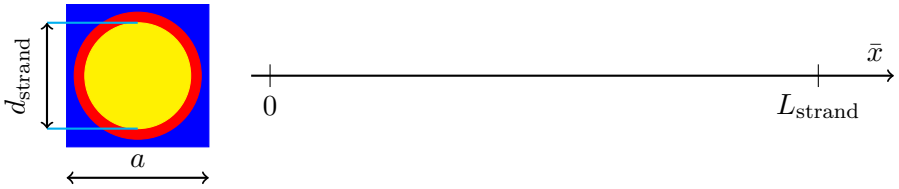

Figure 3.2: 1D strand geometry.

The parameters of the skew quadrupole presented in Table 3.1 do not change in the remainder of the thesis. The last two: (i) residual resistivity ratio, RRR, (ii) copper-to-superconductor ratio, $r_{\text{Cu/sc}}$ are obtained from the measurements of the prototype magnet performed at INFN. [23]

Table 3.1: Geometrical parameters of the skew quadrupole [22, 23].

parameter	value	unit
d_{strand}	0.7	[mm]
a	0.941	[mm]
$r_{\text{Cu/sc}}$	2.2	[-]
RRR	193	[-]

3.4 1D Analysis of a Strand

This analysis considers the case of a strand, i.e. when neither insulation nor epoxy resin are considered. Therefore, the strand cross-sectional area is the yellow part of the domain presented in Fig. 3.2. The strand length is equal to $L_{\text{strand}} = 1$ m.

3.4.1 Geometry, Material Properties and Mesh

Since copper and Nb-Ti are characterised by a different volumetric heat capacity, the total volumetric heat capacity of a strand is given by

$$c_{\text{v, strand}} = f_{\text{Cu}}\ c_{\text{v, Cu}}(T) + f_{\text{Nb-Ti}}\ c_{\text{v, Nb-Ti}}(T, B), \tag{3.2}$$

where $c_{\text{v, Cu}}$ – volumetric heat capacity of copper as a function of temperature, $c_{\text{v, Nb-Ti}}$ – volumetric heat capacity of Nb-Ti as a function of temperature and magnetic field strength, f_{Cu} – volumetric fraction of copper, $f_{\text{Nb-Ti}}$ – volumetric fraction of Nb-Ti. The volumetric heat capacity of the composite strand for a given value of B and $r_{\text{Cu/sc}}$ is shown in Fig. 3.3.

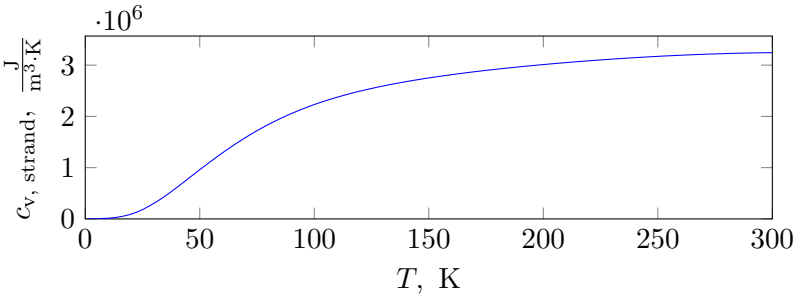

Figure 3.3: Strand volumetric heat capacity as a function of temperature for $B = 2$ T and $r_{\text{Cu/sc}} = 2.2$.

According to [25, p. 46], the thermal conductivity of Nb-Ti changes from 1 to 9 $\frac{\text{W}}{\text{m K}}$ in the temperature range of $T \in (20, 200)$ K. At temperatures below 20 K, the thermal

conductivity of Nb-Ti is lower than 1 $\frac{\mathrm{W}}{\mathrm{m\,K}}$. By comparing the thermal conductivity of Nb-Ti and copper, whose thermal conductivity is presented in Appendix A.1.3, one can conclude that the thermal conductivity of Nb-Ti is negligible and only the part of copper contributes to the longitudinal heat propagation, as

$$k_{\text{strand}} = f_{\text{Cu}}\ k_{\text{Cu}} + f_{\text{Nb-Ti}}\ k_{\text{Nb-Ti}} \approx f_{\text{Cu}}\ k_{\text{Cu}}, \tag{3.3}$$

where k_{strand} – thermal conductivity of a strand, k_{Cu} – thermal conductivity of copper, $k_{\text{Nb-Ti}}$ – thermal conductivity of Nb-Ti. It is important to highlight that only in Chapter 3, the thermal conductivity of copper is calculated according to the Wiedemann-Franz formula, as

$$k_{\text{Cu}} = 2.45 \cdot 10^{-8}\ \frac{T}{\rho_{\text{Cu}}}. \tag{3.4}$$

Wiedeman-Franz formula is used for the sake of comparison of the results with COMSOL in which only such material property is available. In the remainder of the thesis, the thermal conductivity of copper and all other material properties are calculated according to the fits provided by NIST, as described in Appendix A.

In ANSYS, the element LINK33 is used to the 1D quench propagation. It is a uniaxial 2-node linear element with the ability to conduct heat between its nodes suitable for steady-state and transient analyses [26]. Unlike ANSYS, COMSOL does not use explicit element types to define physical equations in a domain. Instead, one has to choose from a list of available physics modules, in this case thermal physics. Therefore, the algebraic physical equations related to heat conduction in COMSOL are applied to the nodes belonging to the composite strand. In both tools, a uniformly distributed mesh is used with mesh size equal to 0.1 mm. The mesh size is based on [21, p. 40] in which it is recommended to discretise the domain in the scale less than or equal to 1 mm for an adiabatic analysis of a 1D quench propagation.

3.4.2 Initial and Boundary Conditions, and Solver Settings

Over the entire length of the strand, the power density is applied according to (1.5). In addition, a Gaussian profile of the initial temperature is assumed according to

$$T(x) = T_{\text{init}} + (T_{\text{max}} - T_{\text{init}})\ e^{-(\frac{x}{\alpha})^2}, \tag{3.5}$$

where $T(x)$ – temperature profile along x-axis, T_{init} – initial bath temperature of a strand, T_{max} – the maximum temperature of the Gaussian profile, α – shape parameter of the Gaussian profile proportional to the standard deviation. The input parameters corresponding to

the Gaussian profile, initial transport current, and initially quenched zone are depicted in Table 3.2.

Table 3.2: Input parameters for the 1D analysis.

parameter	value	unit
I	100	[A]
B	2	[T]
T_{init}	1.9	[K]
T_{max}	20.0	[K]
T_{c}	8.429	[K]
$L_{\text{quench, init}}$	0.1	[m]
α	0.223	[m]

The initially quenched zone is equal to $L_{\text{quench, init}} = 0.1$ m when symmetry is not taken into consideration. It means that at $x = 0.05$ m, the strand is at the critical temperature for the given magnetic field strength. The shape parameter, α in (3.5) is calculated accordingly. The critical temperature T_{c} is calculated as in (A.15) in Appendix A.2. A symmetry condition is applied at the position $x = 0$ m. Thus, a one metre-long strand represents a half of the analysed domain, as presented in Fig. 3.4.

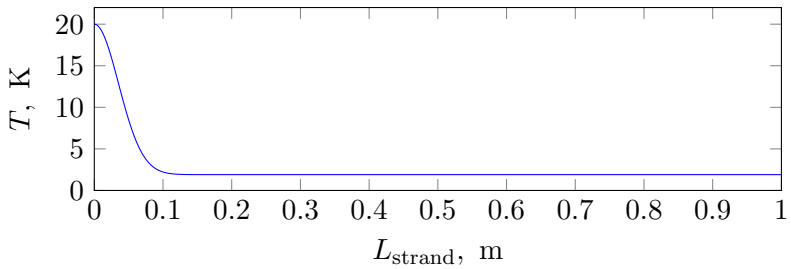

Figure 3.4: Initial Gaussian temperature distribution.

The solution in both ANSYS and COMSOL are obtained with a strict time stepping, i.e. the time step during the solution is fixed. The input parameters related to the time stepping algorithm and the total simulation time are presented in Table 3.3. ANSYS uses a default sparse solver with an automatic solution control setting. In COMSOL, the Multifrontal Massively Parallel Sparse Direct Solver (MUMPS) is applied. Both tools use direct solvers with an implicit Backward Euler time stepping method [27, 28]. In a time-dependent domain, default values for time stepping convergence are applied in both ANSYS and COMSOL. The geometry is built based on the 1$^{\text{st}}$ order elements in each tool.

Table 3.3: Analysis time stepping input parameters.

parameter	value	unit
t_{total}	10^{-1}	[s]
$t_{\text{step, max}}$	10^{-4}	[s]

3.4.3 Results

The temperature distribution profiles are compared at three time steps $t = \{0.03, 0.06, 0.1\}$ s, as presented in Fig. 3.5. One can conclude that the initial heat pulse is sufficient to initiate the quench. In fact, relatively high thermal conductivity and low heat capacity of the strand allow for a longitudinal quench propagation. At each time window, the hot-spot is placed at $x = 0$ m. In general, COMSOL predicts faster quench propagation.

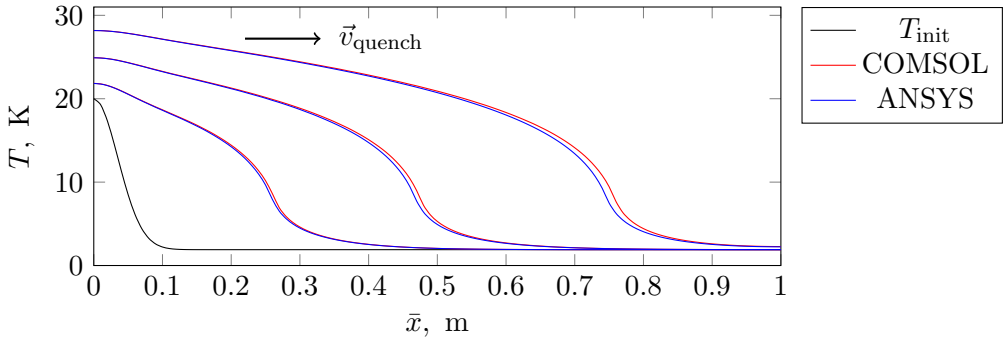

Figure 3.5: Temperature distribution calculated in COMSOL and ANSYS for three time steps: $t = \{0.03, 0.06, 0.1\}$ s with a specified direction of the quench propagation, $\vec{v}_{\text{quench}}$.

The quench velocity is calculated by comparing the position of the quench front at $t = 0.06$ s and $t = 0.1$ s, assuming that at this moment, the quench propagation velocity reaches a steady-state value. The relative error is calculated as

$$E_{\text{r}} = \frac{r_{\text{ANSYS}} - r_{\text{COMSOL}}}{r_{\text{COMSOL}}} \, 100\%, \tag{3.6}$$

where r_{ANSYS} – a given result from ANSYS, r_{COMSOL} – a given result from COMSOL. The relative error is estimated for:

- quench velocity, as presented in Table 3.4,
- temperature along the strand length at $t = 0.1$ s, as presented in Fig. 3.6.

Table 3.4: Quench velocity comparison between COMSOL and ANSYS.

parameter	value	unit
$v_{\text{quench, COMSOL}}$	7.075	[m/s]
$v_{\text{quench, ANSYS}}$	6.968	[m/s]
E_{r}	-1.519	[%]

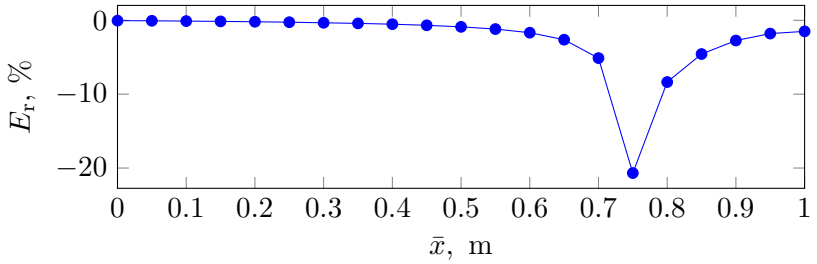

Figure 3.6: Relative error along the strand with respect to the temperature distribution for $t = 0.1$ s.

The difference in the hot-spot temperature at $x = 0$ m does not exceed 0.06 %. The quench velocity in ANSYS is lower than the one in COMSOL by less than 2 % which results in the increase of the relative error corresponding to the nodal temperature distribution up to 20 % close to the quench front at $t = 0.1$ s.

3.5 1D Analysis with Insulation and Epoxy Resin

In this section, the 1D coil is studied with an insulation layer together with the epoxy resin. Therefore, the cross-sectional area of the considered geometry consists of a strand, insulation and epoxy resin marked in yellow, red and blue, respectively; see Fig. 3.2. The length of the considered domain is $L_{\text{strand}} = 1$ m.

3.5.1 Geometry, Material Properties and Mesh

Due to the lack of data about thermal material properties of CTD-101K and S2-glass at the cryogenic temperatures, it is assumed in this thesis that the region outside of the composite strand is homogenised and represented by G10. G10 is a material widely known and applied in the superconducting magnets design community. Its material properties are presented in Appendix A.3.1.

The longitudinal heat transfer outside of the composite strand is neglected with respect to the transversal one. In order to demonstrate the validity of this assumption, the thermal

diffusivity of a composite strand is compared with the one of G10. The ratio of both thermal diffusivities is calculated as

$$\frac{\alpha_{\text{strand}}}{\alpha_{\text{G10}}} = \frac{k_{\text{strand}}}{k_{\text{G10}}} \frac{c_{\text{v, G10}}}{c_{\text{v, strand}}}, \tag{3.7}$$

where α_{strand} – thermal diffusivity of a strand, α_{G10} – thermal diffusivity of G10, k_{G10} – thermal conductivity of G10, $c_{\text{v, G10}}$ – volumetric heat capacity of G10.

As presented in Fig. 3.7, the longitudinal thermal diffusivity of the strand is in the range of between 2 and 5 orders of magnitude higher than the one of G10. The ratio stabilises at $T > 100$ K and is approximately equal to 10^2. Because of such a high difference in the thermal diffusivity of the strand and G10, it is assumed that the longitudinal thermal diffusivity of G10 is neglected. Therefore, the insulation together with resin are modelled as 1D elements placed transversely to the strand elements [20].

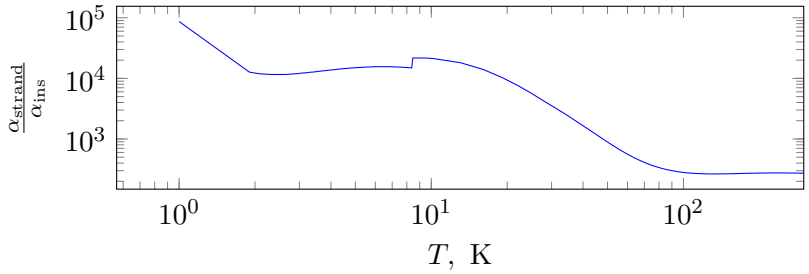

Figure 3.7: Strand equivalent thermal diffusivity to insulation thermal diffusivity ratio for $B = 2$ T and $r_{\text{Cu/sc}}$.

With specified assumptions, the 3D strand domain can be transformed into a 1D+1D domain including the strand as well as the insulation together with resin, as presented in Fig. 3.8. Each of the 1D insulation domains orthogonal to the 1D strand is characterised by an equivalent length, L_{ins} and a conduction area, $A_{\text{ins,cond}}$. Both parameters should be calculated so that the volume of the insulation and resin remains equal in two cases: 1D+1D and 3D. As a result, the volumetric heat capacity of every component of the cable is identical.

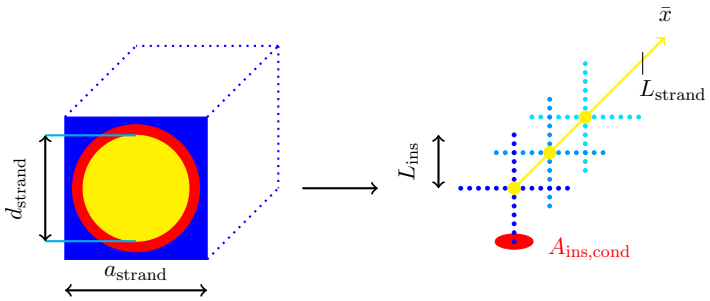

Figure 3.8: 3-dimensional transformation of a 1D+1D strand domain in ANSYS.

In order to define the geometric parameters for the insulation layer with resin, one should calculate the insulation cross-sectional area as

$$A_{\text{ins}} = a_{\text{strand}}^2 - \frac{\pi d_{\text{strand}}^2}{4}, \tag{3.8}$$

where a_{strand} – strand side, d_{strand} – strand diameter. Then, the average insulation perimeter is estimated as

$$p_{\text{avg}} = \frac{4a_{\text{strand}} + \pi d_{\text{strand}}}{2}. \tag{3.9}$$

By combining the formulae (3.8) and (3.9), one can calculate the equivalent insulation length with resin as

$$L_{\text{ins}} = \frac{A_{\text{ins}}}{p_{\text{avg}}}. \tag{3.10}$$

Calculating the equivalent insulation length with resin in such a manner allows for using this approximation also in case both the strand and the insulation are of the same shapes, e.g. if they are rectangles or circles. The conduction area of transverse elements is calculated as

$$A_{\text{ins, cond}} = \frac{1}{4} \frac{A_{\text{ins}}\ L_{\text{strand}}}{L_{\text{ins}}} \frac{1}{n_{\text{divisions, strand}}}, \tag{3.11}$$

where V_{ins} – total insulation volume with resin, $n_{\text{divisions, strand}}$ – number of divisions applied along the 1D strand domain. One should remember that $A_{\text{ins, cond}}$ of the insulation elements placed at two ends of the strand should be divided by two. [29]

3.5.2 Initial and Boundary Conditions, and Solver Settings

The strand geometry, the initial temperature conditions, and the analysis settings for both COMSOL and ANSYS remain the identical with respect to the simulation described in Section 3.4, as presented in Fig. 3.4 and Table 3.2, respectively. The analysis parameters are presented in Table 3.3. The solver settings for both tools remain identical as well. The

additional parameters corresponding to the insulation are described in Table 3.5.

Table 3.5: Insulation input parameters.

parameter	value	unit
L_{ins}	0.1679	[mm]
mesh size (insulation)	28	[µm]

Every orthogonal insulation domain consists of six elements of the same length. It is important to mention that in COMSOL, six orthogonal elements do not represent a quarter, but a full cross-section. Therefore, the number of insulation elements in COMSOL is four times smaller and $A_{\text{ins, cond}}$ – four times larger with respect to the analysis performed in ANSYS.

3.5.3 Results

Similarly to Section 3.4.3, the results are compared at three time steps, as presented in Fig. 3.9. A slower quench propagation over time can be observed when the insulation is considered. At each time window, the hot-spot is placed at $x = 0$ m.

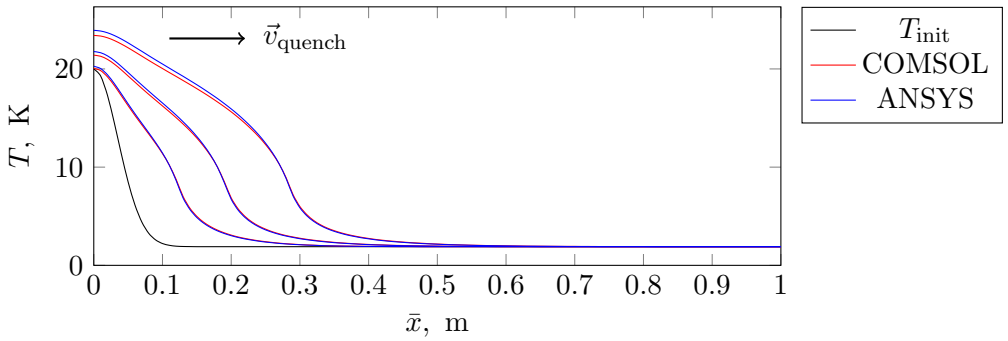

Figure 3.9: Temperature distribution calculated in COMSOL and ANSYS for three time steps: $t = \{0.03, 0.06, 0.1\}$ s with a specified direction of the quench propagation, $\vec{v}_{\text{quench}}$.

The quench velocity is calculated by comparing the position of the quench front at $t = 0.06$ s and $t = 0.1$ s. The relative error is calculated as in (3.6) and is estimated for:

1. Quench velocity (see Table 3.6)
2. Temperature along the strand length at $t = 0.1$ s (see Fig. 3.10)

Table 3.6: Quench velocity comparison between COMSOL and ANSYS.

parameter	value	unit
$v_{\text{quench, COMSOL}}$	2.308	[m/s]
$v_{\text{quench, ANSYS}}$	2.310	[m/s]
E_{r}	0.108	[%]

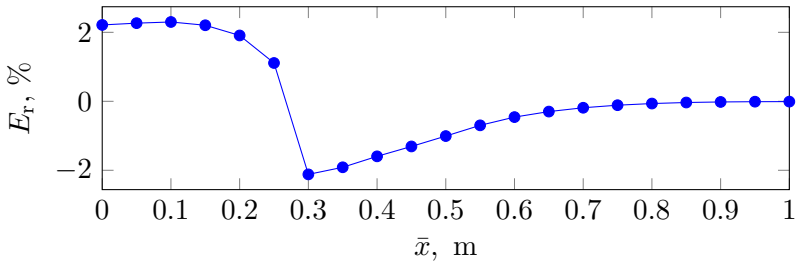

Figure 3.10: Relative error along the strand with respect to the temperature distribution for $t = 0.1$ s.

The difference in quench velocity estimation in both models is less than 1%, as presented in Table 3.6. As shown in Fig. 3.9, the relative error oscillates in the range of +2% at the hot-spot to -2% at the quench front. It means that ANSYS overestimates the hot-spot temperature and underestimates the temperature at the quench front with respect to COMSOL.

3.6 Conclusion

The comparison of the quench velocity and the hot-spot temperature is presented in Table 3.7. The maximum relative error is no larger than 2% in the analyses with insulation and epoxy resin, and 2.5% in the simulations of a bare strand. In both ANSYS and COMSOL, a mesh density study is conducted to verify its influence on the variation of the nodal temperature along the strand. The analysis indicates that a relative error of the nodal temperature distribution is lower than 0.5% when the mesh 10 times coarser is used. The remaining error between COMSOL and ANSYS is a consequence of the convergence discrepancy. Moreover, both of these computing tools are commercial in which the access to various parameters related to the solver settings is either limited or difficult to compare. Despite the error, it is concluded that the ANSYS model is verified with respect to STEAM-BBQ tool.

Table 3.7: Comparison of the quench velocity and the hot-spot temperature between COMSOL and ANSYS.

	bare strand		strand with insulation and resin	
	v_{quench}, $\frac{\text{m}}{\text{s}}$	$T_{\text{hot-spot}}$ at $t = 0.1$ s, K	v_{quench}, $\frac{\text{m}}{\text{s}}$	$T_{\text{hot-spot}}$ at $t = 0.1$ s, K
COMSOL	7.075	28.188	2.308	23.408
ANSYS	6.968	28.174	2.310	23.926
E_{r}, %	-1.519	-0.050	0.108	2.213

As advised in [21, p. 40] the mesh size of 1 mm should be used along the strand to simulate the 1D adiabatic quench propagation. If the insulation layer is considered, the total number of nodes is estimated as

$$n_{\text{nodes, total}} = n_{\text{nodes, strand}} \ (1 + 4 \ n_{\text{nodes, insulation}}), \tag{3.12}$$

where $n_{\text{nodes, strand}}$ – total number of nodes along the strand, $n_{\text{nodes, insulation}}$ – number of nodes across the insulation layer. With 6 nodes across the insulation, the total number of nodes for an 812 m-long cable of the skew quadrupole accounts for approximately 20 million nodes. Taking into account that this is not even a fully 3D problem, the number is relatively large. As presented in Fig. 3.11, the computing time is estimated for the analysis with the mesh size of 1 mm and 6 elements across the insulation. Three analyses are conducted with a varying length of the strand. The analysis of a 10 metre-long domain lasts more than 7 hours. The linear interpolation of the results indicates that, the simulation of a 100 m-long cable would approximately last 72 hours[1].

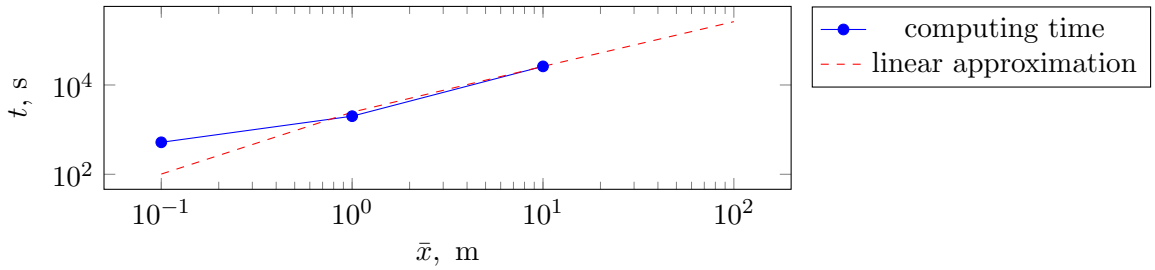

Figure 3.11: Computing time as a function of number of nodes.

One of the methods for reducing the discretised domain is the application of an adaptive

[1]The analysis was performed on the following calculation unit: Intel(R) Xeon(R) CPU E5-2667 V4 @ 3.20 GHz (2 processors) with RAM 128 GB.

mesh which locally refines space remaining in a close contact with the moving quench front. Such a solution would be applicable if a 1D analysis was taken into consideration. The aim of this work is to conduct a multidimensional analysis with both longitudinal and transverse quench propagation. This requires a 3D adaptive mesh refinement, which would be very difficult to perform.

Chapter 4

Quench Velocity-Based Approach

4.1 Concept

When the quench is generated, the point at which the cable transitions from the superconducting to the normal conducting state propagates over time, as shown in Fig. 4.1. This point remains at the critical temperature, T_c for a given magnetic field strength and current density. The time derivative of the quench front position is referred to as quench velocity. At fixed current and magnetic field strength, the quench velocity remains constant.

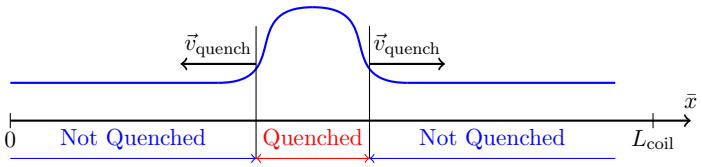

Figure 4.1: Schematic of dividing the coil length into the quenched and non-quenched zone.

Quench velocity can be approximated with analytic formulae. They are usually a function of current density, composite strand material properties and temperature gradient around the quench front. For superconducting accelerator magnets, Wilson's formula is often applied to estimate the quench velocity if cooling with liquid helium is not considered [30, p. 206]. In principle, it is possible to estimate the quench velocity in three different manners. It can be:

1. based on available measurements;
2. calculated analytically based on available formulae; or
3. calculated numerically based on a short numerical model with refined mesh to solve the quench front with sufficient precision (ideally validated against the measurements).

4.2 Problematics

While solving the heat balance equation related to the quench propagation, the solver must handle high nonlinearities of material properties at cryogenic temperatures. In a problem of this kind, the material properties are locally linearised. The solver iterates over the temperature distribution in space until the change of temperature between two consecutive iterations converges to a desirably low value. In order to accurately solve the temperature distribution in the entire region of the cable, one should locally refine mesh in the occurrence of a high temperature difference and/or decrease the simulation time step (up to the order of a µs).

Figure 4.2 presents an approximated temperature distribution over a discretised 1D thermal domain. The quenched and non-quenched regions in a superconducting cable have different, but relatively uniform, temperature distribution except for the regions close to the quench front. Therefore, in a general analysis of the quench propagation, the mesh density relatively far from the quench front can be relaxed due to the lower temperature difference between the nodes.

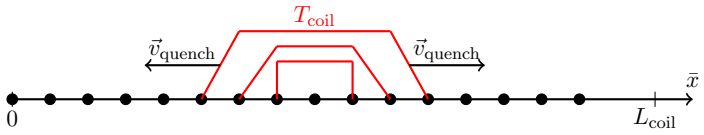

Figure 4.2: Schematic of the temperature propagation with quench velocity modelling.

While solving the quench propagation by means of the quench velocity-based approach, a constant quench velocity along a given winding (subject to local magnetic field) is assumed. The quench velocity is integrated over time in an external routine and assigned to a thermal model as a quench front position. As a result of providing the quench position over time, the mesh refinement along the coil (also at the quench front) is no longer required and the numerical model with less degrees of freedom is solved. In other words, the numerical model still calculates the heat balance equation, but with a fixed and relatively coarse mesh along the entire strand. The temperature distribution close to the quench front is approximated, knowing that it is placed between two extreme temperatures (either inside or outside of the quenched zone).

4.3 Co-simulation of Quench Velocity Calculation Routine with Numerical Thermal Solver

In order to distinguish a quenched zone from a non-quenched one, the superconducting cable is divided into two domains as presented in Fig. 4.3. The non-quenched part is characterised by nearly zero-resistivity (not zero for the sake of numerical stability) whereas the quenched cable has nonlinear resistivity of a strand composite as described in (1.6) [31]. The nonlinear resistivity is reassigned as the quench propagates in the numerical model.

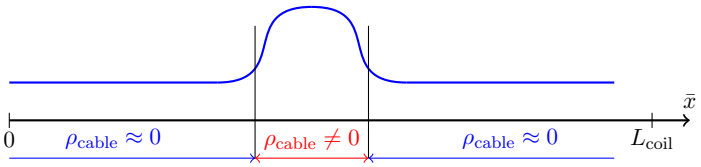

Figure 4.3: Material properties assignment in ANSYS depending on the quenched and non-quenched zone [31].

The reassignment of material properties is handled by an external routine which controls the quench propagation in time. In this case, a thermal model (solved by the numerical model) and quench velocity estimator (calculated by the external routine) are co-simulated. This problem can be resolved by means of:

1. one-directional exchange of signals;
2. bi-directional exchange of signals.

The one-directional exchange of signals is presented in Fig. 4.4. In this case, the numerical solver is provided with a quench front position from the previous communication point t_{j-1} and calculates the temperature distribution in the current communication point t_j. The parameter denoted as x_{j-1} represents the information about the value of the next communication point t_j and the quench front position in the cable. The external routine assumes a constant quench velocity independent of current and magnetic field strength. Therefore, the routine is not updated.

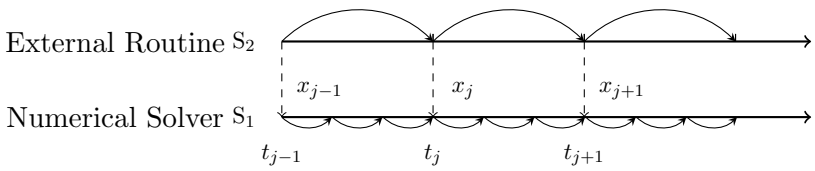

Figure 4.4: Schematic representation of one-directional coupling between a numerical solver and the quench velocity estimator.

The bi-directional exchange of signals is presented in Fig. 4.5. Additionally to the one-directional case, the external routine is updated with the value of current and magnetic field strength from the numerical solver denoted as y_j at communication point t_j to estimate the quench velocity at t_{j-1}.

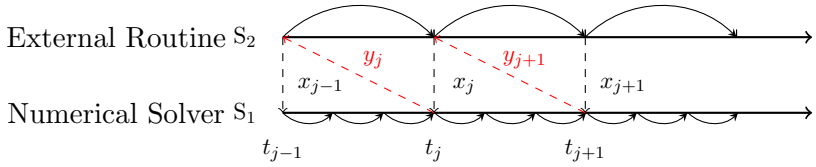

Figure 4.5: Schematic representation of bi-directional coupling between a numerical solver and the quench velocity estimator.

In both cases, the initial quench length is assumed at the beginning of the co-simulation. Between the time steps of the external routine, the numerical solver handles the problem with an adaptive time step smaller than the communication points t_j.

Provided that the data exchange between an external routine and a numerical solver occurs at communication times t_j, the quench velocity assignment algorithm is solved as described in Algorithm 1.

Algorithm 1 Quench velocity assignment algorithm.

1: assume initial quench position x_{j-1}
2: **for** $j = 1, 2, ..., N$ **do**
3: solve temperature distribution at time t_j
4: send solver results y_j to external routine time t_{j-1} (only for a bi-directional case)
5: calculate new quench position x_j for a quench front
6: assign quench position x_j to new nodes for solver time t_j

To sum up, the quench velocity assignment algorithm only requires a one-directional exchange of signals at fixed current and magnetic field strength. If the quench simulation is conducted at varying values of current and/or magnetic field, the bi-directional exchange of signals must be implemented. In this thesis, the quench velocity is estimated numerically. In the following chapters, the communication times t_j between ANSYS and the external routine are described as t_{com}.

Chapter 5

Models and Algorithms

There are two important phenomena in thermal quench simulations: (*i*) longitudinal quench propagation, (*ii*) transverse heat flow through the insulation layer that causes a quench in neighbouring windings. The adopted quench velocity-based approach only allows for estimating the longitudinal quench propagation corresponding to the 1^{st} case. Actually, in general, one could also rely on pre-calculated transverse quench propagation velocity. However, as the number of nodes across the insulation is relatively small, this solution is not chosen. The heat flow across the insulation layer is handled by means of a standard heat balance equation. In order to conduct a multi-strand thermo-electric analysis with the insulation layer using the quench velocity-based approach, the ANSYS models are developed which allow for studying the following physics:

1. Calculation of the longitudinal quench propagation with a quench front location based on a priori known quench velocity map
2. Calculation of a transverse heat propagation across the insulation
3. Calculation of the current during the discharge in a magnet
4. Calculation of the magnetic field strength across the coil based on a plane geometry of a magnet

The multi-strand thermo-electric model in ANSYS covers the first two points. In the third point, the electrical circuit is solved in which the multi-strand coil is represented by the resistance varying over time. The last point corresponds to a separate magnetic model prepared in ANSYS to simulate the magnetic strength distribution for a varying value of current. This model is required to include the dependence of the material properties on

the magnetic field strength. In addition, the following algorithms are implemented in the external routine in order to meet the requirements of the quench velocity-based approach:

1. Quench velocity assignment algorithm. It calculates a time-dependent quench position based on a quench velocity function.
2. Multidimensional mapping algorithm. It maps multi-dimensional magnet geometries onto a one-dimensional cable.
3. Quench detection algorithm. It detects new quenches in a multi-strand domain if a transverse heat propagation results in a quench of windings outside of the already existing quenched zone.
4. Quench front position assignment algorithm. It assigns nodes in a discretised and time-dependent space.

This chapter presents the implementation of the electro-thermal model in ANSYS. The description of the circuit and the magnetic field analyses will be further described based on the analysis of the skew quadrupole in Chapter 8. In addition, all algorithms are also discussed in this chapter except for the quench velocity assignment algorithm already depicted in Section 4.3.

5.1 Model Preparation in ANSYS APDL

5.1.1 Geometry

Creating a 3D magnet geometry within a CAD tool in which a single cable is wound n times is a complicated task, because the model must take into account a relative position of one winding with respect to another. Therefore, every winding is considered to be a separate domain in prepared ANSYS models with winding jumps, as shown in Fig. 5.1. The green lines depict the electro-thermal coupling of windings according to their winding order. With such an approach, one can easily create multi-strand magnet geometries. Moreover, by specifying which windings are coupled together, with the same numerical domain, one can also analyse magnets with a different winding scheme.

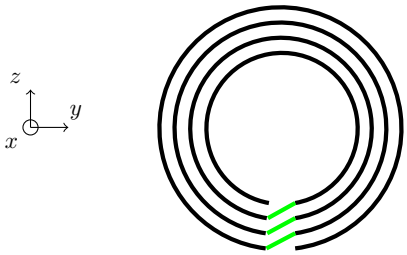

Figure 5.1: Domain representation with multiple windings.

Every winding can have an external homogenised insulation layer represented as a transverse one-dimensional element, as shown in Fig. 5.2. If multiple windings are created, the end nodes of each insulation layer are thermally coupled with the neighbouring insulation elements. The model assumes no diagonal heat propagation across the insulation layer, i.e. the windings are not diagonally coupled. The insulation elements marked in blue are represented by the LINK33 element whereas the composite strand in yellow – LINK68. LINK68 is a uniaxial 1D element which can be used in a 3D space. It has the ability to conduct heat and electrical current along its nodes with an internal heat source corresponding to the Joule heating. LINK68 is used for steady-state and transient simulations. In standard ANSYS simulations, the Joule heating is implemented by introducing a power source over the entire strand domain as a function of resistivity (being a function of temperature). The usage of LINK68 in the quench velocity-based approach allows to omit applying the external power source since the solver computes the heat source by an internal routine based on the material property state. Thus, the element requires electrical resistivity of the composite strand [26].

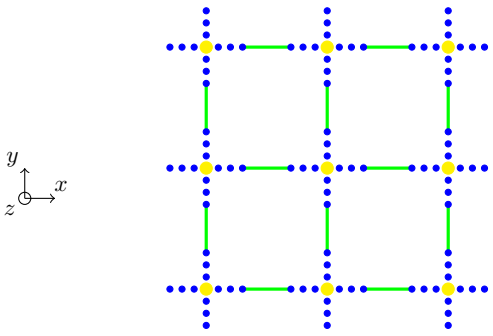

Figure 5.2: Cross-section of a 1D+1D+1D geometry with coupled insulation nodes marked in green.

5.1.2 Mesh Generation

The mesh generation scheme for the magnet geometry is illustrated in Fig. 5.3. The mesh should be distributed over so called "nodal planes", which are always perpendicular to the normal vector $\vec{n}$ of a directional spline. The directional spline is a curve that represents a final shape of the coil. One nodal plane with nine windings is presented in Fig. 5.2. In order to generate mesh, a longitudinal number of divisions has to be specified. When the insulation is considered, the number of divisions across the insulation must be defined as well. Each insulation element is characterised by the same volume.

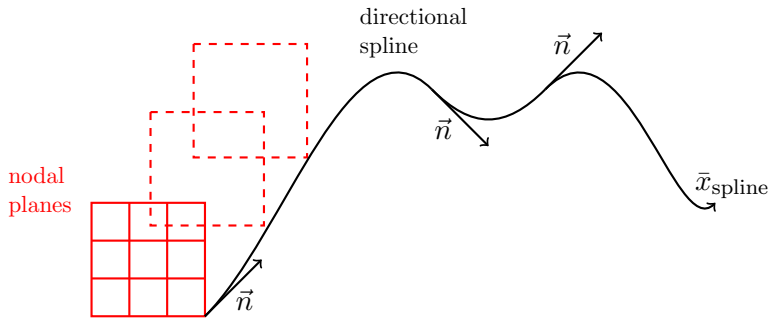

Figure 5.3: Multidimensional mesh generation.

5.2 Multidimensional Mapping Algorithm

The multidimensional mapping algorithm aims to translate a multidimensional geometry into an imaginary 1D strand. This procedure serves for (i) assigning the magnetic field strength to the windings separately, (ii) mapping the geometry for the quench detection algorithm.

As illustrated in Fig. 5.4, a multi-strand domain is subjected to a time- and space-dependent magnetic field. Thus, every winding is exposed to a different magnetic field which also varies with a current change. From the magnetic analysis standpoint, it is assumed that the coil is infinitely long, which allows for using a 2D magnetic map from the centre of a magnet cross-section.

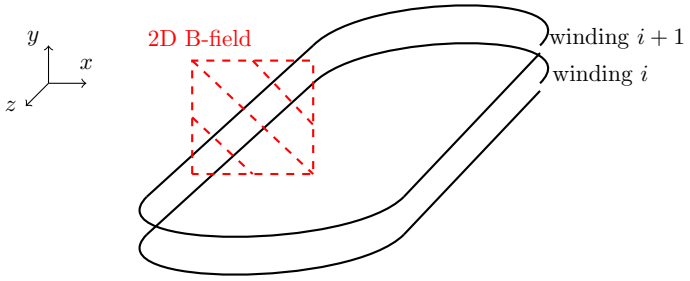

Figure 5.4: Schematic of an assignment of a magnetic field strength to a multi-strand domain.

The magnetic field strength is assigned to every winding separately, as shown in Fig. 5.5. Each of them is characterised by a different thermal and electrical material property. The multi-strand geometry is translated into a realistic one-dimensional cable with a length L_{coil} equal to the total coil length. Each part of the coil corresponding to one winding, is subject to a different magnetic field strength B_{n}. Similarly to Fig. 3.2, the one-dimensional coordinate system $\bar{x}$ in Fig. 5.5 represents the longitudinal direction of the coil.

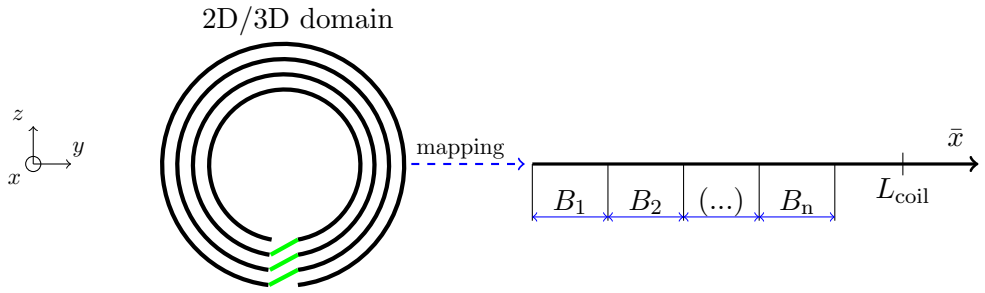

Figure 5.5: Mapping scheme of a magnetic field strength onto an imaginary 1D coil.

5.3 Quench Detection Algorithm

When a multi-strand thermal quench problem is analysed, a quenched zone belonging to one winding heats up the neighbouring ones across the insulation layer. As shown in Fig. 5.6 the heat flux $\vec{\phi}_{\mathrm{q}}$ is directed towards the non-quenched neighbouring windings placed outside of a red, dashed circumference. Therefore, in order to account for the transverse quench propagation, it is required to create an algorithm that detects the temperature above the critical temperature, T_{c} in windings where the quench has not occurred, yet. In detail, the algorithm detects quenches and initiates a new quench front propagation when the temperature outside of the quenched zone exceeds the critical temperature of a superconductor. In other words, it is responsible for a turn-to-turn propagation across

the insulation layer between different windings.

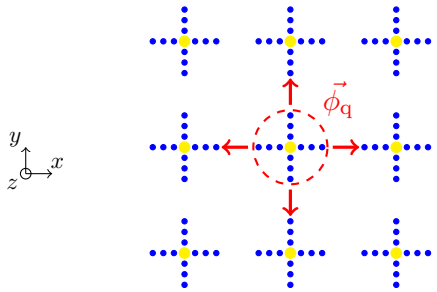

Figure 5.6: Heat flux direction from the quenched winding marked in the red circle.

In order to detect the quench, a list of all ANSYS nodes with their corresponding nodal temperatures is uploaded to the external routine at the communication point t_{com}. The nodes belonging to the strand are found and assigned to the 1D coil presented in Fig. 5.5. At this point, the external routine has a list of nodes with their values of temperature, and position in the coil coordinate $\bar{x}$ (see Fig. 5.5). Provided that searching starts at node N, winding number is W, magnetic field strength of the winding W is B_W, node temperature is T_N and critical temperature of related winding is $T_{\mathrm{c},}$, the problem is solved as described in Algorithm 2.

Algorithm 2 Quench Detection.

1: **for** N not quenched **do**
2: check the winding W which node N belongs to
3: assign magnetic field strength B_W of the winding W
4: calculate $T_{\mathrm{c},W}$ for the magnetic field strength B_W
5: **if** $T(N) > T_{\mathrm{c},}$
6: assign node N to a list of newly quenched nodes

5.4 Quench Front Position Assignment Algorithm

In order to couple an external routine calculating a quench position with a numerical solver, one has to assign a quench position P, being a discrete function of time, to a discrete nodal domain of the 1D imaginary coil with N_i number of nodes. The algorithm assigning a quench position is presented in Fig. 5.7. The algorithm is based on the bi-section search.

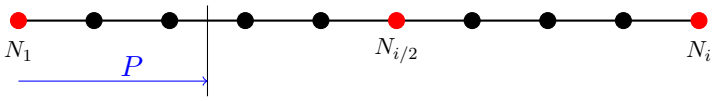

Figure 5.7: Bi-section search algorithm.

The algorithm aims at finding the node N_{search} which is close enough to the quench position P defined by the external routine that fulfills the condition described as

$$|P(N_{\text{search}}) - P| \leqslant \epsilon, \tag{5.1}$$

where $P(N_{\text{search}})$ – position of the found node in the discrete longitudinal mesh, P – quench position estimated by an external routine, ϵ – assumed accuracy.

Provided that searching begins at node N_1, number of nodes to check is N_i, quench position calculated by an external routine is P, accepted accuracy is ϵ, the problem is solved as described in Algorithm 3.

Algorithm 3 Quench Zone Assignment.

1: **while** $|P(N_{i/2}) - P| \geqslant \epsilon$ **do**
2: **if** $P(N_{i/2}) > P$ **do**
3: continue search in domain $D \in (N_1; N_{i/2})$
4: **elseif** $L(N_{i/2}) < L$ **do**
5: continue search in domain $D \in (N_{i/2}; N_i)$

When a mesh is very coarse and ϵ – relatively small, the given algorithm may never converge. Therefore, if the algorithm gives the same node N_{search} in two consecutive iterations, while still not satisfying the condition described in (5.1), it is assumed that the current node N_{search} satisfies the assignment requirements.

Chapter 6

Python Implementation

6.1 Code Overview

As explained in Chapter 4, the co-simulation in the quench velocity-based approach requires an external routine that controls the quench propagation over time. The routine is developed in Python and has four main functions:

1. Translation of the Python code into ANSYS APDL internal commands in order to allow the external routine to communicate with the simulation tool.
2. Implementation of pre-processing, solution and post-processing steps in the electro-thermal simulations conducted in ANSYS APDL.
3. Implementation of all algorithms explained in Chapter 5.
4. Execution of the ANSYS simulation employing the quench velocity-based approach.

The first three points also allow the Python code to conduct a quench simulation in ANSYS without the quench velocity-based approach. Therefore all the ANSYS simulations prepared in this thesis are based on the developed Python script. The exemplary code depicting how the Python communicates with ANSYS APDL is described in Appendix D.1. In order to run the programme, Python 3.6 is required as well as ANSYS APDL 2019 R1. The programme is open source and available on GitLab[1] under the link given in [32].

As presented in Fig. 6.1, the script is divided into four main components: execution script, general purpose classes, workflow classes and main classes. The programme uses a json text file in which all input parameters required for the analysis are specified. The

[1] GitLab is an official code repository at CERN.

execution script is a core of the programme performing consecutive steps of the analysis. The workflow classes are created to execute the code similarly to ANSYS APDL, i.e. the procedures corresponding to individual methods are collected in the steps of preprocessing, solving and postprocessing as it is carried out in GUI[2] of ANSYS APDL. Therefore, the script remains intuitive with respect to standard ANSYS workflows. The workflow classes use the main classes in which the core methods of the Python script are implemented. The general purpose classes are used by most of the remaining classes. Within the set of general purpose classes, a factory pattern is implemented in which a specific case for the analysis is chosen based on the input json file filled by a user.

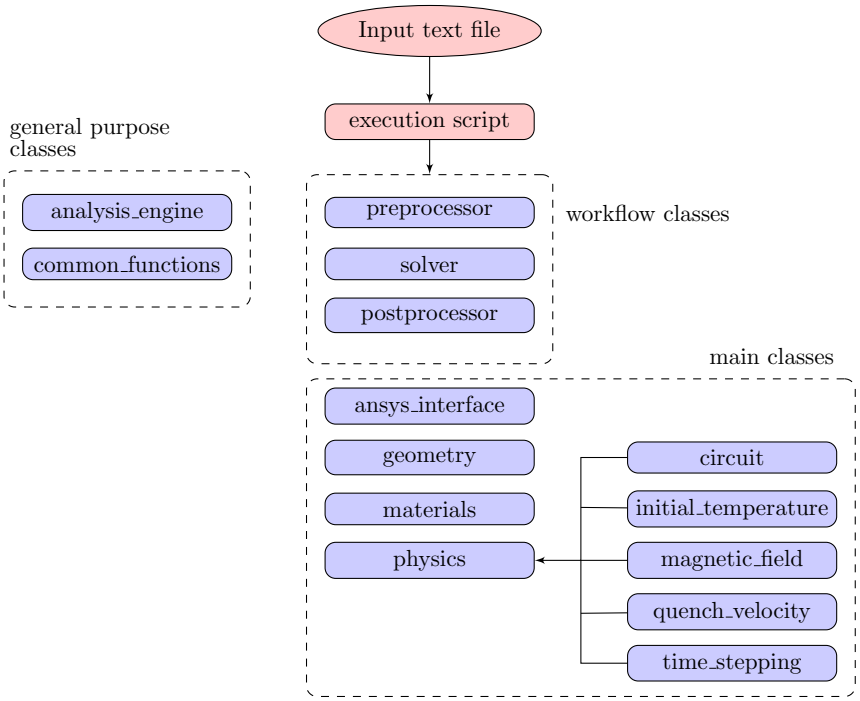

Figure 6.1: Schematic of the Python script.

As described in Table 6.1, the user can conduct a standard quench analysis in ANSYS and the one relying on the quench velocity-based approach. The Python script is able to create two types of geometries: a high-order corrector magnet as well a stack of extruded strands. The analysis may be executed with or without the insulation layer, optionally including the resin. The initial temperature is a Gaussian profile or a step function. The script has a repository of thermal and electric material properties of copper, Nb-Ti and G10 based on the fits provided by NIST (see Appendix A). When the quench velocity-based

[2]GUI – Graphical User Interface

approach is considered, the user can specify the quench velocity as a constant parameter or a function of current and magnetic field strength. The multi-strand simulations are dedicated to use the quench velocity-based approach. The quench velocity-based approach is equipped with a tool able to simulate the discharge of a magnet with a varying inductance as a function of current and the dump resistor connected in series to a magnet. Since some of the material properties of copper and Nb-Ti (see in Appendix A) are a function of the magnetic field strength, the user can specify whether the magnetic field strength is constant or – a function of a position in (x, y)-plane from the magnet cross-section, as presented in Fig. 5.4. One can define multiple maps for different values of current.

Table 6.1: Analysis options.

analysis type	**standard**	**quench-velocity based**
geometry	high-order corrector, extruded (multi-)strand	
analysis with insulation	available	
initial temperature profile	step function, Gaussian profile	
available material properties	Cu, Nb-Ti, G10 (all based on NIST)	
quench velocity model	N/A	constant, function of current and magnetic field strength
circuit	constant current	constant current, RL discharge with dump resistor
B-field	constant	constant, function of current and position in (x, y)-plane

6.2 Quench Analysis Workflow

As illustrated in Fig. 6.2, there are four steps required to conduct the multi-strand analysis of a high-order corrector with the quench velocity-based approach. In the first step, a 2D magnetic field map as a function of current is created. The assignment of a magnetic map to the windings is presented in Section 5.2. The average quench velocity also depends on the magnetic field strength and the current. In the second, it is required to conduct a set of standard analyses to define the quench velocity as a function of the magnetic field strength and the current. The third step of the workflow aims to find the minimum propagation zone of an insulated strand. This is required because the spot heater is not included in the simulation. Therefore, one should set an initial temperature profile that leads to a natural quench initiation. When all three steps are completed, one can start simulating the discharge of a high-order corrector.

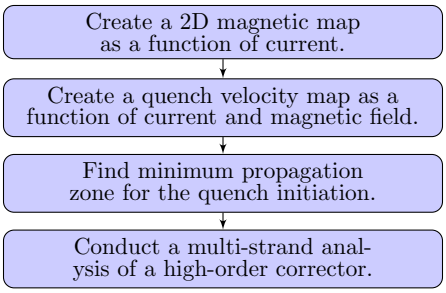

Figure 6.2: Analysis workflow for study of a high-order corrector.

The first step of the workflow aims to create a 2D magnetic map of the coil as a function of current. This is a separate simulation independent of the Python implementation. The Python code only imports the magnetic map from an external folder. The next three analysis steps from the workflow in Fig. 6.2 are run in Python. The analysis options of those steps are presented in Table 6.2.

Table 6.2: Analysis options of specific simulations in the workflow.

parameter	**quench velocity map**	**minimum propagation zone**	**multi-strand analysis**
analysis type	standard	standard	quench velocity-based
geometry	extruded strand	extruded strand	high-order corrector
analysis with insulation	yes/no	yes/no	yes
initial temperature profile	Gaussian	Gaussian	Gaussian
quench velocity model	N/A	N/A	function of current and magnetic field strength
circuit	constant current	constant current	RL discharge with dump resistor
B-field	constant	constant	function of current and position in (x, y)-plane

The quench velocity map and the minimum propagation study have identical settings. They both require the standard analysis with the geometry of an extruded strand. The analysis can be conducted with or without the insulation elements. Both simulations are also conducted at constant current and magnetic field strength, i.e. these parameters do not change for a single analysis. The multi-strand analysis requires the quench velocity-

based approach with the geometry of a high-order corrector. The quench velocity model is uploaded as an external file based on the quench velocity map study. The magnetic field is uploaded from a folder containing a set of magnetic field maps for varying values of current. Moreover, the discharge of the magnet is simulated in an RL-circuit.

6.3 Description of Execution Script

The execution script performs consecutive steps of the analysis. First of all, it uploads the user's input parameters from a json file. Moreover, the magnetic field map as well as the quench velocity settings are uploaded. The script specifies material properties, creates a magnet geometry, applies boundary and initial conditions, sets a time step, communicates with ANSYS to solve the given problem and processes the obtained results. The script also adjusts material properties and nonlinear inductance, if required, as the analysis continues. There is a single execution script for the standard analysis and the quench velocity-based approach. The script is written in such a manner to correspond to all possible cases of the analysis presented in Fig. 6.1. If a given case does not require some of the presented methods, they are omitted.

Provided that the initial time step is t, the total simulation time is t_{total}, the problem is solved as described in Algorithm 4. The statements corresponding to the preprocessor, solver and postprocessor in Fig. 6.1 are marked in red, blue and green respectively. The execution script written in Python, is presented in Appendix D.2. In Algorithm 4, the methods from 1 to 9 are executed only once. The remaining steps are conducted until either the analysis reaches the assumed simulation time or the current in the magnet becomes a desirably low value during the discharge process.

Algorithm 4 Description of the execution script implemented in Python.

```
read user's input parameters from the text file
read magnetic field map
read quench velocity map
define the material properties
create the geometry
connect the circuit to the geometry
create the initial temperature profile
check the initial quench state in the temperature profile
adjust the resistive material properties in the quenched zone in ANSYS APDL
couple separate windings thermally and electrically in ANSYS APDL
apply initial temperature profile in ANSYS APDL
while magnet is not discharged or t < t_total do
    set the current time step t
    solve the temperature distribution and current for time step t
    get the temperature profile and current from ANSYS APDL
    estimate the resistive voltage from ANSYS APDL
    if magnet is not fully quenched do
        check the quench state
        update the magnetic field in windings
        adjust the material properties with quench propagation
    adjust the material properties with current drop
    check if the quench is detected with the quench detection system
    update the nonlinear inductance with a current drop
```

6.4 Implementation of ANSYS Models and Algorithms in Python

In this section, a part of the developed code directly related to ANSYS models and algorithms explained in Chapter 5 is presented. In order to understand the entire programme and the relations between different classes, it is recommended to study the available scripts on GitLab under the link given in [32]. Even if some of the classes are presented in appendices, a detailed explanation of the entire logic is beyond the scope of this thesis.

6.4.1 Models Implementation

As shown in Fig. 6.3, the ansys_interface module consists of two main classes: **AnsysMultiple1DSlab** and **AnsysMultiple1DHighOrderCorrector**. Each of them inserts initial APDL parameters for the analysis corresponding to the simulated case. They also

load a parametrised ANSYS APDL script responsible for the creation of a geometry. An exemplary APDL script for **AnsysMultiple1DSlab** is presented in Appendix C.1. **AnsysMultiple1DSlab** and **AnsysMultiple1DHighOrderCorrector** inherit the generation of material properties as well as the APDL functions corresponding to the boundary conditions from **AnsysMultiple1D**. Moreover, **AnsysMultiple1D** inserts an APDL script corresponding to the solution of a problem. It is presented in Appendix C.2. All three classes require external data from the geometry module concerning the insulation geometric functions. In addition, **AnsysMultiple1D** inherits the translation of APDL commands into a Python script from **Ansys**.

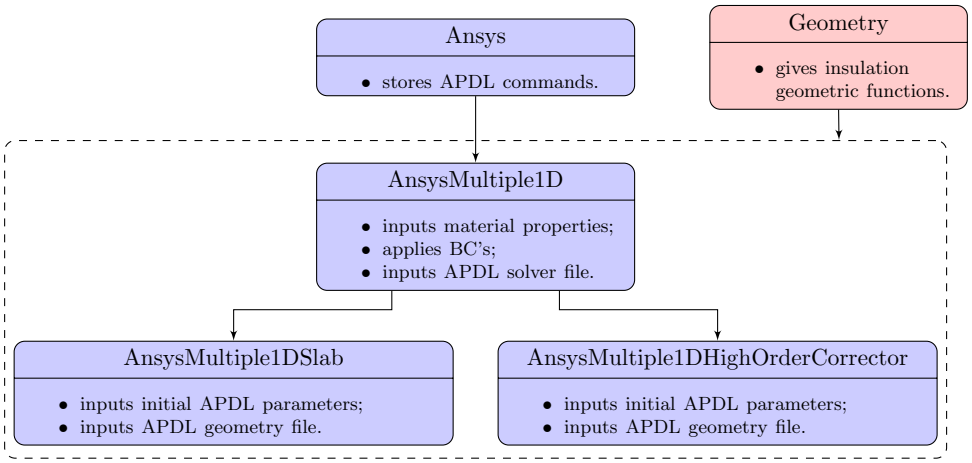

Figure 6.3: Class diagram of the module ansys_interface.

6.4.2 Multidimensional Mapping Algorithm

The multidimensional mapping algorithm is implemented in **GeometryMulti1D**, as shown in Fig. 6.4. It maps a multi-dimensional coil into a 1D coil and lists all important real nodes of the ANSYS geometry that are important for the simulation such as BC's, IC's, thermo-electric coupling, etc. **GeometryMulti1D** inherits from **Geometry** that includes methods applicable to problems of every dimensionality, not only the ones corresponding to the case 1D+1D(+1D for a multi-strand analysis). Therefore, a possibility of the code extension to higher dimensionalities such as 2D or 3D is taken into consideration. The geometry module is not included in the appendices due to its considerable size.

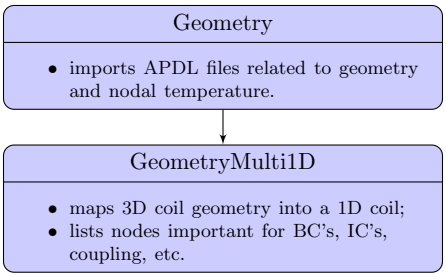

Figure 6.4: Class diagram of the module geometry.

6.4.3 Quench Velocity Assignment and Quench Front Position Assignment Algorithms

The classes related to the quench_velocity module are shown in Fig. 6.5. As mentioned in Table 6.1, there are two quench velocity models: a constant quench velocity and the one being a function of a magnetic field strength and current. The latter is implemented in **QuenchFrontNumerical** and the former – in **QuenchFrontConstant**. They both calculate a continuous quench front position at a given time step. The class **QuenchFrontNumerical** inherits a quench velocity map from **QuenchVelocityMap** being a function of current and magnetic field strength. The blocks marked in red are external modules that communicate with the quench velocity module. **QuenchVelocityMap** uses linear interpolation functions from the common functions module.

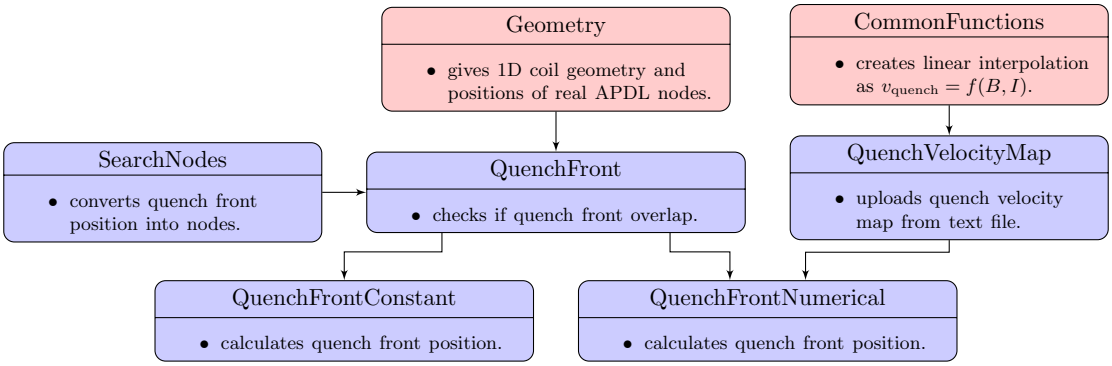

Figure 6.5: Class diagram of the module quench_velocity.

QuenchFrontConstant and **QuenchFrontNumerical** inherit from **QuenchFront** being the main class of the quench_velocity module. **QuenchFront** inherits from **SearchNodes** the quench front position assignment algorithm described in Section 5.4. **SearchNodes** written in Python is presented in Appendix D.3. Class **QuenchFront** also inherits from the geometry module required for searching nodal quench positions.

6.4.4 Quench Detection Algorithm

Figure 6.6, presents an overview of a postprocessor module. It is the only workflow module depicted in this section because it includes an implementation of the quench detection algorithm. There are two main classes in the module used by the execution script: **PostProcessorHeatBalance** with a standard solution and **PostProcessorQuenchVelocity** with a quench velocity-based approach. They both estimate the resistive voltage V_{res} in the coil and check the quench state of a superconducting magnet.

PostProcessorHeatBalance and **PostProcessorQuenchVelocity** inherit from **PostProcessor** that stores a magnetic field map and a list of quenched fronts. Each quench front is a separate class object of **QuenchFront** storing its current quench front position. **PostProcessor** inherits after **QuenchDetect** the quench detection algorithm. It is described in Section 5.3. In addition, **QuenchDetect** in Python is presented in Appendix D.4. In order to initiate **QuenchDetect**, the input data from external class are required: **Geometry** and **MaterialProperties**. **PostProcessor** also inherits from the internal class **Plots** allowing for saving the results as figures and text files.

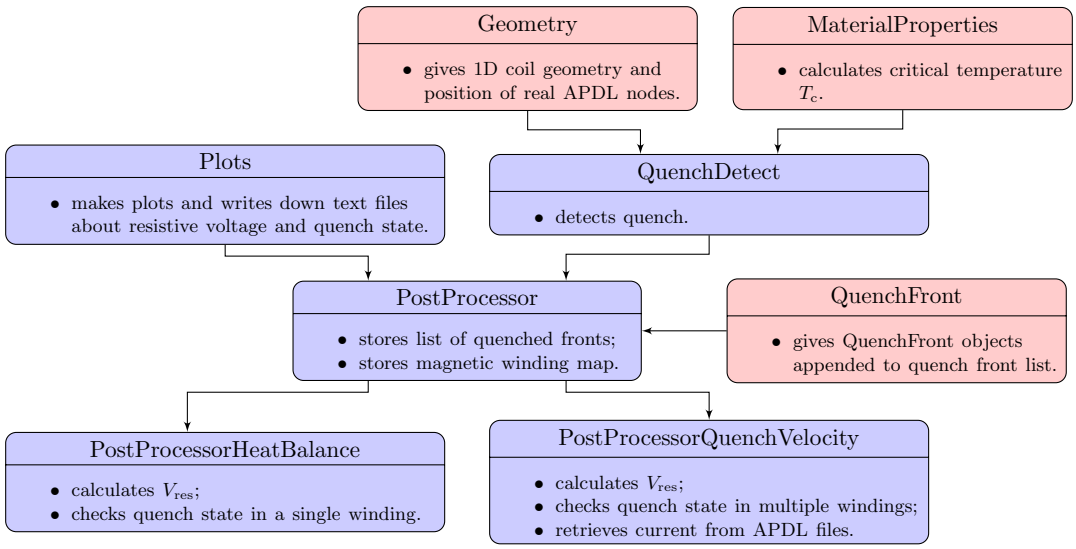

Figure 6.6: Class diagram of the module postprocessor.

Chapter 7

Verification of Quench Velocity-Based Approach

7.1 Motivation

As it is demonstrated in Chapter 3, a refined mesh with a reduced time step is required in order to solve a time-dependent temperature distribution over a 1D strand. This section compares the standard ANSYS analysis of the quench propagation with the quench velocity-based approach presented in Chapter 4. The aim of this benchmark is to verify what the computing time gain is, and a possible longitudinal mesh size relaxation with the quench velocity-based approach. Moreover, the relation between the mesh relaxation, and the evolution of the error associated with the peak temperature is studied. The numerical analyses using the quench velocity-based approach are co-simulated with the external routine written in Python described in Chapter 6, with algorithms implemented as outlined in Chapter 5. The benchmark criteria for the standard analysis and the quench velocity-based approach are:

1. Mesh size
2. Time step
3. Computation time
4. Relative error with respect to the hot-spot temperature and resistive voltage

7.2 Verification Methodology

Similarly to the simulations performed in Chapter 3, the benchmarking study between the standard ANSYS analyses and the quench velocity-based approach is conducted for two cases separately:

1. Analysis of a bare composite strand
2. Analysis of the composite strand with insulation and epoxy resin

In order to conduct the analysis with the quench velocity-based approach, the time evolution of the quench velocity is required. The standard ANSYS analysis serves two purposes:

1. The average quench velocity is calculated based on the standard ANSYS simulation in order to implement it in the quench velocity-based approach.
2. The results from the quench velocity-based approach are compared with the reference standard analysis in order to estimate a relative error with respect to the temperature and the resistive voltage along the strand.

Figure 7.1 illustrates the benchmarking workflow of the verification methodology implemented to study the case of a bare composite strand. The default mesh size of $x = 1$ mm, recommended in previous chapters, is the starting point of the benchmarking process with a relatively large time step range of $t_{\text{step range}} = [0.1, 1]$ ms. The verification is divided into three loops:

1. The time step loop in which an optimised time step range for the reference solution is searched.
2. The mesh size loop that aims at reassuring the sufficient quality of the initial mesh size.
3. The verification loop in which the quench velocity-based simulations with a relatively relaxed longitudinal mesh along the strand are compared with the reference standard ANSYS results.

The first two loops in the workflow have a condition statement of a relative error with respect to the average quench velocity. The relative error range is calculated as

$$E_{\text{r}} = \frac{r_{\text{i}} - r_{i-1}}{r_{\text{i}}}\ 100\%, \tag{7.1}$$

where r_i – result from the reference analysis, r_{i-1} – result from the preceding analysis of the iteration loop $i-1$.

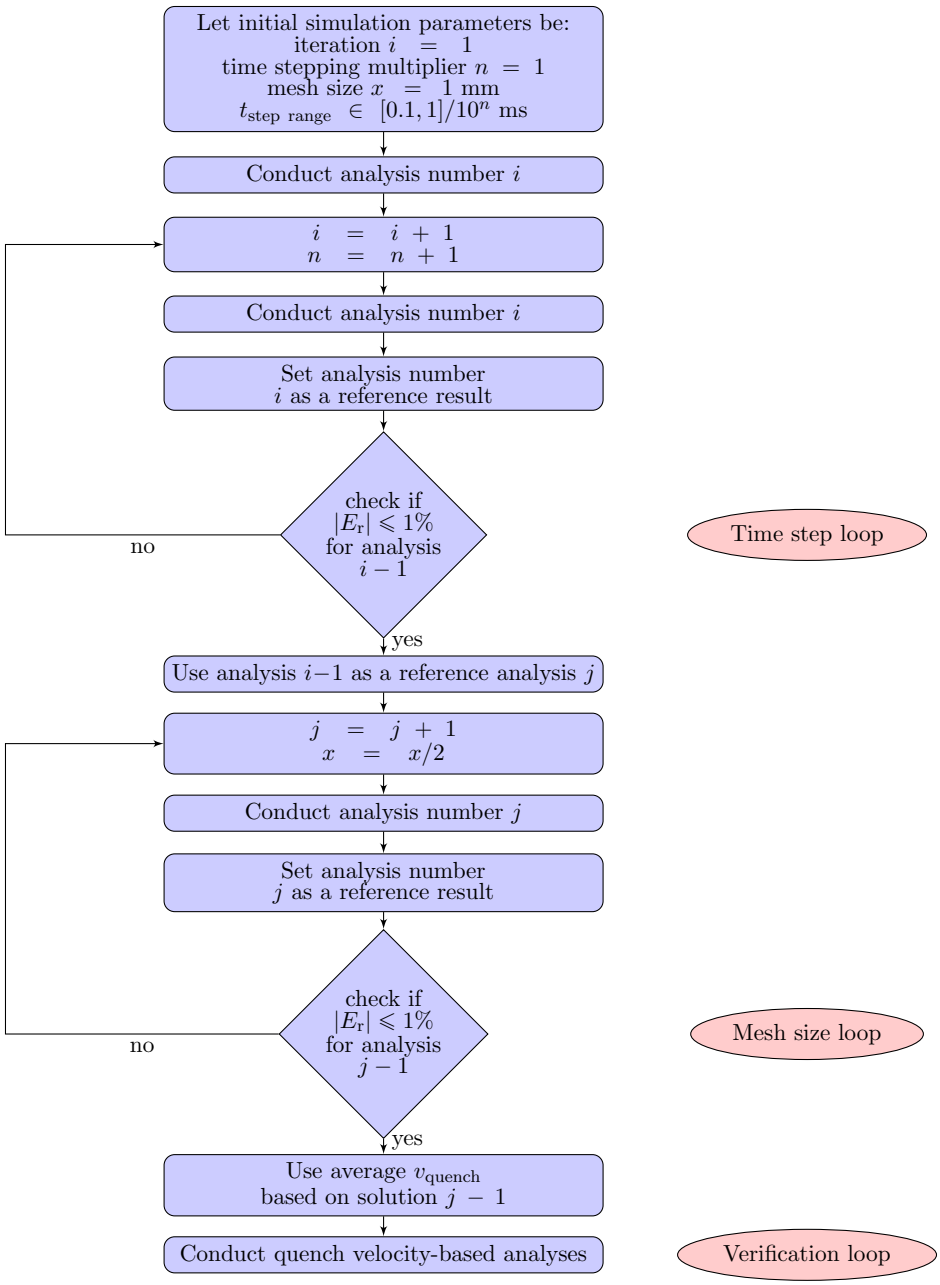

Figure 7.1: Block diagram of the verification methodology for the case of a bare strand.

The condition statement described in (7.1) imposes an absolute value of a relative error

less or equal to 1%. In this chapter, the quench velocity is formulated in a different manner with respect to Chapter 3, where the reference results are obtained with STEAM-BBQ [20]. The average quench velocity is calculated in this case as

$$v_{\text{quench}} = \frac{\sum_{j=1}^{k-1} \frac{x_{j+1}-x_j}{\Delta t}}{k}, \tag{7.2}$$

where x_j – quench front position at time window j with the initial time window at $t = 0$, Δt – time increment between two time windows j and $j+1$, k – number of time windows j. The time increment for the standard analyses is equal to $\Delta t = 10$ ms. In addition, all material properties are based on fits provided by NIST described in Appendix A.

The verification methodology for the analysis of the composite strand with insulation and epoxy resin is illustrated in Fig. 7.2.

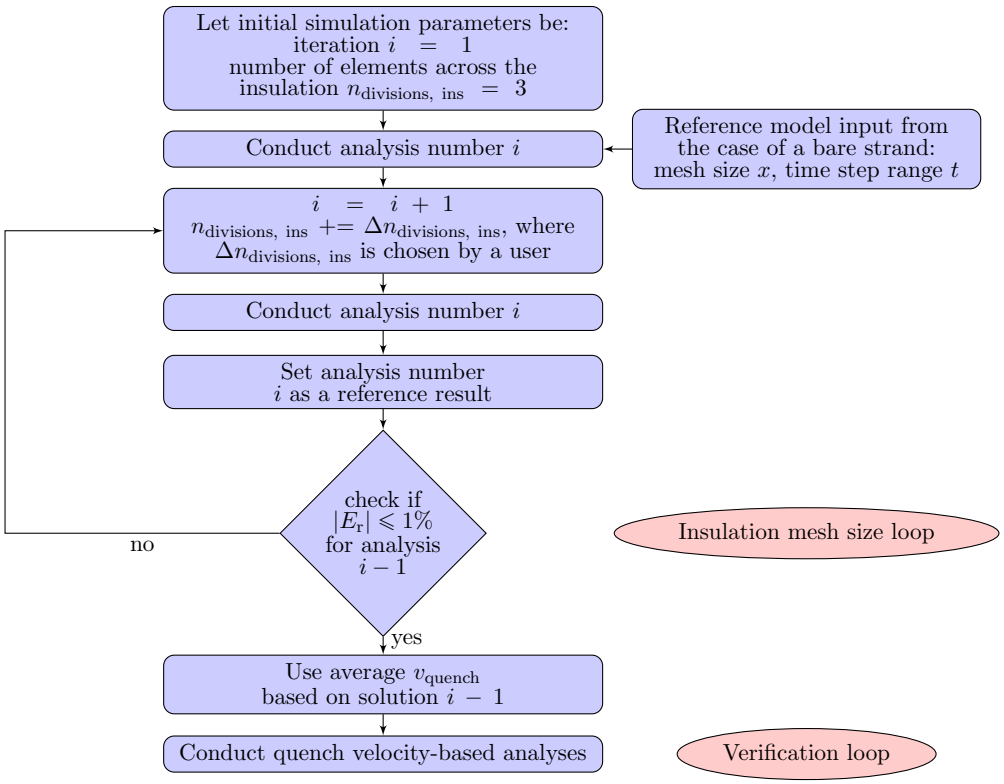

Figure 7.2: Block diagram of the verification methodology for the case of a strand with insulation and epoxy resin.

The presented workflow contains an iteration loop in which the mesh size across the insulation is searched to satisfy the condition (7.1) based on the quench velocity as calculated

in (7.2). When the condition statement of the relative error is fulfilled, the average quench velocity is taken from the reference standard model, and set as an input parameter in the benchmarking loop. The occurrence of only two iteration loops in the diagram is due to the fact that the mesh size, and the time step range are inherited from the analysis of a bare strand.

The study in the verification loop remains identical for the case of a bare strand, and that including insulation and epoxy resin. There are three comparisons made between the standard analysis and the quench velocity-based approach:

1. Difference in the nodal temperature along the strand
2. Evolution of the resistive voltage over time
3. Evolution of the hot-spot temperature over time

The difference in the nodal temperature along the strand is only compared for the nodes existing in the most relaxed mesh. The relative error with respect to the reference standard solution is estimated for the the resistive voltage and the hot-spot temperature. Moreover, the possibility of increasing the time step range $t_{\text{step range}}$ is studied in the verification loop of a bare strand.

7.3 1D Analysis of a Strand

7.3.1 Standard Analysis

In this study, the standard ANSYS simulation of a bare strand is performed based on the LINK33 element. The geometric assumptions, the initial conditions, and the analysis settings related to ANSYS are the same as in Section 3.4. One can observe, that the initial Gaussian temperature profile (see Fig. 3.4) is more approximated, if the mesh along the strand is relaxed. Therefore, the initial energy deposition in the strand depends on the mesh size as well. The initial energy in the discretised domain is calculated as

$$E_{\text{initial}} = \sum_{i=1}^{n-1} V_{i,i+1}\ c_{\text{v, strand}}\left(T = \frac{T_i + T_{i+1}}{2}\right)\ \frac{T_i + T_{i+1}}{2}, \tag{7.3}$$

where n – number of nodes in the initially quenched zone, $V_{i,\ i+1}$ – volume of the domain between two neighbouring nodes, $c_{\text{v, strand}}$ – volumetric heat capacity calculated for an average temperature, T_i – temperature of a node i, T_{i+1} – temperature of a neighbouring node $i+1$.

The relation between the initially deposited energy in the strand and the number of nodes over a 1 metre-long domain is presented in Fig. 7.3. The exact energy deposition starts converging for approximately 1000 nodes which corresponds to the mesh size of 1 mm. A lower number of nodes would slow down the quench front propagation.

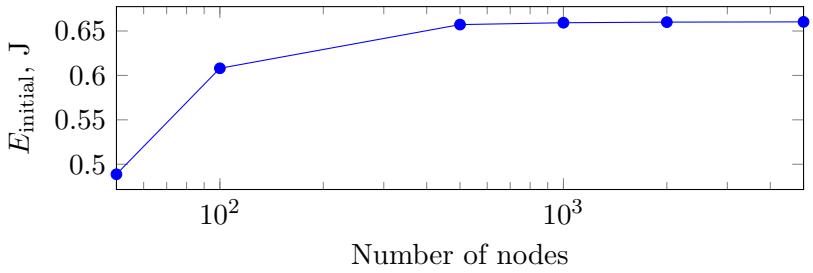

Figure 7.3: Initial energy deposition along the 1 metre-long strand as a function of number of nodes.

As illustrated in the block diagram of the analysis workflow corresponding to the bare strand in Fig. 7.1, the initial simulation is performed with the longitudinal mesh size of 1 mm and the time step range $t_{\text{step range}} = [0.1, 1]$ ms. The time step loop consists of two iterations, as shown in Table 7.1. The time step range $t_{\text{step range}} = [0.01, 0.1]$ ms fulfills the condition statement related to the average quench velocity. In the next step, the mesh size loop consists of a single iteration in which the quality of the initial mesh size of 1 mm is confirmed by comparing the result with the analysis in which the number of nodes is doubled, as shown in Fig. 7.4.

Table 7.1: Quench results for analysed time step ranges.

$t_{\text{step range}}$, ms	[0.001, 0.01]	[0.01, 0.1]	[0.1, 1]
$v_{\text{quench, average}}$, m/s	6.80	6.81	7.1
E_{r}	-	0.17%	4.26%

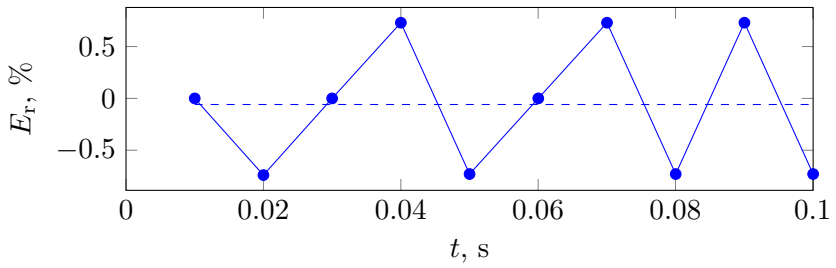

Figure 7.4: Relative error of the incremental and average (dashed line) quench velocity for the mesh size of 1 mm with respect to the mesh size of 0.5 mm; analysis conducted with the time step range of $t_{\text{step range}} = [0.01, 0.1]$ ms.

In conclusion, the reference used for the verification loop in Fig. 7.1 is the ANSYS solution with a mesh size of 1 mm and a time step range of $t_{\text{step range}} = [0.01, 0.1]$ ms. The parameters are summarised in Table 7.2.

Table 7.2: Input parameters of the reference standard solution.

parameter	value	unit
mesh size	1	[mm]
$t_{\text{step range}}$	[0.01, 0.1]	[ms]
$v_{\text{quench, average}}$	6.81	[m/s]

The reference solution is an acceptable compromise between accuracy and computing time. As presented in Fig. 7.5, in a 1D quench propagation, computing time rises monotonically with the mesh refinement while keeping the time step range equal to $t_{\text{step range}} = [0.01, 0.1]$ ms[1].

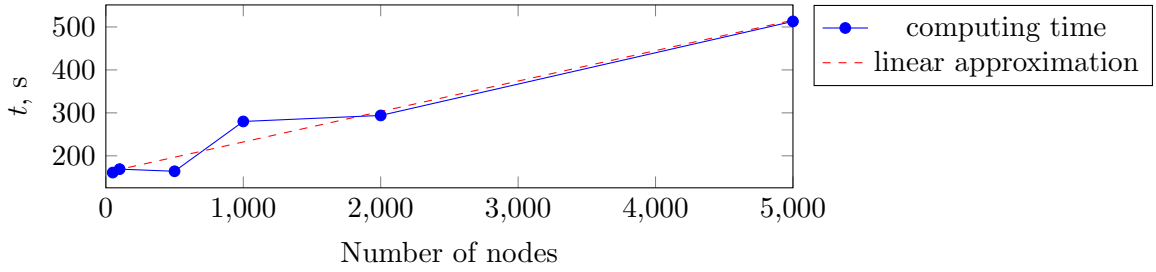

Figure 7.5: Computing time as a function of number of nodes.

[1]The analysis was performed on the following calculation unit: Intel(R) Xeon(R) CPU E5-2667 V4 @ 3.20 GHz (2 processors) with RAM 128 GB.

7.3.2 Analysis with Quench Velocity-Based Approach

In the quench velocity-based approach, an electro-thermal element LINK68 is used to represent the composite strand, as discussed in Section 5.1.1. Figure 7.6 illustrates additional initial and boundary conditions that allow for applying a power source identically to the standard analysis discussed in Section 7.3.1. The parameters related to the co-simulation in the quench velocity-based approach are shown in Table 7.3.

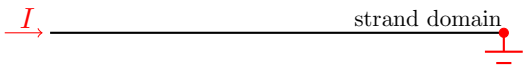

Figure 7.6: Electric boundary conditions.

Table 7.3: Input parameters in the quench velocity-based approach.

parameter	value	unit
t_{com}	2.5	[ms]
$v_{\text{quench, average}}$	6.81	[m/s]

First of all, the possibility of increasing the time step range is checked. Two time step ranges are chosen: $t_{\text{step range}} \in \{[0.01, 0.1], [0.1, 1]\}$ ms. In this comparison, the average quench velocity cannot be the condition statement corresponding to the relative error because it is an input parameter in the quench velocity-based approach. Therefore, the condition statement is based on the relative error of the resistive voltage with respect to the reference standard analysis. The results from two simulations having a different time step range, and relying on the quench velocity-based approach were compared with the reference solution. It appeared that the relative error remained below 0.1% in both cases. Therefore, the time step range $t_{\text{step range}} = [0.1, 1]$ ms is used in the quench velocity-based approach for the analysis of a bare strand, and with insulation and epoxy resin. In the analyses of a bare strand, the study over a 1 metre-long domain is conducted with a varying mesh size, $m = \{1, 2, 10, 20, 33.3\}$ mm. The geometric assumptions as well as the initial conditions are the same as in Section 3.4. As presented in Fig. 7.7, the quench velocity-based approach results in an underestimation of the nodal temperature along the strand.

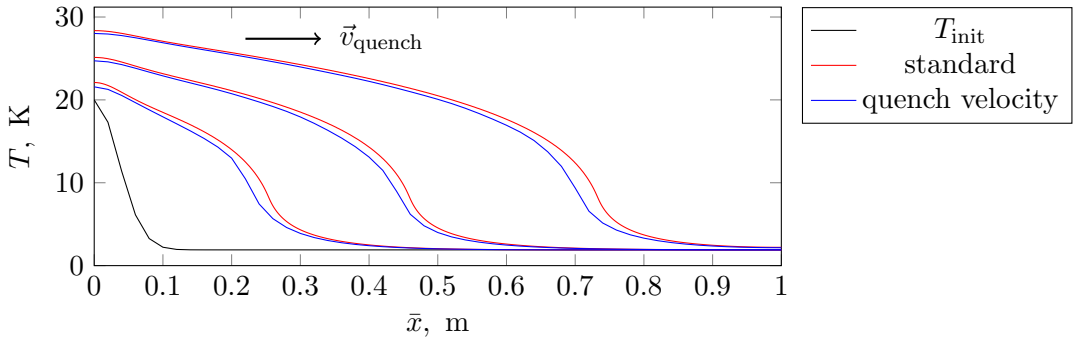

Figure 7.7: Temperature distribution of the reference solution and the quench velocity-based approach with the mesh size of 20 mm for three time steps: $t = \{0.03, 0.06, 0.1\}$ s with a specified direction of quench velocity, $\vec{v}_{\text{quench}}$.

The reasons for the differences in the nodal temperature along the strand are twofold:

1. In Fig. 4.4, in which the schematic of a one-directional coupling is presented, it is explained that the external routine updates resistive material properties at communication point t_{j-1} and ANSYS solves the case for t_j. Therefore, the quenched zone is underestimated and 'delayed' with respect to the standard solution. Depending on the applied current and magnetic field to the model as well as the material properties of the geometry components, the error in the temperature distribution may stabilise or grow in time. To reduce this error, the number of communication points t_{com} is increased to 40, which corresponds to a time window of $t = 2.5$ ms. Nevertheless, this error is a natural consequence of applying the co-simulation.

2. In the quench velocity-based approach, the material properties assignment to the strand is binary. The material has either resistive properties of the strand composite above its critical temperature or no resistance below this value. The transition region of a current sharing temperature, being lower than the critical temperature, is not taken into account. The heat capacity of both Nb-Ti and copper is relatively low at critical temperatures. Therefore, even a small difference in heat deposition results in a large temperature difference in the transition region.

Not including the current sharing phenomenon in the standard analysis would reduce the discrepancy between two approaches during the benchmarking. However, the comparison of two analysis types aims at finding the maximum numerical error that the quench velocity-based approach gives. The study of separate factors, having an influence on the total discrepancy between the approaches, is not the goal of this chapter. As shown in Fig. 7.8, the resistive voltage in the quench velocity-based approach follows the curve of

the standard solution. However, the resistive voltage is underestimated because the strand resistivity is a function of temperature, which is underestimated as well.

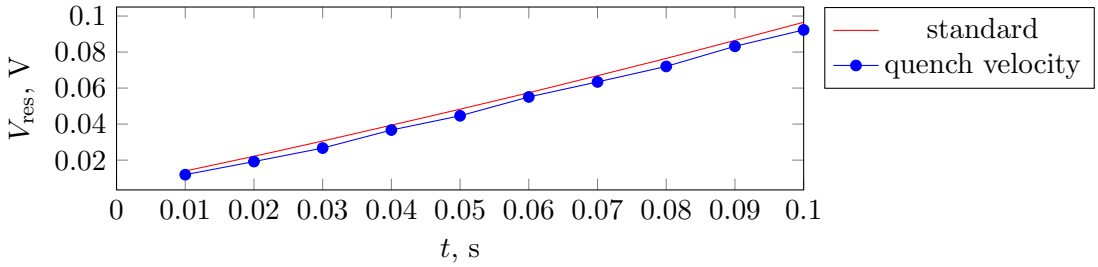

Figure 7.8: Comparison of the resistive voltage between the standard solution and the quench velocity-based approach with the mesh size of 20 mm.

Figures 7.9 and 7.10 depict the evolution of the resistive voltage and the hot-spot temperature in time, respectively. The longer the simulation lasts, the lower the relative error with respect to the reference solution in both cases becomes. One can observe that analysis with the mesh size of 1 mm also shows a similar evolution of the relative error. As the reference solution is characterised by the identical mesh size of 1 mm, the error cannot be explained by the decrease of mesh density, as presented in Fig. 7.3. This is due to the assumption of constant quench velocity which is not constant in the reference model. In fact, the quench propagation is a dynamic process which requires some time for the quench velocity to reach a steady-state value. To conclude, the error corresponding to the hot-spot temperature and the resistive voltage has to be accepted if one performs a simulation relying on the quench velocity-based approach.

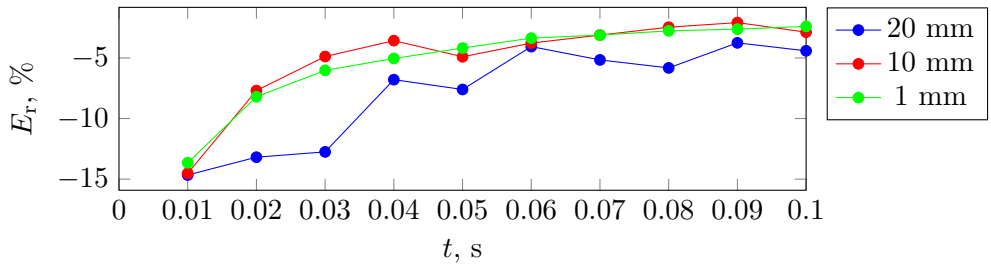

Figure 7.9: Relative error of the resistive voltage for three different mesh sizes used for the quench velocity-based approach.

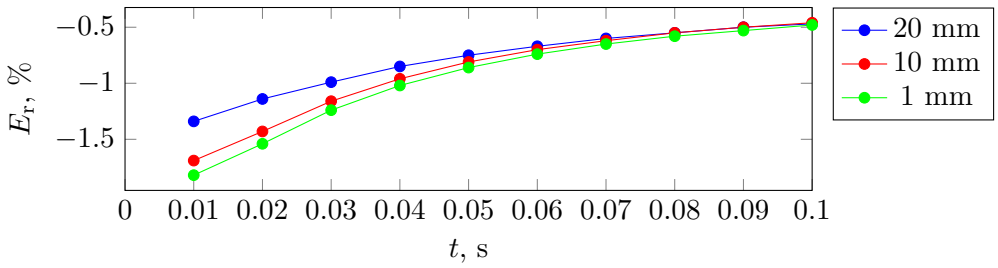

Figure 7.10: Relative error of the hot-spot temperature for three different mesh sizes used for the quench velocity-based approach.

7.3.3 Conclusion

As presented in Table 7.4, when the quench velocity-based approach is used, the mesh density is reduced by a factor ranging from 10 to 20. In addition, the time step range is increased by a factor of 10 while keeping the results within a precision of 0.01% with respect to the evolution of the resistive voltage. However, the computation time of the quench velocity-based approach remains similar to the standard analysis. It is a result of applying the co-simulation procedure with an external routine which costs time when the signal is exchanged with ANSYS. In the presented studies, the co-simulation lasts approximately 80 s. Subtracting this value from the total computation time gives a real value which ANSYS requires to solve the quench problem.

Table 7.4: Bare strand benchmark summary.

	mesh size, mm	$t_{\text{step range}}$, ms	computing time, s
standard analysis	1	[0.01, 0.1]	280
quench velocity-based approach	10-20	[0.1, 1]	280
improvement rate	10-20 times	10 times	none

Table 7.5 summarises the evolution of the resistive voltage and the hot-spot temperature in time. With the mesh size of 20 mm, the relative error converges to -5% for the resistive voltage and to -0.5% for the hot-spot temperature. The study indicates that the results converge to a stable error as the quench propagates during the simulation. Nevertheless, the results do not imply that the convergence occurs in more extreme cases when more energy is deposited in the strand, i.e. when the strand is subjected to higher current or magnetic field strength. The case of a bare strand is a relatively simple example. In the simulation of the skew quadrupole, the insulation and epoxy resin ought to be taken into consideration as well.

Table 7.5: Comparison of relative error: resistive voltage and hot-spot temperature.

mesh size, mm	V_{res}		$T_{\text{hot-spot}}$	
	$t = 0.01$ s	$t = 0.1$ s	$t = 0.01$ s	$t = 0.1$ s
1	-13.65%	-2.39%	-1.82%	-0.48%
10	-14.52%	-2.87%	-1.69%	-0.46%
20	-14.66%	-4.41%	-1.34%	-0.47%

7.4 1D Analysis with Insulation and Epoxy Resin

7.4.1 Standard Analysis

As described in the block diagram in Fig. 7.2, this section focuses on finding the mesh size across the insulation layer. The longitudinal mesh size and the time step range are taken from the reference solution obtained in the study of the bare strand. The geometric assumptions, the initial conditions, and the analysis settings are the same as in Section 3.5. In this analysis, the initial number of nodes accross the insulation is $n_{\text{divisions, ins}} = 3$ which corresponds to the minimum number of nodes required to study the nonlinear material properties of the insulation. The iteration loop i is performed four times for the insulation mesh size with the increment of $\Delta n_{\text{divisions, ins}}$ in order to obtain the following number of insulation nodes $n_{\text{divisions, ins}} = \{3, 5, 10, 20, 30\}$. The number of nodes accross the insulation layer corresponds to the mesh size of $m = \{56, 34, 17, 8, 6\}$ µm, respectively. Figure 7.11 shows that with 20 nodes across the insulation, the relative error of an average quench velocity remains within the tolerance of 1% with respect to the analysis with 30 nodes.

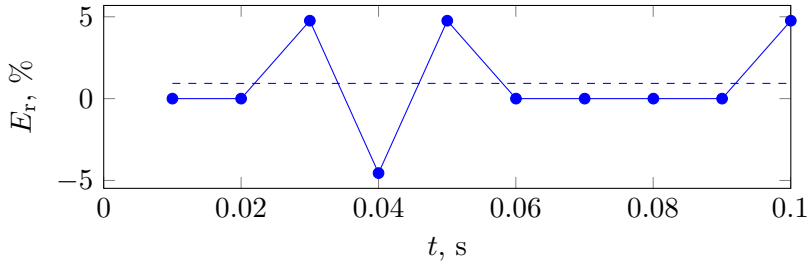

Figure 7.11: Incremental and average (dashed) quench velocity relative error for 20 nodes across the insulation layer.

The stabilisation of the solution can also be observed in Fig. 7.12. Here, the evolution of the hot-spot temperature in time is presented as a function of a number of nodes across the insulation layer. The temperature clearly converges with 20 nodes.

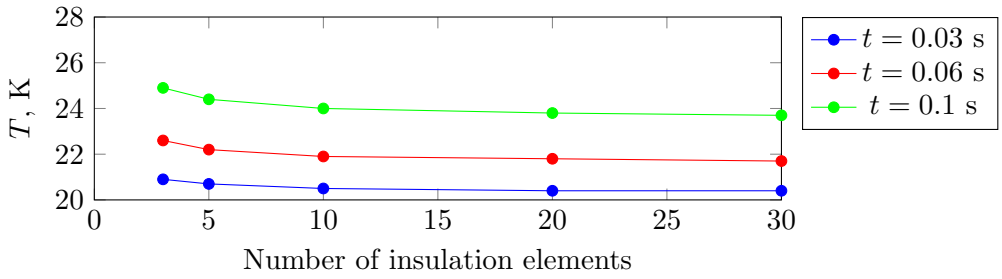

Figure 7.12: The evolution of the hot-spot temperature at three different time steps $t = \{0.03, 0.06, 0.1\}$ s as a function of a varying number of insulation elements.

The parameters used for the benchmarking purposes with the quench velocity-based approach are summarised in Table 7.6. The reference analysis is that with a mesh size of 1 mm, a time step range of $t = [0.01, 0.1]$ ms, and 20 nodes across the insulation layer.

Table 7.6: Analysis input parameters.

parameter	value	unit
longitudinal mesh size	1	[mm]
insulation mesh size	8	[µm]
$t_{\text{step range}}$	[0.01, 0.1]	[ms]
$v_{\text{quench, average}}$	2.18	[m/s]

In the 1D quench analysis of a 1 metre-long strand with insulation and epoxy resin, the computing time rises monotonically when the number of nodes across the insulation layer increases. This is presented in Fig. 7.13. The analysis with 20 nodes lasts over 2 hours[2]. Therefore, the further mesh refinement is not recommended.

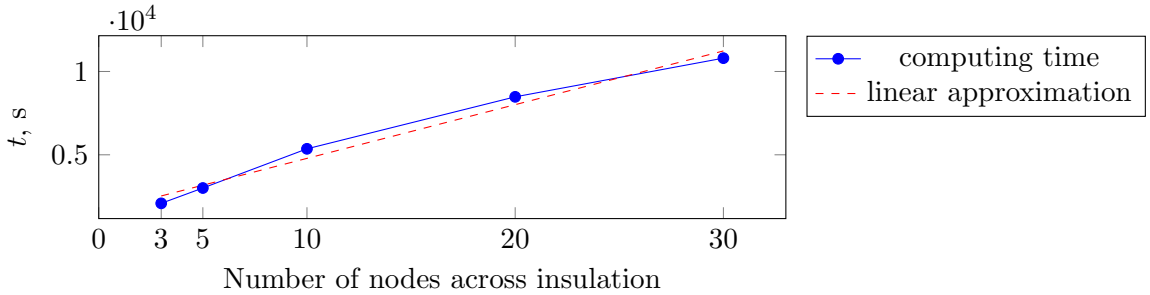

Figure 7.13: Computing time vs. number of insulation elements.

[2]The analysis was performed on the following calculation unit: Intel(R) Xeon(R) CPU E5-2667 V4 @ 3.20 GHz (2 processors) with RAM 128 GB.

7.4.2 Analysis with Quench Velocity-Based Approach

Similarly to the analysis of a bare strand employing the quench velocity-based approach, the electro-thermal element LINK68 is used to model a strand. The insulation with epoxy resin is represented by the LINK33 element since no heat is generated in that region. In order to observe the relevant evolution of a relative error in resistive voltage and hot-spot temperature, the total simulation time is extended to $t = 0.5$ s. The study of a 1.5 metre-long cable is performed with a varying longitudinal mesh size, $m = \{1, 10, 20\}$ mm. The cable is longer than in the analysis of a bare strand so that the quench front could propagate during the entire simulation time without reaching the end of the domain. The time step range in the quench velocity-based approach is $t_{\text{step range}} = [0.1, 1]$ ms as deduced in Section 7.3.2. The geometric assumptions, the initial conditions, and the analysis settings are identical with those discussed in Section 3.5. The parameters related to the co-simulation are shown in Table 7.7.

Table 7.7: Input parameters in the quench velocity-based approach.

parameter	value	unit
t_{com}	2.5	[ms]
$v_{\text{quench, average}}$	2.18	[m/s]
t_{total}	0.5	[s]

Fig. 7.14 compares the temperature distribution along the strand with the mesh size of 20 mm between the standard analysis and the quench velocity-based approach. The quench front remains behind the reference solution in analogy to the case of a bare strand, as presented in Fig. 7.7.

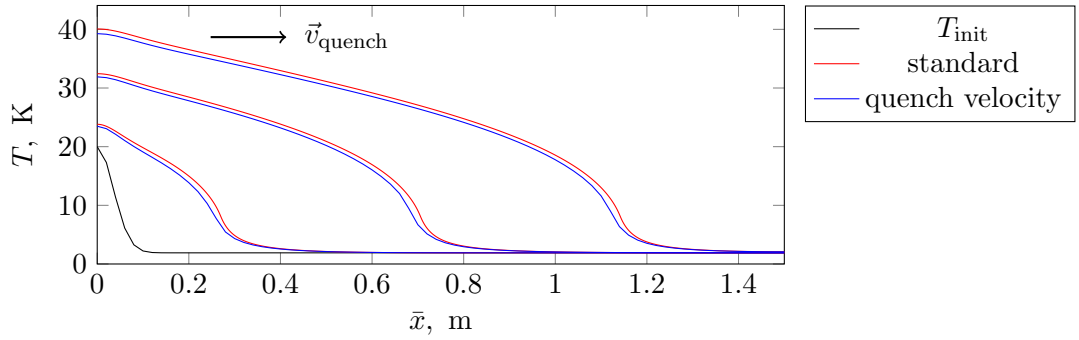

Figure 7.14: Temperature distribution with a specified direction of quench velocity, $\vec{v}_{\text{quench}}$ of the reference solution and the quench velocity-based approach with the longitudinal mesh size of 20 mm and the mesh size of 8 µm (20 nodes) across the insulation layer at three time steps: $t = \{0.1, 0.3, 0.5\}$ s.

As presented in Fig. 7.15, the resistive voltage rises monotonically in the quench velocity-based approach and follows the reference solution similarly to the example of a bare strand.

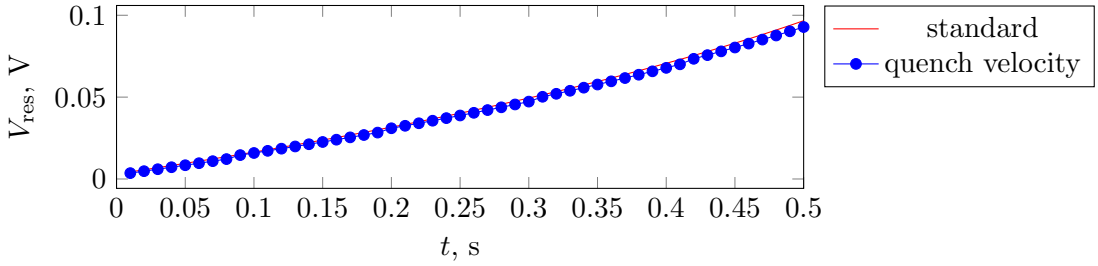

Figure 7.15: Comparison of the resistive voltage between the reference solution and the quench velocity-based approach with the longitudinal mesh size of 20 mm.

Halving the mesh size from 20 to 10 mm results in a better convergence of the resistive voltage in the quench velocity-based approach with respect to the reference solution, as presented in Fig. 7.16. For the mesh size of 10 mm, the relative error remains in the range of -10% and converges to -2.5% as the quench propagates. With a coarser mesh of 20 mm, the relative error converges to the approximate value of -5%. It is also worth mentioning, that the decrease of the mesh size to 1 mm does not decrease the relative error with respect to the solution with a larger mesh size. The mesh size of 1 mm corresponds to the case of a reference standard solution. Similarly to the example without the insulation layer, one has to accept the difference between two approaches.

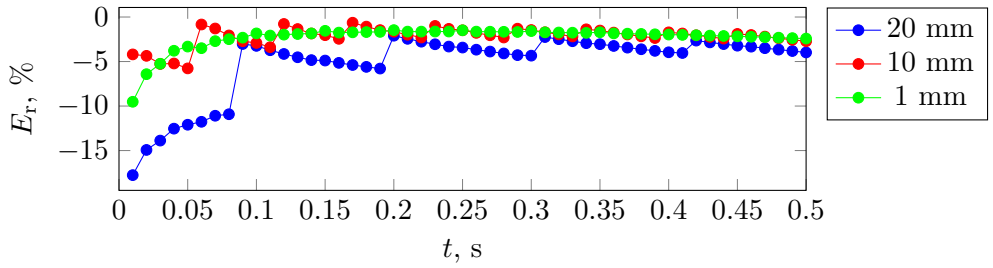

Figure 7.16: Relative error of the resistive voltage for the varying mesh size along the strand in the quench velocity-based approach.

Figure 7.17 presents the evolution of the relative error with respect to the hot-spot temperature. The error stabilises at approximately $t = 0.2$ s. After that moment, the error starts diverging with an approximate rate of -2%/s, assuming a linear change of error over time. The divergence occurs for every mesh size. At $t = 0.5$ s, the result for a mesh size of 20 mm is characterised by a relative error 0.5% lower with respect to higher mesh densities.

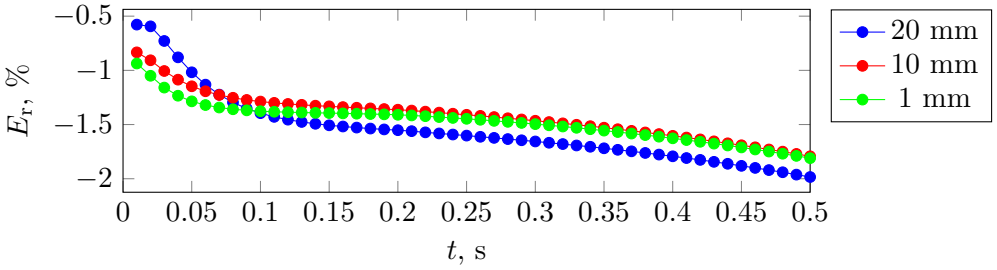

Figure 7.17: Relative error of the hot-spot temperature for the varying mesh size along the strand used for the quench velocity-based approach.

7.4.3 Conclusion

Table 7.8 summarises different improvement rates when the quench velocity-based approach is applied with respect to the standard analysis. It is possible to increase the time step range and increase the mesh size with the quench velocity-based approach. Unlike the case of a bare strand, shortening the computation time is clearly visible. The mesh relaxation along the strand also reduces the total number of nodes used to model the zone outside of the composite strand. The evolution of the nodal size of the model can be easily deduced from (3.12). In fact, the reduction of the insulation elements is one of the main reasons for accelerating the computing speed, the gain of which is higher than the loss of time due to

the data exchange during the co-simulation process[3].

Table 7.8: Benchmark summary of the analysis with insulation and epoxy resin.

	mesh size, mm	$t_{\text{step range}}$, ms	computing time, s
standard analysis	1	[0.01, 0.1]	52740
quench velocity-based approach	10	[0.1, 1]	9480
quench velocity-based approach	20	[0.1, 1]	5940
improvement rate	10-20 times	10 times	5.5-8.9 times

Table 7.9, summarises the evolution of the relative error in time with respect to the resistive voltage and the hot-spot temperature. One can observe, that the relative error of the resistive voltage converges to -4% for the largest mesh size of 20 mm. The smaller mesh relaxation leads to the lower relative error of the resistive voltage. The relative error corresponding to the hot-spot temperature increases with time independently of the mesh size. It reaches the level of approximately -2% after $t = 0.5$ s independently of the mesh size.

Table 7.9: Comparison of relative error for resistive voltage and hot-spot temperature.

mesh size, mm	V_{res}		$T_{\text{hot-spot}}$	
	$t = 0.01$ s	$t = 0.5$ s	$t = 0.01$ s	$t = 0.5$ s
1	-9.52%	-2.42%	-0.94%	-1.81%
10	-4.20%	-2.72%	-0.83%	-1.79%
20	-17.76%	-3.98%	-0.58%	-1.98%

7.5 General Remarks

The quench velocity-based approach is a promising tool when a simulation of a large numerical domain is conducted exceeding 100 000 degrees of freedom for a standard solution. If this method is used, a certain error should be assumed concerning the resistive voltage and the hot-spot temperature (both evolve in time). In the benchmarking process presented in this chapter, the analysis settings for a standard solution are chosen to remain in an assumed relative error of 1% with respect to an average quench velocity. In the comparison of two methodologies, one can conclude that the quench velocity-based approach results in

[3]The analysis was performed on the following calculation unit: Intel(R) Xeon(R) CPU E5-2667 V4 @ 3.20 GHz (2 processors) with RAM 128 GB.

an underestimation of the quench front position with respect to the standard simulation performed in ANSYS.

While simulating a multi-strand case with the quench velocity-based approach, one should bear in mind, that the results will be less precise with respect to a standard solution. An estimation of the relative error is presented in Table 7.10 for the case of a bare strand. The tolerances are based on relative errors at $t = 0.01$ s of the corresponding parameters as well as the final relative errors to which the parameters converged to at $t = 0.3$ s. As the convergence of all errors is clearly visible in this example, it is assumed that they do not change in further time steps of the performed simulation. However, the convergence cannot be assured for higher values of current and magnetic field strength which were not studied. Therefore, this estimation should be carried out case by case.

Table 7.10: Tolerance range in the quench velocity-based approach for the case of a bare strand.

mesh size strand, mm	$t_{\text{step range}}$, ms	tolerance, E_{r}	
		V_{res}	$T_{\text{hot-spot}}$
10	[0.1, 1]	-15%	$> -5\%$
20	[0.1, 1]	-20%	$> -5\%$

The results obtained from the analysis of a strand with insulation and epoxy resin are shown in Table 7.11. Similarly to the case of a bare strand, the tolerances are based on relative errors at $t = 0.01$ s of the corresponding parameters as well as the final relative errors to which the parameters converged to at $t = 0.5$ s. However, the relative error of the hot-spot temperature does not converge to one specific value. Thus, it is assumed that the error corresponding to the hot-spot temperature increases by -2% per second during the simulation by assuming a linear divergence from the standard solution.

Table 7.11: Tolerance range in a quench velocity-based approach with insulation.

mesh size		$t_{\text{step range}}$, ms	tolerance, E_{r}	
strand, mm	insulation, µm		V_{res}	$T_{\text{hot-spot}}$
10	8	[0.1, 1]	-5%	-1.5% + (-2%/s)
20	8	[0.1, 1]	-20%	-1.5% + (-2%/s)

It is expected that the lack of convergence of the hot-spot temperature has an influence on the evolution of the resistive voltage being a function of temperature due to the resistivity of copper. Therefore, two relative errors can be a coupled problem in a longer simulation

lasting several seconds. It is recommended to include the correction of the relative error as a function of time in the case of the resistive voltage as well. However, since the visible divergence does not occur in this study, the given step is omitted in this thesis.

Chapter 8

Skew Quadrupole Quench Analysis

8.1 Motivation

As demonstrated in the previous chapter, the quench velocity-based approach results in a faster solution of a one-dimensional thermal quench propagation with insulation and epoxy resin while introducing an error in nodal temperature and resistive voltage. This chapter aims at validating the quench velocity-based approach by using the real case of a skew quadrupole magnet. As described in Section 3.3, the skew quadrupole is one of the high-order corrector magnets developed within the scope of the HL-LHC project. This magnet was selected for two reasons for the analysis in this thesis:

1. The skew quadrupole was tested in LASA laboratories. The resistive voltage and current drop were measured during a provoked quench of the magnet. Therefore, with the quench velocity-based approach, one can simulate a real magnet case and validate the results against available measurements.

2. The geometry of the skew quadrupole is similar to other high-order corrector magnets which are self-protected. One has to remember that the skew quadrupole does have an energy extraction system, i.e. it is not self-protected. However, the analysis based on the skew quadrupole will allow for examining other self-protected corrector magnets in the future. A 3D thermal study is required for a self-protectability analysis. Such studies reduce the safety margin of magnet design parameters.

8.2 Skew Quadrupole Design

Most of the data about the skew quadrupole is presented in Section 3.3. Figure 8.1 shows the transversal cross-section of one coil of the skew quadrupole. The coil is enclosed

within an area of 24.5x27.3 mm covered with a 1 mm-thick ground insulation layer. The ground insulation is made of BT-S2, which is a mixture of resin and S2-glass provided by the Arisava company [33]. The ground insulation on the internal side of a coil is thinner (0.15 mm).

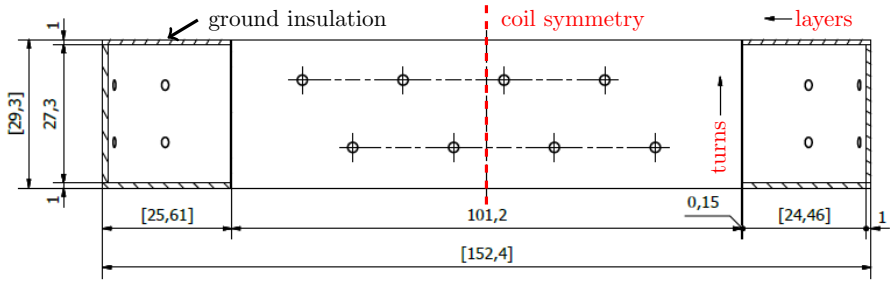

Figure 8.1: The transversal cross-section of one coil of a skew quadrupole [24].

The remaining dimensions of the coil are shown in its top view in Fig. 8.2.

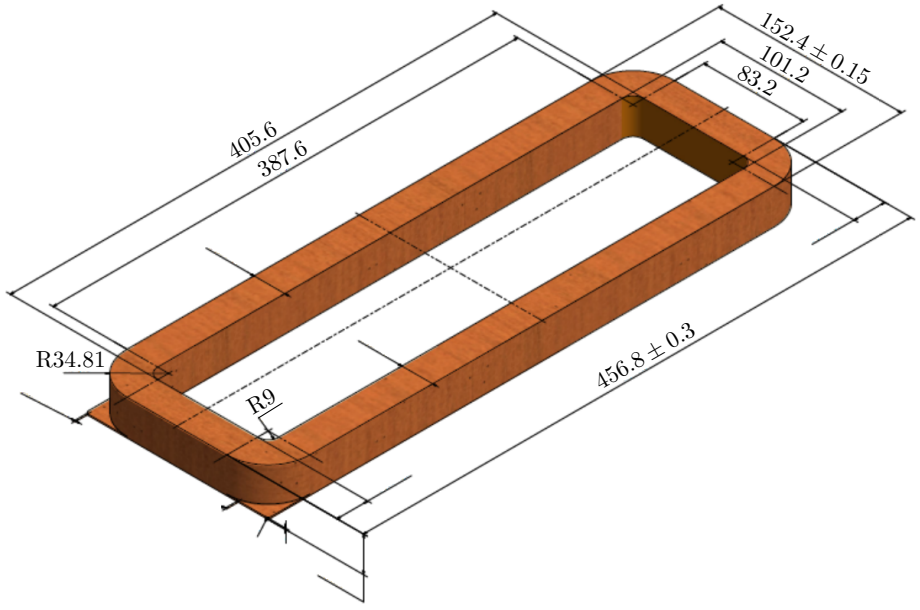

Figure 8.2: Top view of a skew quadrupole wrapped with the ground insulation [24].

As shown in Fig. 8.3, each coil has 29 turns in each of its 26 layers. In total, the number of windings per coil is 754 [22].

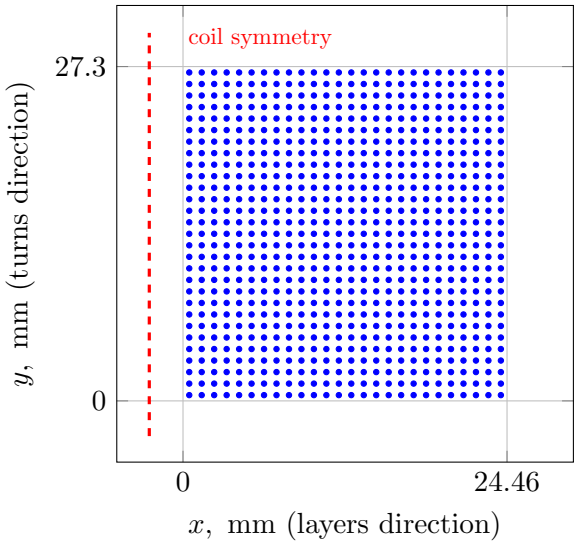

Figure 8.3: Location of the windings in the cross-section of a half-coil.

The winding scheme is shown in Fig. 8.4. Winding 1 is placed at the bottom left of the 2D cross-section. The last winding number 754 is placed further in the x-direction. The 2$^{\text{nd}}$ half of the coil is a mirror image of the presented winding scheme. [23]

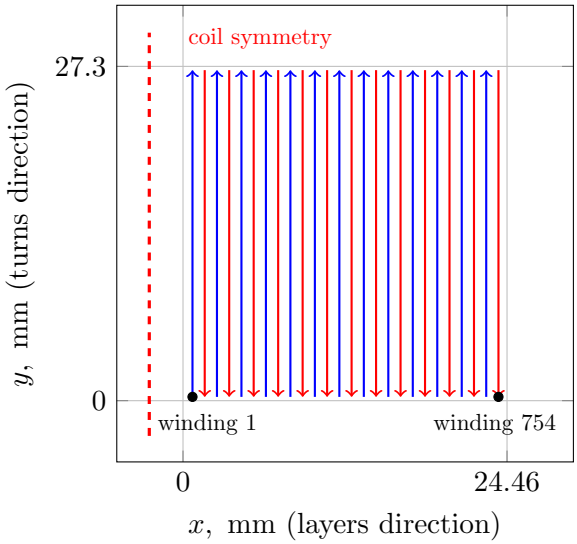

Figure 8.4: Winding scheme of a single coil of the skew quadrupole.

Unlike in the analyses with a single strand presented in previous chapters, when a multi-strand thermal simulation of a magnet is considered, it is important to specify its winding

scheme because the windings interact with each other thermally across the insulation layer. The general parameters of the skew quadrupole are summarised in Table 8.1. Four coils of the magnet are connected in series and powered by a single power converter. The coil length, being a quarter of the total strand length of the entire magnet, is more than 812 m. This is a very large number for thermal quench simulations, as mentioned in Section 3.6. The operating current of the skew quadrupole is $I = 174$ A.

Table 8.1: Geometrical parameters of the skew quadrupole [22, 23].

parameter	value	unit
number of circuits	1	[-]
L_{coil}	812	[m]
$I_{\text{operating}}$	174	[A]

8.3 Quench Measurements

During the measurements, the spot heater was attached to the external ground insulation of one of the four coils of the magnet. The heater contact area of 4 mm^2 was centered at the 13$^{\text{th}}$ layer on the aperture side of the magnet, as presented in Fig. 8.5.

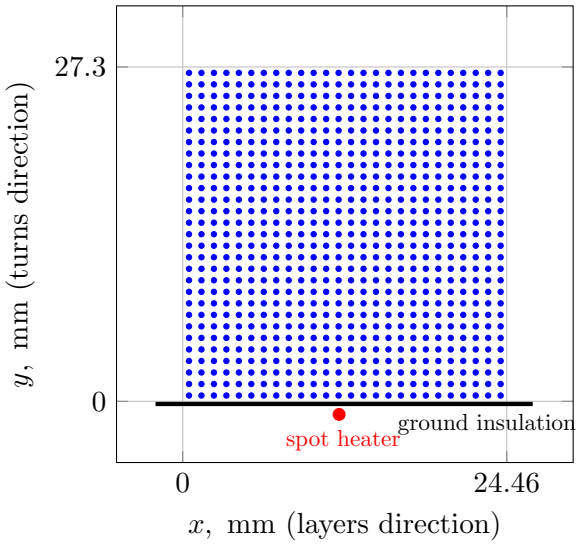

Figure 8.5: Location of the spot heater with respect to the coil cross-section [23].

The coil is quenched by the heat generated in the spot-heater. The heat is a result of a current discharge in the 1$^{\text{st}}$ order RC-circuit as shown in Fig. 8.6. The results from the

measurements include the capacitance of $C = 14$ mF and the voltage accross the capacitor over time.

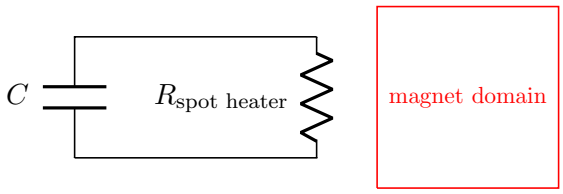

Figure 8.6: Schematic of heating the magnet by means of a spot heater.

The spot heater resistance is calculated as

$$R_{\text{spot heater}} = \frac{\tau}{C}, \tag{8.1}$$

where τ – time constant, C – capacitance. The electric energy of the capacitor is directly dissipated in the spot-heater. Therefore, the power dissipation at the spot heater is calculated as

$$P(t)_{\text{spot heater}} = -P(t)_{\text{capacitor}} = -\frac{1}{\text{dt}}(E_{\text{C}}) = -\frac{1}{\text{dt}}(\frac{1}{2}\text{C}V_{\text{C}}^2), \tag{8.2}$$

where V_{C} – voltage across the capacitor, E_{C} – energy stored in the capacitor, $P(t)$ – power being a function of time. Figure 8.7 presents the measured drop of a capacitive voltage while the capacitor was discharging. Figure 8.7 also presents the deduced power dissipation at the spot heater based on (8.2). The sampling frequency of the measurements was equal to $f = 5000$ Hz. Based on the power curve in Fig. 8.7, most of energy was transmitted to the coil during the first $t = 5$ ms and the capacitor was fully discharged at $t = 10$ ms. The coil's bath temperature was $T_0 = 4.3$ K during the measurements.

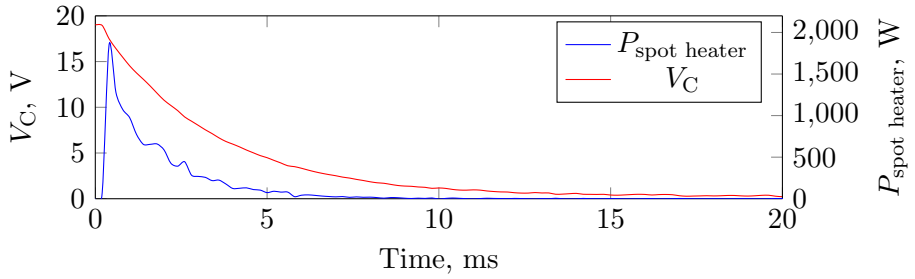

Figure 8.7: Voltage drop across the capacitor and heating power curve at the spot heater.

With the voltage drop shape, the time constant and spot heater resistance were deduced, as shown in Table 8.2. Moreover, the entire capacitive energy deposited in the magnet was approximately equal to $E_{\text{C}} = 2.5$ J.

Table 8.2: RC heating circuit characteristics.

parameter	value	unit
τ	3.5	[ms]
$R_{\text{spot heater}}$	0.25	[Ω]
E_{C}	2.5	[J]

The electrical circuit schematic of the skew quadrupole is presented in Fig. 8.8. Each coil is characterised by the same self-inductance L_{coil}. Since all coils are connected in series, it is possible to include their mutual inductance in the self-inductance. The heat was not transmitted to other coils of the skew quadrupole during the measurements, i.e. only one coil out of four quenched. R_{coil} in Fig. 8.8 represents the internal resistance of the coil where the spot heater was placed. When the skew quadrupole is in the superconducting state, $R_{\text{coil}} = 0$. After the quench is provoked by depositing heat through the spot heater, R_{coil} represented by a varying resistance starts to increase. In order to detect a quench, the voltages V_1 and V_2 are compared. If their difference exceeds a threshold value, $V_{\text{th}} = 0.2$ V during a certain period of time, the quench is detected. The time, after which the quench is detected after exceeding the threshold V_{th}, is referred to as the validation time, and equals $t_{\text{validation}} = 0.20$ ms. When the quench is detected, the switch s_{power} closes and cuts off the power converter whereas the switch s_{dump} opens leading to the discharge of the energy stored in the circuit in the dump resistor R_{dump}. The schematic in Fig. 8.8 neither includes the protection devices of the energy extraction system nor those of the power converter as they do not affect the quench protection study in this thesis.

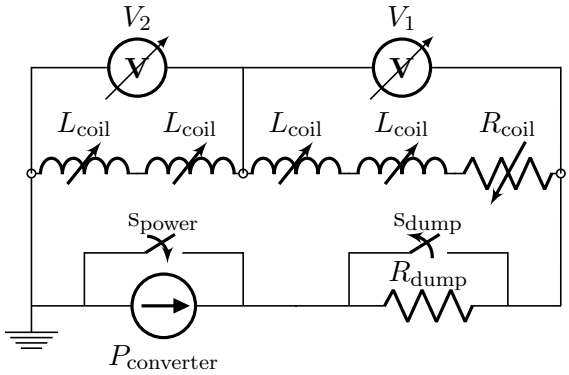

Figure 8.8: Circuit schematic of the skew quadrupole with marked directions of closing/opening switches when a quench is detected.

After the quench is detected, the circuit is represented, as shown in Fig. 8.9. The total

magnet inductance L_{magnet} is calculated as [23]

$$L_{\text{magnet}} = 4L_{\text{coil}} + 8M_{\text{a}} + 4M_{\text{b}}, \tag{8.3}$$

where M_{a} – mutual inductance between adjacent coils, M_{b} – mutual inductance between opposite coils. The initial discharge current I_0 is equal to the operating current of the magnet during the measurements. If the magnet is self-protected, $R_{\text{dump}} = 0$. The dump resistance of the energy extraction system equals $R_{\text{dump}} = 1.557\ \Omega$ [23].

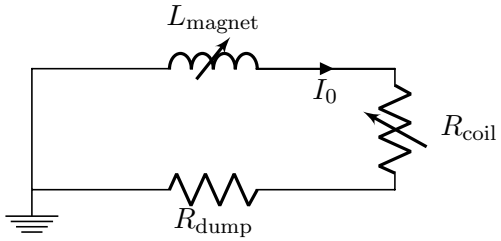

Figure 8.9: Simplified circuit scheme of the skew quadrupole after the quench is detected.

The quench measurements were conducted for $I = 86$ A, which is a lower value than the designed nominal operating current of the skew quadrupole. As presented in Fig. 8.10, the quench resistive voltage reaches the threshold value for the quench detection system at $t = 130$ ms. The quench is detected at $t = 150$ ms. [23]

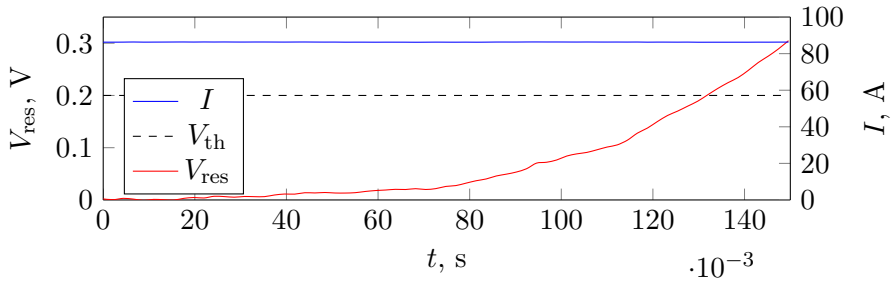

Figure 8.10: Resistive voltage rise as a function of time at constant current before the quench detection [23].

Figure 8.11 presents the current profile as well as the change of resistive voltage across the magnet during the discharge of the skew quadrupole. The resistive voltage V_{res} is a function of coil resistance R_{coil} and the value of current in the circuit. Until $t = 1.5$ s, a rise of voltage up to approximately 16 V is observed due to the increase of coil resistance, mainly because of a turn-to-turn propagation. After this time, the voltage drops down because the current decreases rapidly in the circuit. The smooth shape of the current curve

indicates that the quench back did not occur during the discharge which can be explained by a relatively long discharge time. The quench measurements ended at $t = 9$ s.

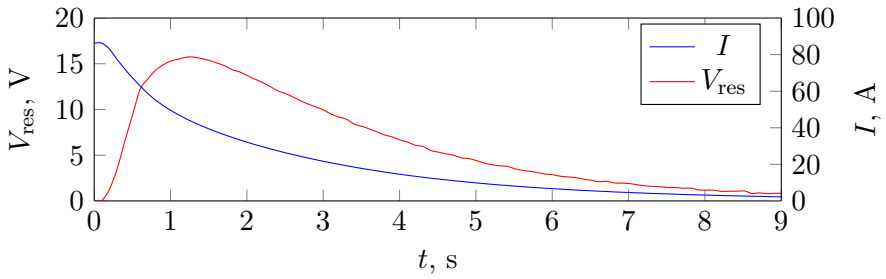

Figure 8.11: Current and Resistive Voltage change during the discharge of skew quadrupole [23].

8.4 Magnetic Map of Skew Quadrupole

8.4.1 Material Properties and Geometry

Figure 8.12 presents a cross-section of the skew quadrupole used for a magnetic analysis. The iron yoke, the coil, and the remaining parts of the model considered as air are marked in red, blue, and violet, respectively. The magnet cross-section has three axes of symmetry. Therefore, only one-eighth of the skew quadrupole is simulated.

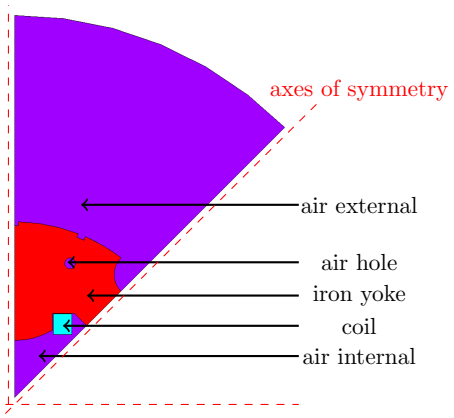

Figure 8.12: The 8$^{\text{th}}$ part of the skew quadrupole cross-section with specified component names.

The coil cross-section is homogenised magnetically and electrically, i.e. there is no distinction between the insulation and the strand composite. In the magnetic analysis of the skew quadrupole, it is assumed that the magnet is infinitely long. This assumption allows for considering this case as a plane 2D problem. The areas considered as air and

the coil are characterised by a magnetic relative permeability equal to one. The iron yoke is characterised by a nonlinear B-H curve, as presented in Fig. 8.13.

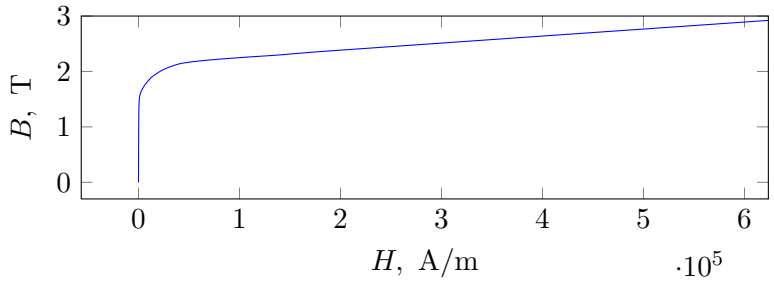

Figure 8.13: B-H curve of an iron yoke of the skew quadrupole.

8.4.2 Mesh, Initial and Boundary Conditions

The meshed geometry is illustrated in Fig 8.14. The element PLANE233 is used for the analysis. It is a 2-D element for simulating 2D planar and axisymmetric magnetic fields. The element is defined by 8 nodes [26].

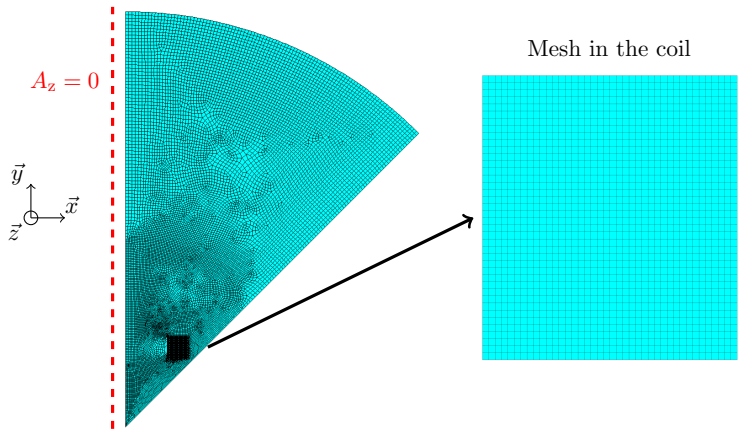

Figure 8.14: The 8^{th} part of the skew quadrupole cross-section.

For the requirements of the given analysis, the only degrees of freedom related to the magnetic vector potential A_z are used. In order to consider the symmetry condition, the magnetic vector potential A_z is set to zero on the left side of the geometry. This means that the directional magnetic field B_x is equal to zero on the left side marked in red. On the other side of the symmetry axis, it is assumed by default in ANSYS APDL, that the magnetic flux is perpendicular to the wall. As illustrated in Fig 8.14, the mesh is refined

inside of the coil. There are 40 elements placed on each side of a rectangular area. The element size in the remaining components of the model is depicted in Table 8.3.

Table 8.3: Mesh size characteristics of the skew quadrupole.

area name	element size	unit
iron yoke	2	mm
air internal	1.5	mm
air external	4	mm
air hole	2	mm

A uniformly distributed current density is applied to the coil area calculated as

$$J_{\text{coil}} = \frac{n_{\text{windings}}\, I}{A_{\text{coil}}}, \tag{8.4}$$

where n_{windings} – number of windings, I – current in each winding, A_{coil} – cross-sectional area of the coil with the resin and insulation. 86 static analyses are conducted with current linearly rising from 1 up to 86 A.

8.4.3 Results and Magnetic Field Mapping in the Skew Quadrupole

Two values of current are chosen for the comparison of the magnetic field strength distribution inside the cross-section of the skew quadrupole, as shown in Fig. 8.15. As expected, one can observe that the magnetic field strength increases with the rise of current. Moreover, the difference in the magnetic field strength distribution at different current levels is a result of the nonlinear magnetic properties of the iron yoke.

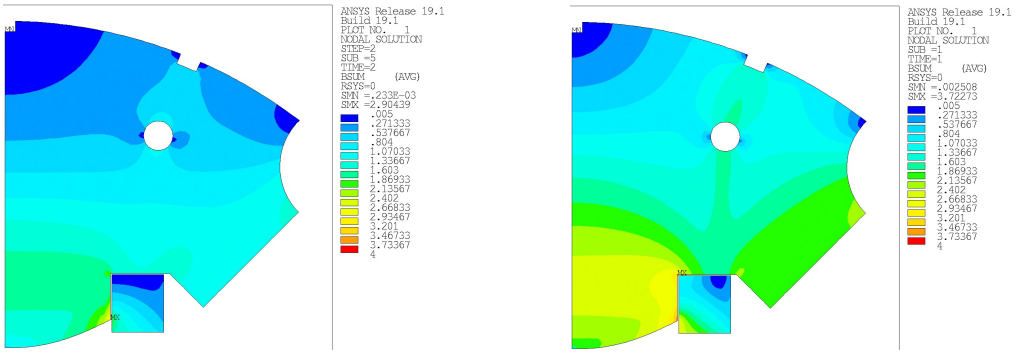

Figure 8.15: Magnetic field strength distribution in the coil and the iron yoke for $I = 40$ A (left) and $I = 86$ A (right).

Figure 8.16 presents the magnetic field strength distribution in the coil for two values of transport current. In order to use the results from the coil for the quench velocity-based approach, the vector sum of the magnetic flux density is extracted at corner nodes of the coil elements. The magnetic field is linearly interpolated in the (x, y)-space for the current levels between the simulated values.

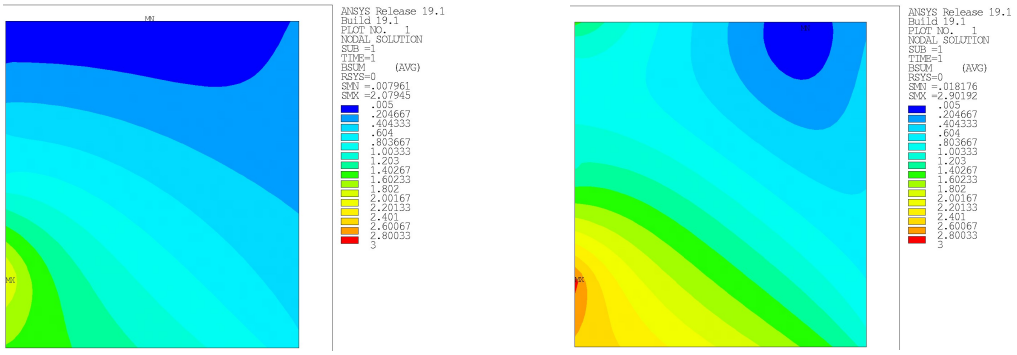

Figure 8.16: Magnetic field distribution in the coil for $I = 40$ A (left) and $I = 86$ A (right).

Figure 8.17 illustrates the magnetic field map applied to 754 windings for the nominal current of $I = 86$ A. The central position of each winding is used to interpolate the magnetic field strength from the ANSYS results presented in Fig. 8.16. As current changes in time during the discharge process, the magnetic field strength of each winding varies with its position in the (x, y)-space.

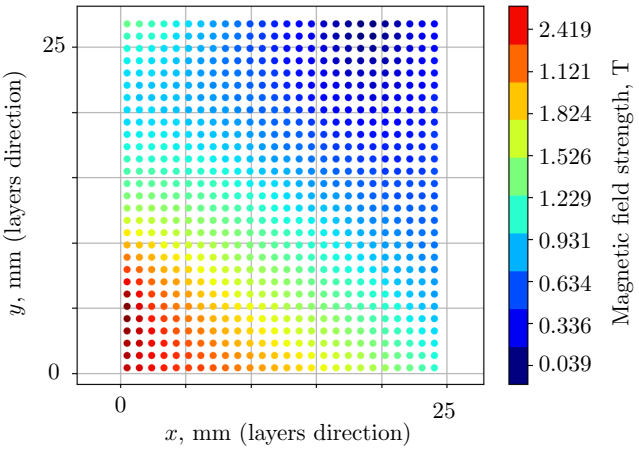

Figure 8.17: Magnetic field strength distribution in the coil at $I = 86$ A applied separately to each winding.

It is important to emphasise that ANSYS APDL allows for creating the material properties only as a function of a single-dependent variable (in this case temperature). It is not possible to include the dependence of material properties on the magnetic field strength in a relatively simple manner in this tool. The only possible method to add an additional variable to the repository of the material properties is to replace the repository at a certain moment of a transient simulation. The update of material properties is conducted at the communication time windows t_{com} at which Python communicates with ANSYS to update the quench position in the coil. Since the update of the material properties is a time-consuming process, it is not recommended to perform it at each communication time window t_{com} when hundreds of windings are modelled. It is rather recommended to update the material properties when the current drops by the assumed threshold value of current.

8.5 Quench Velocity Map of Skew Quadrupole

In order to conduct the multi-strand analysis with the quench velocity-based approach, one should estimate the quench velocity for varying values of magnetic field strength and current that are reached in the magnet. In the previous section, it was concluded that the maximum magnetic field strength in the skew quadrupole equals $B = 3$ T and occurs at the initial value of current, $I = 86$ A. Therefore, two following main parameters, that have a direct impact on the quench velocity are studied in this section: (i) magnetic field strength, (ii) transport current. In order to conduct the analyses, the standard ANSYS simulations are performed based on 1D models of a relatively short length.

8.5.1 Update of Skew Quadrupole Geometry

After preliminary study of the quench propagation in the modelled coil, the modelling of an insulated strand is reformulated in this chapter. The new model aims at parametrising the influence of resin on the heat capacity of the coil. The actual volume of resin between the windings remains an open question from the design standpoint. In the fabrication process, the insulated strand is wound first and the resin is added later to the pre-assembled coil. Therefore, including the resin volume in LINK33 elements is a too conservative assumption leading to a relatively slow turn-to-turn propagation. Therefore, the heat capacity of resin is optionally added with MASS71 elements.

Figure 8.18 presents a new 1D+1D model representing a three-dimensional strand. In the right picture, the nodes marked in red are modelled with the LINK33 element and the strand marked in yellow – with LINK68. The blue points represent MASS71 elements including an additional heat capacity in the strand. MASS71 is a point element with temperature

as a degree of freedom characterised by a negligible thermal resistance [26]. LINK68 has the material properties of the composite strand whereas both LINK33 and MASS71 – of G10. The main reason for a change is that, in reality, the external insulation layers are in direct contact (marked in red in the left picture of Fig. 8.18) omitting the resin (marked in blue in the left picture of Fig. 8.18).

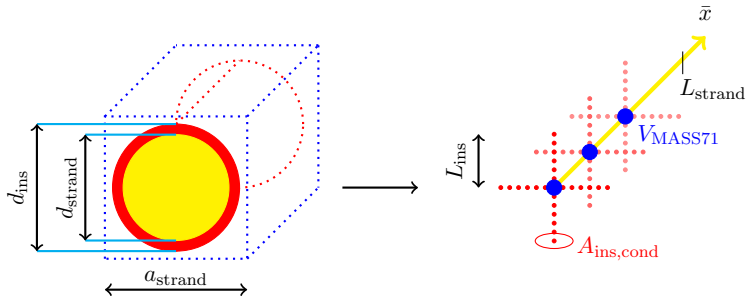

Figure 8.18: 3-dimensional transformation of a 1D+1D strand domain in ANSYS.

This time, the insulation cross-sectional area A_{ins} only takes into account the external insulation layer marked in red as

$$A_{\text{ins}} = \frac{\pi}{4}\left(d_{\text{ins}}^2 - d_{\text{strand}}^2\right), \tag{8.5}$$

where d_{ins} – strand diameter with insulation, d_{strand} – strand diameter. Then, the average perimeter is calculated as

$$p_{\text{avg}} = \frac{\pi}{2}\left(d_{\text{ins}} + d_{\text{strand}}\right). \tag{8.6}$$

By combining (8.5) and (8.6) with (3.10) one can obtain an equivalent insulation length L_{ins}. The conduction area $A_{\text{ins, cond}}$ is calculated as

$$A_{\text{ins, cond}} = \frac{1}{4}\,\frac{A_{\text{ins}}\,L_{winding}}{L_{\text{ins}}}\,\frac{1}{n_{\text{divisions, winding}}}\,u_{\text{ins}}, \tag{8.7}$$

where $n_{\text{divisions, winding}}$ – number of divisions applied along the winding, u_{ins} – coefficient of the insulation area, L_{winding} – straight winding length of the coil calculated as

$$L_{\text{winding}} = 2\,(d + e), \tag{8.8}$$

where the parameters d and e are presented in Fig. 8.19. The curvature lengths are not taken into account because they vary with an increasing number of a winding layer.

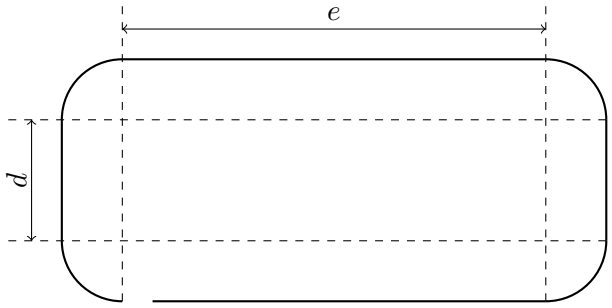

Figure 8.19: Top view of one winding of a high-order corrector geometry.

One can observe that Equation (8.7) is similar to (3.11) but without considering u_{ins}. If $u_{\text{ins}} = 1$, the entire insulation volume is packed in the LINK33 elements. If $u_{\text{ins}} < 1$, the remainder of the insulation volume is added to the volume of point MASS71 elements. A reduction of u_{ins} serves for limiting the contact area between the neighbouring wires and, thus, the heat transfer across the insulation. The volume of a point MASS71 element is calculated as

$$V_{\text{MASS71}} = \frac{\left(a^2 - \frac{\pi d_{\text{ins}}^2}{4}\right) L_{\text{winding}}}{n_{\text{nodes, winding}}} \, u_{\text{resin}} + L_{\text{ins}} \; A_{\text{ins, cond}} \; (1 - u_{\text{ins}}) \,, \tag{8.9}$$

where u_{resin} – resin packing coefficient. If $u_{\text{resin}} = 0$, no resin is taken into account, whereas if $u_{\text{resin}} = 1$, the entire resin volume is modelled.

8.5.2 Quench Velocity Map

In this section, a set of 16 standard ANSYS simulations with varying magnetic field strength and current is conducted. As presented in Table 8.4, a 50 cm-long winding is studied. It is important to mention that the winding length L_{winding} refers to a straight strand section from Fig. 3.2 defined as L_{strand}. The model simulates a strand with full insulation without resin. Therefore, $u_{\text{ins}} = 1$ and $u_{\text{resin}} = 0$. The number of nodes in the longitudinal direction equals $n_{\text{nodes, winding}} = 500$. That corresponds to a mesh size of 1 mm. 20 nodes across the insulation layer corresponds to a transverse mesh size of 3.5 µm. The mesh size in all dimensions is based on the reference solution of a standard ANSYS simulation with insulation and epoxy resin deduced in Section 7.4.1. The mesh size of the insulation elements is smaller in this section, because no resin is considered. Therefore, the model is even more accurate than the one discussed in Section 7.4.1.

Table 8.4: Geometry input parameters.

parameter	value	unit
L_{winding}	0.5	[m]
L_{ins}	0.07	[mm]
u_{ins}	1.0	[-]
u_{resin}	0.0	[-]
$n_{\text{nodes, winding}}$	500	[-]
$n_{\text{nodes, insulation}}$	20	[-]

Table 8.5 depicts input parameters for the quench velocity study. The initial temperature profile presented in Fig. 8.20 is calculated based on (3.5). The initial temperature T_{init} corresponds to the bath temperature of the measured magnet.

Table 8.5: Parameters for the quench velocity study.

parameter	value	unit
I	26, 46, 66, 86	[A]
B	0, 1, 2, 3	[T]
T_{init}	4.3	[K]
T_{max}	20.0	[K]
$L_{\text{quench, init}}$	0.1	[m]
α	0.223	[m]
t_{total}	0.1	[s]
$t_{\text{step range}}$	$[0.01, 0.1]$	[ms]

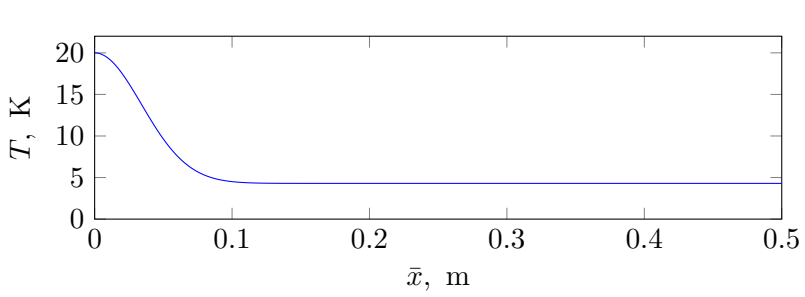

Figure 8.20: Initial Gaussian temperature profile.

In the first instants of the quench analysis, the quench velocity is usually higher. However, there is a moment in the simulations when the heat generation predominates the

longitudinal heat propagation in the quenched zone. At that moment, the quench velocity stabilises and remains relatively constant.

The quench velocity is calculated as in (7.2) with a time window equal to $\Delta t = 10$ ms. If the difference in the quench velocity between two consecutive time windows varies less than 0.1 m/s, it is assumed that the quench propagation stabilises. As soon as this condition is met, the analysis finishes and the final value of the quench velocity is saved. Figure 8.21 presents the quench velocity map as a function of the magnetic field strength and current.

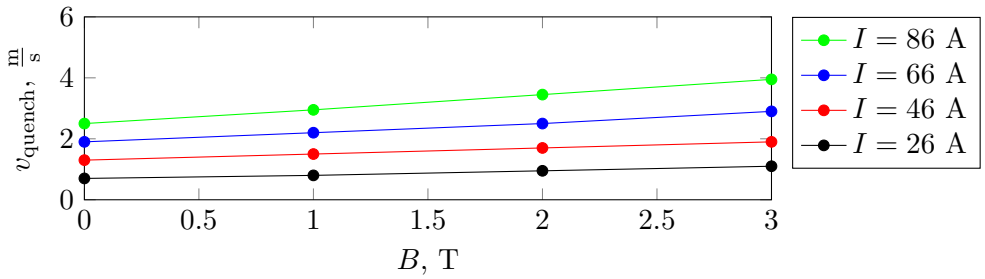

Figure 8.21: Quench velocity map as a function of magnetic field strength and current.

8.6 Minimum Propagation Zone Analysis

8.6.1 Motivation

The quench simulation of the skew quadrupole predicts the evolution of design quantities for the operation of a magnet once the minimum propagation zone is already transitioned. Therefore, the heat transfer from the spot heater to the coil described in Section 8.3 is not modelled in this thesis. The reasons for this are as follows:

1. The spot heater is placed outside of the ground insulation. The distance between the spot heater and the winding equal to more than a millimetre requires an accurate simulation of the heat transfer in that zone.

2. The ground insulation, is made of a material different from G10. Assuming that the material properties of the ground insulation are identical to G10, is an additional approximation in the model.

3. The heat transfer from the spot heater to the coil is affected by additional parameters including thermal properties of helium bath and methods of gluing the heater to the external side of the ground insulation.

8.6.2 Simulation

In order to find the minimum propagating zone, a 1D standard ANSYS analysis is conducted with insulation and no resin. The geometric parameters of the model are identical with those described in Table 8.4. The remaining parameters are summarised in Table 8.6. The analysis is only performed for an operating current of $I = 86$ A, at which the quench is initiated. The initially quenched winding is presented in Fig. 8.5. The magnetic field strength in the simulation corresponds to the interpolated value of the initially quenched winding in Fig. 8.17. The maximum temperature $T_{\max}$ is chosen to be close to the critical temperature of the strand approximately equal to $T_c \approx 9$ K.

Table 8.6: Input parameters for boundary and initial conditions.

parameter	value	unit
I	86	[A]
$B(I = 86\ \text{A})$	1.962	[T]
T_{init}	4.3	[K]
$T_{\max}$	10.0	[K]
α	0.223	[m]
t_{total}	0.1	[s]
$t_{\text{step range}}$	$[0.01, 0.1]$	[ms]

With an accuracy of less than 5 mm, it is deduced, that the minimum propagation zone is equal to $L_{\text{quench, init}} = 20$ mm. The temperature profile of the final result is shown in Fig. 8.22. As depicted in Table 8.7, the initial energy deposition in the strand (without taking into the insulation layer account) is less than one Joule. It is a relatively small value with respect to the total energy deposited by the capacitor accounting for approximately $E_C = 2.5$ J (see Table 8.2).

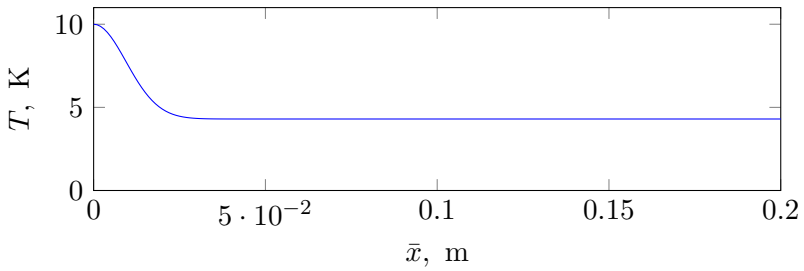

Figure 8.22: Initial Gaussian temperature profile.

Table 8.7: Results for MPZ analysis.

parameter	value	unit
$E_{\text{init, winding}}$	15	[mJ]
$L_{\text{quench, init}}$	0.02	[m]

8.7 Quench Detection Analysis

The quench simulation of the skew quadrupole is divided into two parts: (*i*) analysis before the quench detection, when the current is constant, (*ii*) analysis after the quench detection, when the current starts varying over time until the energy stored in the magnetic field is fully discharged. This section covers the simulation of the first case.

8.7.1 Mesh and Geometry

Fig. 8.23 shows the top view of a meshed ANSYS geometry. It consists of 754 insulated windings. In total, the coil is nearly 810 m-long. It is shorter than 812 m, which is the real magnet length, because of sharp edges of the rounded coil parts.

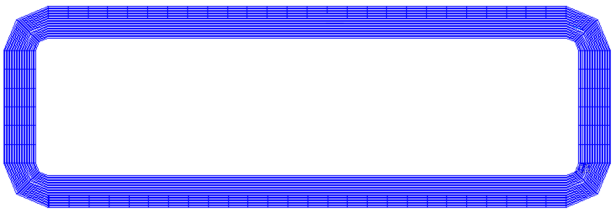

Figure 8.23: ANSYS meshed geometry of the skew quadrupole - top view.

Parameters of the meshed geometry are summarised in Table 8.8. They are based on the discussion in Section 8.5.1. The insulation domain transverse to the strand is equal to $L_{\text{ins}} = 0.07$ mm, which corresponds to the thickness of a real insulation layer (of only one strand) without resin. The insulation conduction area $A_{\text{ins, cond}}$ remains unchanged in this set of analyses, i.e. $u_{\text{ins}} = 1.0$. The geometry is discretised on each side of the coil to uniformly distribute the the insulation conduction area $A_{\text{ins, cond}}$ along the entire domain of the windings. The parameter $n_{\text{divisions, insulation}}$ corresponds to an average length of a coil rounding. The parameters $n_{\text{divisions, side d}}$ and $n_{\text{divisions, side e}}$ refer to Fig. 8.19.

There are 72 nodes in each winding representing the composite strand. The insulation layer of every winding is reduced to two elements. This means that the insulation layer

between two neighbouring windings accounts for four elements, out of which two are coupled with one another. Therefore, the insulation modelling fulfils the basic condition of simulating nonlinear phenomena, that require using at minimum three nodes. The longitudinal mesh size is equal to 15 mm whereas the insulation mesh size – 35 μm. The entire discretised model contains approximately half a million of nodes. Reducing the number of insulation elements aims at lowering the total model size.

Table 8.8: Geometry and mesh characteristics.

parameter	value	unit
$L_{\text{coil, ANSYS}}$	809.9	[m]
L_{ins}	0.07	[mm]
$A_{\text{ins, cond}}$	$8.75 \cdot 10^{-6}$	[m^2]
u_{ins}	1.0	[-]
$n_{\text{divisions, insulation}}$	2	[-]
$n_{\text{divisions, winding}}$	64	[-]
$n_{\text{divisions, side d}}$	26	[-]
$n_{\text{divisions, side e}}$	6	[-]
mesh size, insulation	35	[μm]
average mesh size, strand	15	[mm]
nodes in coil	478 000	[-]

There are three quench analyses conducted with a varying heat capacity of resin represented by the parameter u_{resin}, as depicted in Table 8.9. Adding 0D elements to the strand increases the total number of elements in the model from 480 000 to 530 000.

Table 8.9: Geometry and mesh characteristics.

u_{resin}	V_{MASS71}, [m^3]	elements in coil
0.0	0.0	476 528
0.5	$2.4 \cdot 10^{-9}$	531 574
1.0	$4.8 \cdot 10^{-9}$	531 574

8.7.2 Analysis Settings

The co-simulation time window remains identical with respect to the previous chapters, as depicted in Table 8.10. The ANSYS simulation time step is assumed to be constant and equal to the maximum value of a time step range used for the quench velocity-based

approach in Chapter 7. The time step is increased to keep the computing time in a reasonable limit of several days. The initial energy deposition is placed in the centre of a winding directly neighbouring with the spot heater corresponding to a position of 367.67 m in the meshed model.

Table 8.10: Input parameters in the quench detection analysis of the skew quadrupole.

parameter	value	unit
t_{com}	2.5	[ms]
t_{step}	1	[ms]
hot-spot position in coil	367.97	[m]

8.7.3 Results

The evolution of resistive voltage up to the point of the quench detection is shown in Fig. 8.24. The results from three analyses are compared with the measurements. The initial slope of the simulated models and the measured magnet remains similar at $t < 0.1$ s. This means that the initial quench velocity set in the model corresponds to its real value in the quenched winding at $I = 86$ A. In the measurements, the first turn-to-turn propagation occurs at $t \approx 0.08$ s which is much faster than what any simulation predicts. However, the initial heat impulse is much larger in the measurements than in the simulated cases. Therefore, the first transverse propagation also occurs more slowly in the simulations than in the experiment. Including resin in the models results in a slower turn-to-turn propagation as well. Such a phenomenon is expected due to a higher heat capacity of the windings with resin.

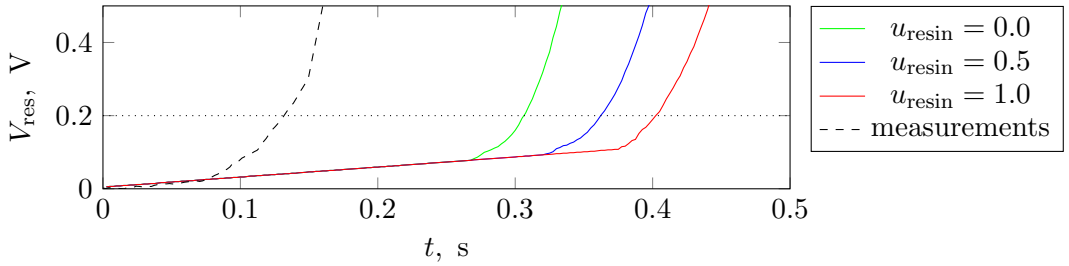

Figure 8.24: Rise of resistive voltage.

Once the first turn-to-turn propagation occurs, the simulated models start resembling the measurements because the current discharged in the coil becomes the predominant heat source. In order to compare the measurements with the performed analyses, each resistive

voltage obtained in the simulations is translated in time, so that the threshold voltage V_{th} is reached within the same time. Table 8.11 presents the time instants at which the threshold voltage is obtained for every case. The translation times are used in the next section to translate the plots of the resistive voltage and the resistance. Hereby, the comparison of the simulations with the results becomes more clear.

Table 8.11: List of time windows at which the simulations reaches V_{th}.

parameter	$t(V_{\text{th}})$	$t_{\text{translation}}$	unit
measurements	0.13	-	[s]
$u_{\text{resin}} = 0$	0.30	-0.17	[s]
$u_{\text{resin}} = 0.5$	0.36	-0.23	[s]
$u_{\text{resin}} = 1.0$	0.40	-0.27	[s]

8.8 Analysis of Magnet Discharge

8.8.1 Additional Input Parameters

After the threshold voltage V_{th} and the time delay t_{delay} are reached, the quench is detected, as discussed in Section 8.3. At that moment, ANSYS solves a discharge of an RL-circuit with an initially applied current of $I = 86$ A. The model takes the nonlinear inductance of the skew quadrupole presented in Fig. 8.25 into account.

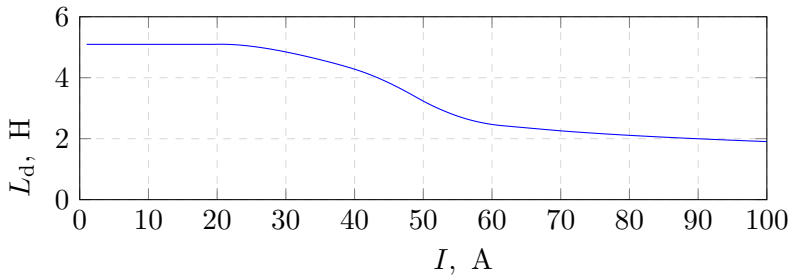

Figure 8.25: Differential inductance as a function of current for the skew quadrupole [23].

The circuit represented in Fig. 8.9 is solved with additional electrical elements in ANSYS, referred to as CIRCU124. Voltage is a single degree of freedom of the resistors and inductors modelled by CIRCU124 elements [26]. The nonlinear inductance is updated together with the quench position in time at each communication point t_{com} between Python and ANSYS. Table 8.12 depicts additional simulation parameters for the discharge phase of the magnet. The quench velocity-based approach is only needed until the entire coil

quenches. After that point, therefore, the communication time step is increased to $t = 0.2$ s. The remaining communication time steps are required for updating the inductance of a magnet and extracting the current. The material properties need to be updated as they are a function of the magnetic field strength. This, however, is very time consuming for 754 windings and, is, thus, only done if if the current I drops by such an amount that implies a significant change in the magnetic field strength. For this simulation, the value of current is set to $I = 10$ A. It is assumed that the magnet is fully discharged if the transport current drops below $I = 8$ A, i.e. if the current decreased by approximately 90% of its initial value.

Table 8.12: Input parameters in the analysis of the skew quadrupole discharge.

parameter	value	unit
t_{com} before the entire coil quenches	0.0025	[s]
t_{com} after the entire coil quenches	0.2	[s]
material properties update criterion	10	[A]
magnet discharge criterion	8	[A]

8.8.2 Results

Fig. 8.26 presents the evolution of the resistive voltage V_{res} in time. The more resin is considered in the model, the lower is the peak resistive voltage obtained. Moreover, the peak voltage occurs faster with lower u_{resin}. In principle, the peak voltage corresponds to the moment when both longitudinal and turn-to-turn quench propagation result in a quench of the entire coil. In Fig. 8.26, one can also notice sharp changes of resistive voltage which occur in gradually increasing time periods. These changes are due to the update of material properties every 10 A.

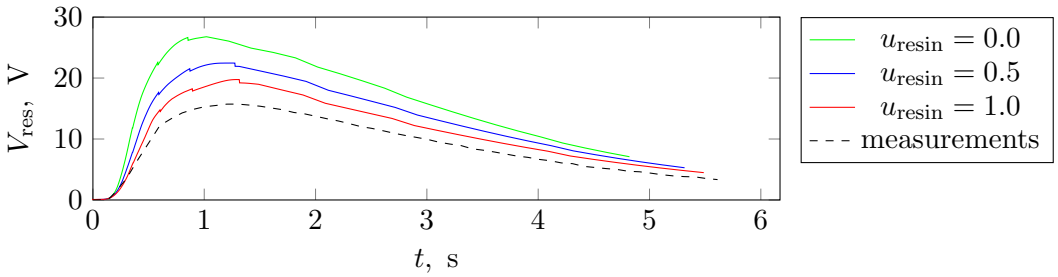

Figure 8.26: Coil resistive voltage change in time.

All simulations result in an overestimation of the peak resistive voltage with respect to the measurements. This is associated with the change of the coil resistance during the dis-

charge presented in Fig. 8.27. The more resin is included in the model, the lower the coil resistance becomes. Resin is an additional heat capacity meaning that the coil requires more energy to warm up.

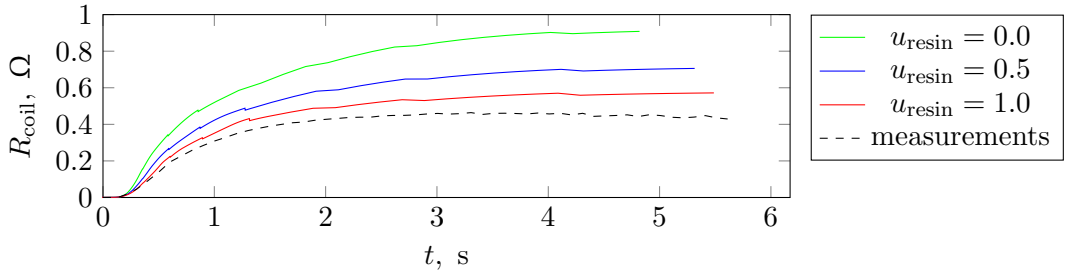

Figure 8.27: Coil resistance evolution in time.

The final temperature of the coil for every simulated case is shown in Fig. 8.28. In every model, the temperature oscillates in the range of approximately $T = 20$ K along the coil. The initially quenched winding is characterised by a higher temperature than the rest of the coil. The hot-spot remains, thus, in the zone where the heat impulse was initially deposited.

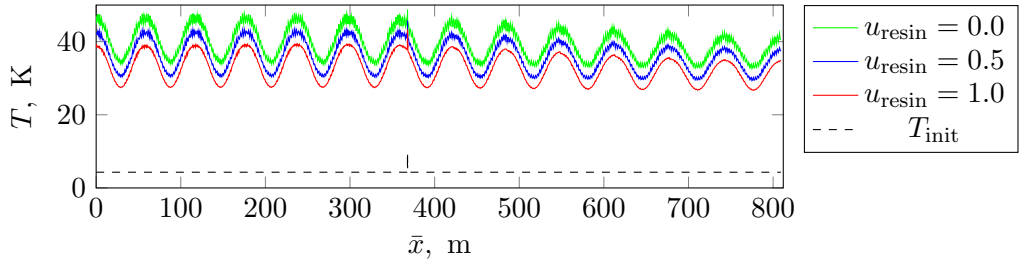

Figure 8.28: Final temperature of the coil after the discharge.

The evolution of the hot-spot temperature in time is presented in Fig. 8.29. The hot-spot temperature remains relatively constant until $t \approx 0.4$ s in each model. This period corresponds to the transverse heat propagation across the insulation, when the turn-to-turn propagation does not occur yet. In the region of $t \in (0.4, 2.0)$ s, the quench rapidly propagates longitudinally and transversely. Over this time a large amount of energy is deposited in the coil while discharging the circuit. At $t \approx 2.0$ s the entire coil is quenched in all three considered cases. The evolution of the hot-spot temperature quickly slows down after that point because: (i) the transport current decreases with time, (ii) the heat capacity of all materials increases with rising temperature. In addition, the hot-spot temperature decreases with a higher volume of resin in the coil.

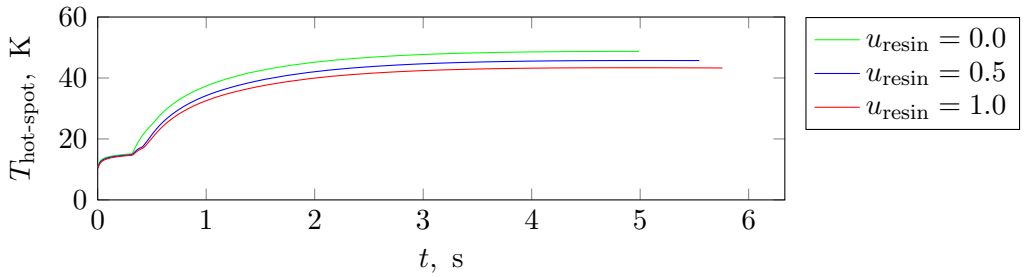

Figure 8.29: Hot-spot temperature evolution in time.

As presented in Fig. 8.30, the current discharge occurs faster in all simulated cases than what was measured. This is the result of a higher coil resistance in the simulations compared to the measurements. In principle, the more resin is considered in the model, the more slowly the discharge occurs.

There is one additional analysis added to the plot, in which the coil resistance is not included in the simulation, i.e. the magnet discharge is only dependent on the dump resistor of the energy extraction system. The analysis is less computationally demanding with respect to the previous simulations, because it only solves an electrical circuit without any thermal dependence. The communication time window t_{com}, at which the inductance is updated, is equal to $t = 0.0025$ s for the entire simulation. The ANSYS time step is also lower and equal to $t = 0.1$ ms. The difference in the drop of current between the measurements and this case indicates the influence of the coil resistance on the speed of the magnet discharge.

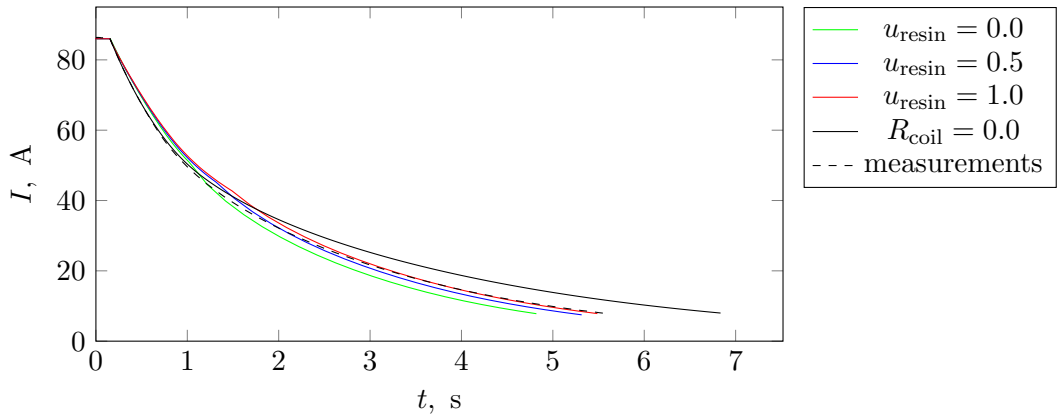

Figure 8.30: Current discharge curve of the skew quadrupole.

8.8.3 Computation Time of Skew Quadrupole Analysis

As depicted in Table 8.13, the total computation time[1] is different for every simulated case. In case of $u_{\text{resin}} = 0.5$, the computation time is not presented, because the analysis was conducted on a different calculation unit. Therefore, a direct comparison of the computation time is not possible. The creation of ANSYS geometry lasts 12 hours in every analysis because each winding is created separately together with its insulation elements. The reuse of created geometry across different studies would lead to a reduction of the computation time. However, this optimisation is not explored in the thesis. The material properties are updated nine times, which, in total, costs one additional hour during the analysis. The ANSYS computation time is relatively constant in all cases with a speed rate of 10.8 hours for one second of the simulation. The co-simulation time depends strongly on the number of co-simulation windows. In principle, the time of the signal exchange is proportional to the number of the quenched zones to assign in the coil in the quench velocity-based approach. Right before the coil fully quenches, there are approximately 700 quenched zones to consider. Moreover, the co-simulation time includes the period of saving the analysis after each co-simulation time window and the ANSYS restarting process.

Table 8.13: Computation time summary.

parameter	$u_{\text{resin}} = 0.0$	$u_{\text{resin}} = 0.5$	$u_{\text{resin}} = 1.0$	unit
geometry creation time	12	12	12	[h]
material update time	1	1	1	[h]
ANSYS computation time	54	-	63	[h]
co-simulation time	52	-	80	[h]
number of co-simulation signals	494	637	722	[-]
total computation time	119	-	156	[h]
$t_{\text{simulation}}$	5.0	5.55	5.8	[s]

It should be clarified, that the ANSYS computation time will be higher with a decreased simulation time step t_{step}. The recommended time step for the quench velocity-based approach is $t_{\text{step range}} = [0.1, 1]$ ms. Therefore the applied time step of $t_{\text{step}} = 1$ ms is the maximum value of the recommended time step range in the quench velocity-based approach. If a simulation without the quench velocity-based approach was considered, one would have to take a prolonged time for the creation of geometry into account, due to an increased number of degrees of freedom. Moreover, the ANSYS computation time would be

[1] The analysis was performed on the following calculation unit: Intel(R) Xeon(R) CPU E5-2667 V4 @ 3.20 GHz (2 processors) with RAM 128 GB.

even longer in this case because of a considerable rise in the number of degrees of freedom.

A possibility of improving the computation time in both the quench velocity-based approach and the standard simulation could rely on using the CERN High-Performance Computing (HPC) cluster. Nevertheless, the cluster application for solving quench propagation problems remains outside of the scope of this thesis.

8.8.4 Correction of Results

In the last step of the quench analysis, it is required to take the relative error associated with the quench velocity-based approach into account, as discussed in Chapter 7. An estimation of corrected results of the peak resistive voltage and the hot-spot temperature are presented in Tables 8.14, and 8.15, respectively. For both parameters, it is assumed that the longitudinal mesh size is equal to 20 mm, meaning that the error margin is higher with respect to the model of the skew quadrupole with an average mesh size of 15 mm. In the corrected value of the resistive voltage, one assumes a constant relative error independent of time. In case of the hot-spot temperature, the evolution of the error over time is included in the correction method. It is assumed that all simulations last approximately 6 seconds. As discussed in the previous section, all design parameters reach the highest value when no resin is included in the skew quadrupole model. Multiplying the resistive voltage by the correction factor increases the discrepancy between the model and the measurements. The relative error $E_{\text{r, measurements}}$ compares $V_{\text{res, max}}$ obtained from the measurements with the corrected values of $V_{\text{res, max, corrected}}$. The maximum resistive voltage is more than doubled when resin is neglected. If the full resin volume is considered, the peak voltage is 56% higher than the measured value. Since, the hot-spot temperature is not measured during the tests, one can only rely on the simulations.

Table 8.14: Correction of the resistive voltage.

	measurements	$u_{\text{resin}} = 0.0$	$u_{\text{resin}} = 0.5$	$u_{\text{resin}} = 1.0$
$V_{\text{res, max}}$, V	15.8	26.8	22.5	19.8
$E_{\text{r, initial}}$	-	-20%	-20%	-20%
$V_{\text{res, max, corrected}}$, V	-	33.6	28.1	24.7
$E_{\text{r, measurements}}$	-	113%	78%	56%

Table 8.15: Estimation of the real hot-spot temperature.

	$u_{\text{resin}} = 0.0$	$u_{\text{resin}} = 0.5$	$u_{\text{resin}} = 1.0$
$T_{\text{hot-spot}}$, K	48.7	45.7	43.4
$E_{\text{r, initial}}$	-1.5%	-1.5%	-1.5%
$E_{\text{r, additional}}$	-2.0%/s	-2.0%/s	-2.0%/s
$t_{\text{simulation}}$, s	≈ 6	≈ 6	≈ 6
$E_{\text{r, total}}$	-13.5%	-13.5%	-13.5%
$T_{\text{hot-spot, corrected}}$, K	56.3	52.9	50.1

8.9 General Remarks

Three simulated cases of the magnet discharge allowed for predicting the final temperature profile of the coil, including the evolution of the hot-spot temperature. In principle, when a turn-to-turn propagation is considered, the temperature rise in the hot-spot is less steep than in case of a 1D longitudinal heat propagation, because a part of the accumulated heat is evacuated to neighbouring windings. Therefore, the multi-strand analysis allows for lowering the safety coefficients with respect to the peak hot-spot temperature in case of a quench. On the other hand, the skew quadrupole model overestimates the temperature distribution in the coil regardless of the analysis case conducted with or without resin. Such a conclusion can be made by observing the curves of coil resistance being higher than in case of the measurements. An overestimation of the temperature in the coil is contradictory to the assumptions related to the quench velocity-based approach in which the temperature is underestimated as outlined in Chapter 7. The discrepancy in the results with respect to the measurements can be explained as follows:

1. The mesh size across the insulation is relatively large when the entire skew quadrupole is simulated. As discussed in Section 7.4.1, a coarser mesh across the insulation results in an overestimation of the hot-spot temperature (see Fig. 7.12).

2. No influence of helium is considered. In principle, helium is an important additional heat capacity in the system that should reduce the coil temperature during the discharge.

3. The insulation and resin are homogenised and modelled with G10. The material properties of G10 are assumed to be similar to S2-glass and CTD-101K epoxy resin. By comparing the simulation results with and without resin, it can be concluded that the

heat capacity of the coil elements not belonging to the composite strand have a considerable influence on the temperature evolution of the system during the discharge.

By taking the aforementioned arguments into account, it can be concluded, that the model is validated against available measurements of the magnet discharge while assuming a given accuracy. The overestimation of the quench velocity-based approach for the skew quadrupole remains in the range of 50-120% with respect to the peak resistive voltage. In fact, an overestimation of the resistive voltage or the hot-spot temperature increases the safety margin from the magnet design standpoint. Therefore, it is a favourable case of the discrepancy between the model and the measurements.

Chapter 9

Summary

First of all, the physics implemented in ANSYS models was cross-checked in this thesis with STEAM-BBQ, being a standard tool at TE-MPE-PE Section to simulate a 1D quench propagation. Then, the quench velocity-based approach was introduced, which aims at reducing the number of degrees of freedom in the quench problem allowing for simulating a full-scale coil of a magnet. The ANSYS models and the algorithms governing the quench velocity-based approach were described. The quench velocity-based approach was verified with standard ANSYS simulations, which do not apply the proposed method (instead, they are based on the heat balance equation). The aim of this study was to estimate the error associated with relaxing the mesh density in the quench velocity-based approach. The verification between the standard ANSYS analysis and the quench velocity-based approach was conducted with simplified models only simulating the longitudinal quench propagation. The quench velocity-approach allowed for simulating the skew quadrupole discharge being one of the high-order corrector magnets for the HL-LHC upgrade. One coil of the magnet was simulated in which both the longitudinal and the turn-to-turn propagation were analysed and validated against available measurements.

The implementation of the quench velocity-based approach leads to a considerable decrease in the number of degrees of freedom of the numerical model. The lower number of degrees of freedom allows for simulating a full-scale coil of a magnet. Nevertheless, the simulation remains an approximation, in which a certain error should be considered. It is recommended to take the error into account because the key magnet design parameters, such as the hot-spot temperature as well as the voltage-to-ground, are underestimated in the quench velocity-based approach with respect to a standard numerical solution. In the quench velocity-based approach, the improvement rate in computation time is the highest when an external insulation layer (optionally including the epoxy resin) with a relatively high number of nodes is simulated. In this case, a high mesh relaxation in the longitudinal

direction of the coil allows for a considerable decrease in the number of insulation nodes nodes with respect to the standard numerical analysis. Based on the example of the skew quadrupole, the discharge of the magnet, in which the quench occurs in one coil, can be simulated within five-seven days. Unfortunately, even the quench velocity-based approach did not allow for simulating the skew quadrupole with a refined insulation layer due to an exceeding computation time. The mesh refinement of 20 elements across the insulation would make the model too heavy to compute for ANSYS (in the order of 4 million elements). Therefore, the high improvement of the computation time is not visible when the quench velocity-based approach is used. In principle, the more accurate a discretisation of the insulation layer is applied in the model, the higher the gain in the computation time that is reached with the quench velocity-based approach as compared to a standard analysis with the same mesh across the insulation. Overall, the multi-strand quench simulation of full coils, relying on the quench velocity-based approach, will remain a tool implemented in the final design stage of the quench protection study, rather than in its beginning when short iteration loops are required.

The quench velocity-based approach is successfully implemented in ANSYS by using an external routine developed in Python imposing the position of the quenched zone as the simulation proceeds. The ANSYS model allows for studying the current discharge in an RL-circuit with a constant dump resistance and the inductance being a function of the transport current. It is possible to neglect the dump resistance in case of future quench simulations of self-protected magnets. The temperature evolution inside of the coil relies on the longitudinal and the turn-to-turn quench propagation. Moreover, the material properties are a function of temperature and magnetic field strength during the quench propagation in a magnet. The simulation of the full magnet discharge requires using the following ANSYS elements:

1. LINK68 used for the composite strand.
2. LINK33 applied to simulate the insulation layer.
3. CIRCU124 required for connecting the coil to an external power source (independent current source), resistor and inductor.
4. MASS71 optionally used to take resin into account when fully impregnated coils are simulated. Moreover, a part of the insulation layer may also be compressed in the given elements.

Appendices

Appendix A

Material Properties

The material properties used in this thesis are presented and discussed in the following Appendix. Since July 2019, it is recommended at TE-MPE-PE to use material fit functions provided by NIST. Except for the volumetric heat capacity of Nb-Ti, which NIST does not provide, all material properties are based on an update ROXIE[1] database prepared at CERN [16]. Nb-Ti is defined based on CUDI being Arjan Verweij's CERN internal routine for modelling of superconducting strands and cables [34].

A.1 Copper

A.1.1 RRR

Residual resistivity ratio, RRR is a parameter comparing copper resistivity at two different temperatures when no magnetic field is applied. It is calculated as:

$$\mathrm{RRR} = \frac{\rho(T_{\mathrm{high}}, B = 0\ \mathrm{T})}{\rho(T_{\mathrm{low}}, B = 0\ \mathrm{T})}, \tag{A.1}$$

where $T_{\mathrm{high}} = 273$ K and $T_{\mathrm{low}} = 4$ K.

A.1.2 Resistivity

Copper resistivity depends on temperature and RRR:

$$\rho_{\mathrm{N}}(T, \mathrm{RRR}) = \rho_0 + \rho_{\mathrm{i}} + \rho_{\mathrm{i0}}, \tag{A.2}$$

[1] ROXIE is CERN internal software for electromagnetic simulation and optimisation of accelerator magnets.

where:

$$\rho_0 = \frac{1.5553 \cdot 10^{-8}}{\mathrm{RRR}}, \tag{A.3}$$

$$\rho_\mathrm{i} = \frac{\mathrm{P}_1}{1 + \mathrm{P}_1\mathrm{P}_3, T^{\mathrm{P}_2 - \mathrm{P}_4}, \exp\left[-(\frac{\mathrm{P}_5}{T}^{\mathrm{P}_6})\right]}, \tag{A.4}$$

$$\rho_\mathrm{i0} = \mathrm{P}_7 \frac{\rho_\mathrm{i}\rho_0}{\rho_\mathrm{i} + \rho_0}, \tag{A.5}$$

with fit parameters depicted in Table A.1.

Table A.1: Fit P-parameters for copper electrical resistivity

P_1	P_2	P_3	P_4	P_5	P_6	P_7
$1.171 \cdot 10^{-17}$	4.49	$3.841 \cdot 10^{10}$	1.14	50	6.428	0.4531

In order to apply the resistivity dependence on magnetic field strength, the following formula is applied:

$$\rho_\mathrm{N}(T, B, \mathrm{RRR}) = \rho_\mathrm{N}(T, \mathrm{RRR}) \cdot (1 + 10^{a(x)}), \tag{A.6}$$

where:

$$a(x) = \sum_{n=0}^{N} a_\mathrm{n} (\log_{10} x)^n, \tag{A.7}$$

$$x \approx \frac{1.553 \cdot 10^{-8}}{\rho_\mathrm{N}(T, \mathrm{RRR})} \cdot B, \tag{A.8}$$

with B – magnetic field strength, a – fit parameters described in Table A.2.

Table A.2: Fit a-parameters for copper electrical resistivity

a_0	a_1	a_2	a_3	a_4
-2.662	0.3168	0.6229	-0.1839	0.001827

As presented in Fig. A.1, the resistivity of copper rises with rise of temperature. It rises with increase of magnetic field strength as well as with decrease of RRR.

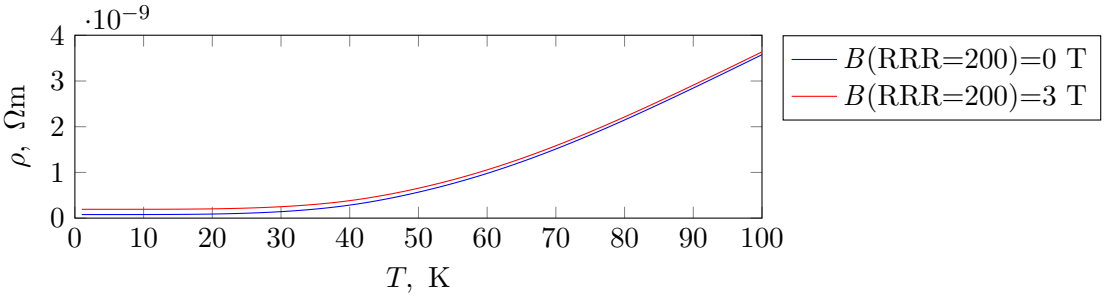

Figure A.1: Copper resistivity temperature dependence for two values of magnetic field strength.

A.1.3 Thermal Conductivity

Copper thermal conductivity depends on temperature and RRR:

$$k_{\mathrm{N}}(T, \mathrm{RRR}) = \frac{1}{\mathrm{W}_0 + \mathrm{W}_\mathrm{i} + \mathrm{W}_{\mathrm{i}0}}, \tag{A.9}$$

where:

$$\mathrm{W}_0 = \frac{\beta}{T}, \tag{A.10}$$

$$\mathrm{W}_\mathrm{i} = \frac{\mathrm{P}_1 T^{\mathrm{P}_2}}{1 + \mathrm{P}_1 \mathrm{P}_3 T^{\mathrm{P}_2+\mathrm{P}_4} \exp\left[-(\frac{\mathrm{P}_5}{T}^{\mathrm{P}_6})\right]}, \tag{A.11}$$

$$\mathrm{W}_{\mathrm{i}0} = \mathrm{P}_7 \frac{\mathrm{W}_\mathrm{i} \mathrm{W}_0}{\mathrm{W}_\mathrm{i} + \mathrm{W}_0}, \tag{A.12}$$

with fit parameters presented in Table A.3.

Table A.3: Fit β- and P-parameters for copper thermal conductivity

β			β_r			
0.634/RRR			β/0.0003			
P_1	P_2	P_3	P_4	P_5	P_6	P_7
$1.754 \cdot 10^{-8}$	2.763	1102	-0.165	70	1.756	$0.838/\beta_\mathrm{r}^{0.1661}$

In order to include the magnetic induction dependence in the function of thermal conductivity, the following formula is applied:

$$k_{\mathrm{N}}(T, B, \mathrm{RRR}) = \frac{\rho\mathrm{N}(T, B = 0, \mathrm{RRR})}{\rho\mathrm{N}(T, B = 0, \mathrm{RRR})} \cdot k_{\mathrm{N}}(T, \mathrm{RRR}) \tag{A.13}$$

As presented in Fig. A.2, the thermal conductivity of copper decreases with rise of magnetic field strength. One can also notice that the peak value of thermal conductivity is higher with increase of RRR.

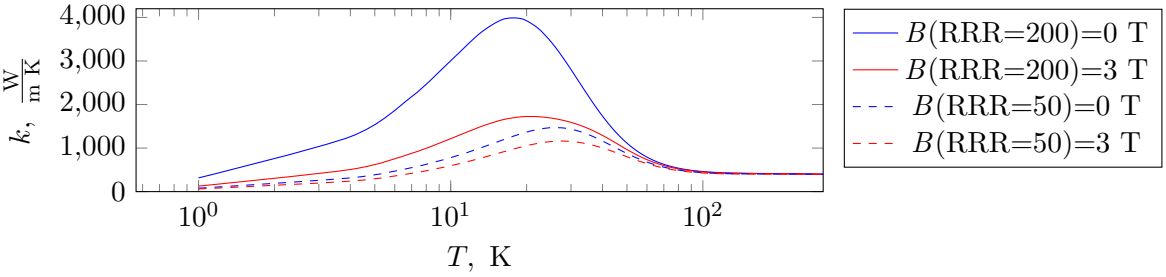

Figure A.2: Thermal conductivity of copper as a function of temperature for two values of magnetic field strength and RRR.

A.1.4 Mass Density

The mass density of copper is assumed to be constant and equal to $\rho = 8960$ kg/m^3.

A.1.5 Volumetric Heat Capacity

Copper specific heat capacity is approximated with a polynomial interpolation as:

$$x(T) = 10^{\sum_{n=0}^{N} a_n (\log_{10} T)^n}, \tag{A.14}$$

where N – order of polynomial, a – fit parameters described in Table A.4.

Table A.4: Fit parameters for copper specific heat capacity

a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7
-1.91844	-0.15973	8.61013	-18.996	21.9661	-12.7328	3.54322	-0.3797

In order to convert specific into volumetric heat capacity, the polynomial function is divided by copper density. The volumetric heat capacity of copper is presented in Fig. A.3.

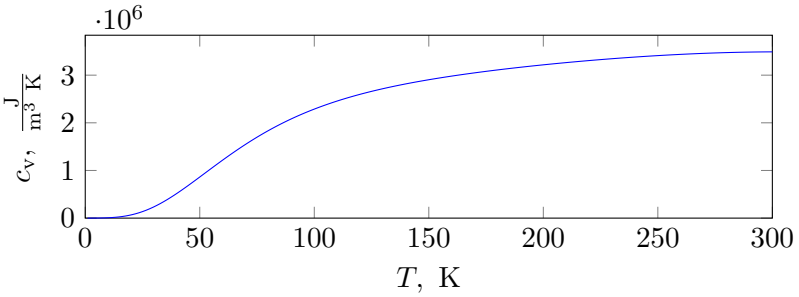

Figure A.3: Volumetric heat capacity of copper as a function of temperature.

A.2 Niobium Titanium

A.2.1 Mass Density

The mass density is assumed to be constant and equal to $\rho = 6000$ kg/m^3.

A.2.2 Volumetric Heat Capacity

Volumetric heat capacity of Nb-Ti is estimated by means of a piece-wise polynomial function which takes into account a linear magnetic dependence below the critical temperature. The critical temperature is calculated as:

$$T_{\mathrm{c}}(B) = T_{\mathrm{c0}} \cdot (1 - \frac{B}{B_{\mathrm{c20}}})^{0.59}, \tag{A.15}$$

where T_{c0} – maximum critical temperature for $B = 0$ T, B_{c20} – maximum upper critical magnetic field for $T = 0$ K. Their numerical values for Nb-Ti are presented in Table A.5. In both parameters, the current density is equal to zero.

Table A.5: Critical temperature parameters for Nb-Ti [35].

parameter	value	unit
T_{c0}	9.2	[K]
B_{c20}	14.5	[T]

The volumetric heat capacity is calculated in the following manner:

$$c_{\mathrm{v,\ Nb\text{-}Ti}}(T, T_{\mathrm{c}}) = \frac{\mathrm{a}_0 + \mathrm{a}_1 \cdot T^1 + \mathrm{a}_2 \cdot T^2 + \mathrm{a}_3 \cdot T^3 + \mathrm{a}_4 \cdot T^4}{\rho_{Nb-Ti}}, \tag{A.16}$$

where a – fit parameters presented in Table A.6.

Table A.6: Polynomial parameters for Nb-Ti volumetric heat capacity.

	a_0	a_1	a_2	a_3	a_4
$T \in (0, T_c)$ K	0	$64 \cdot B$	0	49.1	0
$T \in [T_c, 28.358)$ K	0	928	0	16.24	0
$T \in [28.358, 50.99)$ K	41383	-7846.1	553.71	11.9838	-0.2177
$T \in [50.99, 165.8)$ K	$-1.53 \cdot 10^6$	83022	-716.3	2.976	-0.00482
$T \in [165.8, 496.54)$ K	$1.24 \cdot 10^6$	13706	-51.66	0.09296	$-6.29 \cdot 10^{-5}$
$T \in [496.54, 1000)$ K	$2.45 \cdot 10^6$	955.5	-0.257	0	0

As presented in Fig. A.4, there occurs a step jump in Nb-Ti heat capacity at critical temperature. Since the critical temperature decreases with rise of magnetic field strength, the transition occurs at lower temperatures when a higher B-field is applied.

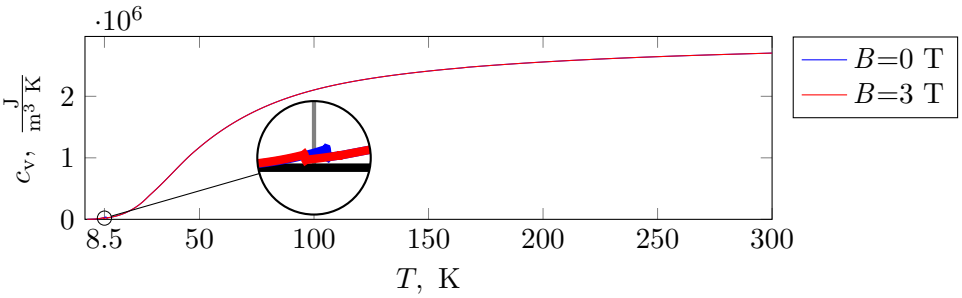

Figure A.4: Volumetric heat capacity of Ni-Ti as a function of temperature.

A.3 G10 Material Properties

A.3.1 Thermal Conductivity

G10 is an epoxy-impregnated fiberglass, thermally anisotropic. It means that its thermal conductivity along the fibers is higher than across them. In order to simplify computation in the scope of this thesis, the material is assumed to be isotropic. A conservative assumption is made which states that thermal conductivity of G10 in all directions is equal to the one which is normal to the direction of the fibers. G10 thermal conductivity as well as its specific heat capacity is approximated with the NIST polynomial interpolation as:

$$x(T) = 10^{\sum_{n=0}^{N} a_n (\log_{10} T)^n}, \tag{A.17}$$

where $x(T)$ – considered material property, N – order of polynomial, a – fit parameters described in Table A.7.

Table A.7: Fit parameters for G10 thermal conductivity

a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8
-4.1236	13.788	-26.068	26.272	-14.663	4.4954	-0.6905	0.0397	0

The thermal conductivity of G10 as a function of temperature is presented in Fig. A.5.

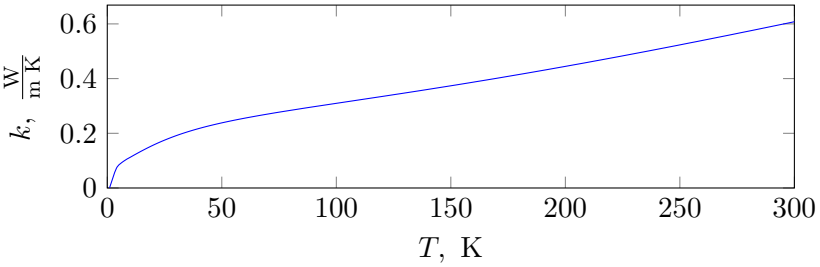

Figure A.5: G10 thermal conductivity as a function of temperature.

A.3.2 Mass Density

The mass density is assumed to be constant and equal to $\rho = 1948$ kg/m^3.

A.3.3 Volumetric Heat Capacity

Specific heat capacity of G10 is based on equation described in (A.17) with fit parameters described in A.8.

Table A.8: Fit parameters for G10 specific heat capacity

a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7
-2.4083	7.6006	-8.2982	7.3301	-4.2386	1.4294	-0.24396	0.015236

In order to convert specific into volumetric heat capacity, the polynomial function is divided by the density of G10. The volumetric heat capacity of G10 is presented in Fig. A.6.

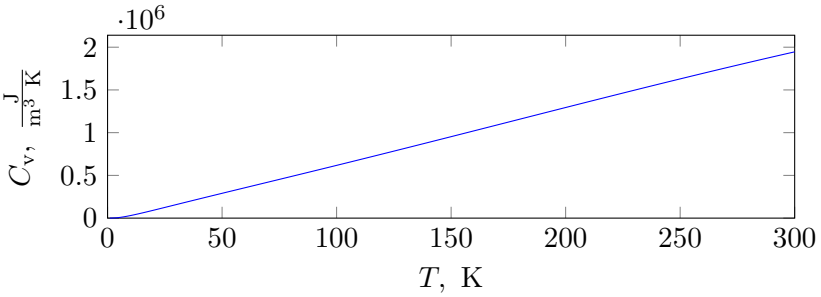

Figure A.6: Volumetric heat capacity of G10 as a function of temperature.

Appendix B

Finite Element Method in Heat Conduction Problems

This appendix gives an overview on consecutive steps applied while solving a heat conduction problem by means of a Finite Element Method. It is based on the the introductory course for FEM Modelling at ETH Zurich [14]. One can refer to this position to understand this topic in more details. Let us consider a partial differential equation of an unsteady one-dimensional heat conduction equation as

$$\frac{\partial T}{\partial t} = \frac{\partial}{\partial x}\left(k(x)\frac{\partial T}{\partial x}\right) + s(x), \tag{B.1}$$

where $k(x)$ – thermal conductivity as a function of space, $s(x)$ – external heat source as a function of spaces. A temperature distribution is searched over the space and time, $T(x,t)$ that satisfies the given formula over the domain $\Omega \in [0, L]$. Two types of boundary conditions can be specified in the following problem:

1. Dirichlet condition as

$$\begin{cases} T(x_{\mathrm{A}}, t) = T_{\mathrm{A}}, \\ \\ T(x_{\mathrm{B}}, t) = T_{\mathrm{B}}, \end{cases} \tag{B.2}$$

   where T_{A}, T_{B} – constant temperatures at points A and B.

2. Neumann condition as

$$\begin{cases} k\frac{\partial T}{\partial x}\Big|_{x=x_{\mathrm{A},t}} = q_{\mathrm{A}}, \\ \\ k\frac{\partial T}{\partial x}\Big|_{x=x_{\mathrm{B},t}} = q_{\mathrm{B}}, \end{cases} \tag{B.3}$$

where q_A and q_A heat flux at point A and B.

In order to solve the equation being a function of time, the initial condition should be specified over the space as

$$T(x, t = 0) = T_0(x). \tag{B.4}$$

Discretisation involves a division of the analysed space into a finite number of elements consisting of nodes. Figure B.1 shows a one-dimensional element with two nodes at its extremities.

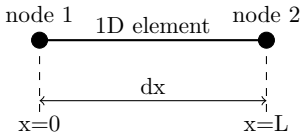

Figure B.1: Exemplary one-dimensional 2-node element [14].

The more nodes is used in the element, the higher accuracy is obtained. However, the computing time increases as well. If two nodes are used for a 1D element, the temperature between them is interpolated linearly. The temperature over the longitudinal space is represented as

$$T(x) \approx \begin{bmatrix} N_1(x) & N_2(x) \end{bmatrix} \begin{bmatrix} T_1(x) \\ T_2(x) \end{bmatrix} = \mathbf{NX}, \tag{B.5}$$

where $N_1(x)$ and $N_2(x)$ – the linear shape functions that allow for approximating the temperature over the continuous space. The shape functions can be reformulated as

$$\begin{cases} N_1(x) = 1 - \frac{x}{L} \\ \\ N_2(x) = \frac{x}{L}, \end{cases} \tag{B.6}$$

where L – length between two nodes in Fig.B.1, x – spatial variable varying for $x \in [0, L]$. The shape functions are illustrated in Fig. B.2.

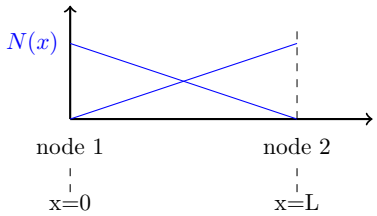

Figure B.2: Representation of linear shape functions $N(x)$ [14].

After substituting the approximated temperature distribution from (B.5) to the differential equation from (B.1), one obtains

$$\frac{\partial}{\partial t}\left(\begin{bmatrix} N_1(x) & N_2(x) \end{bmatrix}\begin{bmatrix} T_1(x) \\ T_2(x) \end{bmatrix}\right) - \frac{\partial}{\partial x}\left(k(x)\frac{\partial}{\partial x}\left(\begin{bmatrix} N_1(x) & N_2(x) \end{bmatrix}\begin{bmatrix} T_1(x) \\ T_2(x) \end{bmatrix}\right)\right) - s(x) = R, \tag{B.7}$$

where R – a residual error being a result of the discretisation process. In the continuous solution, the residual is equal to zero. Equation (B.7) still cannot be solved because the number of variables exceeds the number of equations. In order to solve this problem, Equation (B.7) is multiplied by a set of so called "weighting functions". The weighted residual of such an expression must be equal to zero. One of the method, proposing a specific weighting function is called a Galerkin method. It is more explained in [14, p. 1-11].

In the next steps, the analysis is simplified by assuming the thermal conductivity and heat source independent of space. After applying the weighting functions, a set of algebraic equations is obtained as

$$\begin{bmatrix} \frac{L}{3} & \frac{L}{6} \\ \frac{L}{6} & \frac{L}{3} \end{bmatrix}\begin{bmatrix} \frac{\partial T_1}{\partial t} \\ \frac{\partial T_2}{\partial t} \end{bmatrix} + \bar{k}\begin{bmatrix} \frac{1}{L} & -\frac{1}{L} \\ -\frac{1}{L} & \frac{1}{L} \end{bmatrix}\begin{bmatrix} T_1 \\ T_2 \end{bmatrix} = \begin{bmatrix} N_1\ q_{\mathrm{A}}|_{x_{\mathrm{A}}} \\ N_2\ q_{\mathrm{A}}|_{x_{\mathrm{A}}} \end{bmatrix} + \bar{s}\begin{bmatrix} \frac{L}{2} \\ \frac{L}{2} \end{bmatrix}, \tag{B.8}$$

where $\bar{k}$ – constant thermal conductivity over each volume element $[x_{\mathrm{A}}, x_{\mathrm{B}}]$, $\bar{s}$ – constant heat source over the same element volume, q_{A} – heat flux at point A, q_{B} – heat flux at point B. In the derivation process, Neumann boundary conditions were assumed, as shown in (B.3). Equation (B.8) can be written in a matrix form as

$$\mathbf{A}\frac{\partial \mathbf{T}}{\partial t} + \mathbf{B}\mathbf{T} = \mathbf{F}, \tag{B.9}$$

where

$$\mathbf{A} = \begin{bmatrix} \frac{L}{3} & \frac{L}{6} \\ \frac{L}{6} & \frac{L}{3} \end{bmatrix}, \tag{B.10}$$

$$\mathbf{B} = \begin{bmatrix} \frac{1}{L} & -\frac{1}{L} \\ -\frac{1}{L} & \frac{1}{L} \end{bmatrix}, \tag{B.11}$$

$$\mathbf{T} = \begin{bmatrix} T_1 \\ T_2 \end{bmatrix}, \tag{B.12}$$

$$\mathbf{F} = \begin{bmatrix} N_1\ q_{\mathrm{A}}|_{x_{\mathrm{A}}} \\ N_2\ q_{\mathrm{A}}|_{x_{\mathrm{A}}} \end{bmatrix} + \bar{s}\begin{bmatrix} \frac{L}{2} \\ \frac{L}{2} \end{bmatrix}. \tag{B.13}$$

Since the problem is transient, one should discretise the temperature time derivative. A finite difference method is used with an implicit discretisation as

$$\mathbf{A}\frac{\mathbf{T}^{n+1}-\mathbf{T}^{n}}{\Delta t}+\mathbf{B}\mathbf{T}^{n+1}=\mathbf{F}, \tag{B.14}$$

where Δt – a time step between two iterations of the solution, $\mathbf{T}^{n}$ – known temperature vector from the current iteration, $\mathbf{T}^{n+1}$ – unknown temperature vector from the next iteration to compute. After rearranging the formula, one obtains

$$\mathbf{K}\mathbf{T}^{n+1}=\mathbf{M}\mathbf{T}^{n}+\mathbf{F}, \tag{B.15}$$

where

$$\mathbf{K}=\frac{1}{\Delta t}\mathbf{A}+\mathbf{B}, \tag{B.16}$$

$$\mathbf{M}=\frac{1}{\Delta t}\mathbf{A}. \tag{B.17}$$

The matrix $\mathbf{K}$ is called the element stiffness matrix. While conducting a simulation with more degrees of freedom (DOF), i.e higher number of nodes, the stiffness matrices of every single element add up to form a large global stiffness matrix. The more DOF's are used, the more time consuming the problem becomes.

Appendix C

Implementation in ANSYS APDL

C.1 Creation of Slab Geometry

```
winding_counter = 1
reel_number = 1
is_reel_number_odd = mod(reel_number, 2)
kp_per_winding = division_per_winding+1
ins_elem_number = number_of_windings*2+1
x=0
y=0
z=0

*do,a,1,number_of_windings/number_of_windings_in_reel
  *do,b,1,number_of_windings_in_reel
    allsel
    *get,kp_first_winding_i,kp,,count
    kp_first_winding_i = kp_first_winding_i + 1
    *do,c,1,kp_per_winding,1
      allsel
      *get,knum,kp,,count
        k,knum+1,(c-1)*(length_per_winding)/(division_per_winding),y,z
    *enddo
    allsel
    *get,kp_last_winding_i,kp,,count
    allsel
    *get,knum,kp,,count
    line_counter = knum - division_per_winding
    type,winding_counter
    real,winding_counter
    mat,winding_counter
    *do,k,line_counter,line_counter+division_per_winding-1,1
      l,k,k+1,1
      allsel
      *get,lnum,line,,count
      lmesh,lnum
    *enddo
```

```
allsel
*get,lnum,line,,count
lsel,s,line,,lnum+1-division_per_winding,lnum
allsel,below,line
cm,winding%winding_counter%,line

!!!!!!!!!!!!!!!!!!!!!!!!
!!! CREATE INSULATION !!!
!!!!!!!!!!!!!!!!!!!!!!!!
*if,insulation_analysis,eq,1,then

  *dim,insulation_tip_nodes_array%winding_counter%,
        array,kp_per_winding*4,1,1
  node_ins_tip_counter = 1

  ! get number of keypoints in one
  cmsel,s,winding%winding_counter%
  allsel,below,line
  *get,kps_winding,kp,,count

  ! get maximum keypoint number in the given winding
  cmsel,s,winding%winding_counter%
  allsel,below,line
  *get,kps_max_winding,kp,,num,max
  *get,kps_min_winding,kp,,num,min

  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  ! k creates four sides of the insulation layer !!!!!!!!!!!!!!!!!
  ! j creates the longitudinal elements of each insulation layer !
  ! i creates each layer element separately !!!!!!!!!!!!!!!!!!!!!
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

  *do,k,1,4,1

    x_insulation = 0
    y_insulation = 0
    z_insulation = 0
    *do,j,1,trans_division_insulation,1
      *do,i,kps_min_winding,kps_max_winding,1

        allsel
        *get,knum,kp,,count
        *if,j,eq,1,then
          *get,kposx,kp,i,loc,x
          *get,kposy,kp,i,loc,y
          *get,kposz,kp,i,loc,z
        *else
          *get,kposx,kp,knum+1-kp_per_winding,loc,x
          *get,kposy,kp,knum+1-kp_per_winding,loc,y
          *get,kposz,kp,knum+1-kp_per_winding,loc,z
        *endif
        *if,k,eq,1,then
          y_insulation =
```

```
          + eq_trans_dimension_insulation/
              trans_division_insulation
  *elseif,k,eq,2,then
    y_insulation =
          - eq_trans_dimension_insulation/
              trans_division_insulation
  *elseif,k,eq,3,then
    z_insulation =
          + eq_trans_dimension_insulation/
              trans_division_insulation
  *elseif,k,eq,4,then
    z_insulation =
          - eq_trans_dimension_insulation/
              trans_division_insulation
  *endif

  k,knum+1,kposx+x_insulation,kposy+ \
          y_insulation,kposz+z_insulation
  *if,j,eq,1,then
    l,i,knum+1,1
  *else
    l,knum+1-kp_per_winding,knum+1,1
  *endif

   ! application of corner nodes
   *if,i,eq,kps_min_winding,
           or,i,eq,kps_max_winding,then
   type,ins_elem_number+1
   real,ins_elem_number+1
   mat,ins_elem_number+1
   *else
   type,ins_elem_number
   real,ins_elem_number
   mat,ins_elem_number
   *endif

  allsel
  *get,lnum,line,,count
  lmesh,lnum

  !!! get the tip node number for further coupling purposes
  *if,j,eq,trans_division_insulation,then
    ksel,s,kp,,knum+1
    nslk,s
    *get,num_node,node,,num,max
    insulation_tip_nodes_array%winding_counter%(
          node_ins_tip_counter) = num_node
    node_ins_tip_counter = node_ins_tip_counter + 1
  *endif
*enddo
allsel
*get,enum,elem,,count
esel,s,elem,,enum-division_per_winding,enum
```

```
          cm,insulation_layer%j%_side%k%,elem
        *enddo
      *enddo

      ! create named components for windings with insulation
      *do,m,1,4,1
        *do,l,1,trans_division_insulation,1
          *if,m,eq,1,and,l,eq,1,then
            cmsel,s,insulation_layer%l%_side%m%
          *else
            cmsel,a,insulation_layer%l%_side%m%
          *endif
        *enddo
      *enddo
      cm,winding_insulation%winding_counter%,elem
      cmsel,s,winding%winding_counter%
      allsel,below,line
      cmsel,a,winding_insulation%winding_counter%
      cm,winding_with_insulation%winding_counter%,elem
      allsel,below,elem
      cm,winding_with_insulation_nodes%winding_counter%,node
    *endif

    x = x
    *if,is_reel_number_odd,ne,0,then
      y = y + trans_dimension_winding
    *else
      y = y - trans_dimension_winding
    *endif
    z = z
    winding_counter = winding_counter + 1
  *enddo

  ! winding_counter = winding_counter + 1
  reel_number = reel_number + 1
  is_reel_number_odd = mod(reel_number, 2)

  *if,is_reel_number_odd,ne,0,then
    y = 0
  *else
    y = y - trans_dimension_winding
  *endif
  z = z + trans_dimension_winding
*enddo

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! Coupling algorithm for the insulation nodes !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

*if,insulation_analysis,eq,1,then
  allsel
  tolerance_range = trans_dimension_winding - \
        2*eq_trans_dimension_insulation
```

```
  *do,j,1,number_of_windings,2
    *do,i,1,kp_per_winding*4,1,1
      nnode = insulation_tip_nodes_array%j%(i)
      csys,0
      allsel
      *get,nposx,node,nnode,loc,x
      *get,nposy,node,nnode,loc,y
      *get,nposz,node,nnode,loc,z
      clocal,20,sphe,nposx,nposy,nposz
      csys,20
      nsel,s,loc,x,tolerance_range
      cm,cur_node_set,node
      cmsel,u,winding_with_insulation_nodes%j%
      *get,num_node,node,,count
      *if,num_node,eq,1,then
        *get,num_node,node,,num,max
        allsel
        nsel,s,node,,num_node
        nsel,a,node,,nnode
        cp,next,temp,all
      *endif
    *enddo
  *enddo
*endif

! definition of winding nodes without insulation
*do,i,1,number_of_windings
  *dim,node_list_winding_%i%,array,division_per_winding+1,1
  cmsel,s,winding%i%
  allsel,below,line
  *vget,node_list_winding_%i%,node,,nlist
  *mwrite,node_list_winding_%i%,Nodes_winding_without_insul_%i%,txt
  (1E20.10)
*enddo

! definition of winding nodes with insulation
*if,insulation_analysis,eq,1,then
  *do,i,1,number_of_windings
    cmsel,s,winding_with_insulation%i%
    allsel,below,elem
    *get,node_wind_ins_count,node,,count
    *dim,node_list_winding_ins%i%,array,node_wind_ins_count,1
    *vget,node_list_winding_ins%i%,node,,nlist
    *mwrite,node_list_winding_ins%i%,Nodes_winding_with_insul_%i%,txt
    (1E20.10)
  *enddo
*endif

! definition of node position in coordinate system
csys,0
allsel
*get,nnum_windings,node,,count
*dim,node_loc_windings,array,nnum_windings,4
```

```
*vget,node_loc_windings(1,1),node,,nlist
*vget,node_loc_windings(1,2),node,,loc,x
*vget,node_loc_windings(1,3),node,,loc,y
*vget,node_loc_windings(1,4),node,,loc,z
*mwrite,node_loc_windings,Node_Position,txt
(4E20.10)

*do,i,1,number_of_windings
  *if,i,eq,1,then
    cmsel,s,winding%i%
  *else
    cmsel,a,winding%i%
  *endif
  allsel,below,elem
  *get,nnode,node,,count
*enddo

! definition of nodes in every plane
tolerance_range = 1e-8
*do,i,1,kp_per_winding,1
  nsel,s,node,,i
  *get,node_num,node,0,num,max
  *get,node_num_locx,node,node_num,loc,x
  nsel,s,loc,x,node_num_locx-tolerance_range,node_num_locx+tolerance_range
  *get,node_plane_count,node,,count
  *dim,node_list_plane%i%,array,node_plane_count,1
  *vget,node_list_plane%i%,node,,nlist
  *mwrite,node_list_plane%i%,Nodes_plane_%i%,txt
  (1E20.10)
*enddo

! definition of coil domain
*do,i,1,number_of_windings
*if,i,eq,1,then
    cmsel,s,winding_with_insulation%i%
*else
cmsel,a,winding_with_insulation%i%
*endif
cm,magnet_coil,elem
*enddo

*cfopen,Nnode,txt
*vwrite,nnode
(1(ES16.7))
*cfclose

! definition of point heat capacities
*if,create_point_thermal_capacity,eq,1,then
    *do,i,1,number_of_windings
        *if,i,eq,1,then
            cmsel,s,winding%i%
        *else
            cmsel,a,winding%i%
```

```
        *endif
        cm,magnet_coil_without_insulation,line
    *enddo
    cmsel,s,magnet_coil_without_insulation
    allsel,below,line
    type,ins_elem_number+2
    real,ins_elem_number+2
    mat,ins_elem_number+2
    kmesh,all
*endif

*dim,process_finished,array,1,1,1
process_finished(1) = 1
*mwrite,process_finished,Process_Finished,txt
(1E20.10)
*del,process_finished
save
```

C.2 Solver and Post-Processing Script

```
allsel
solve
save
finish

!!!!!!!!!!!!!!!!!
! PostProcessing !
!!!!!!!!!!!!!!!!!

allsel
cmsel,s,magnet_coil
allsel,below,elem
*get,maxnode,node,,num,max
*get,minnode,node,,num,min
*get,numnode,node,,count

*dim,node_Temp,array,numnode,2
*vget,node_Temp(1,1),node,,nlist
*vget,node_Temp(1,2),node,,temp,
*mwrite,node_Temp,Temperature_Data,txt
(2E20.10)

*if,electric_analysis,eq,1,then
*get,resistive_voltage,node,minnode,volt
*cfopen,Resistive_Voltage,txt
*vwrite,resistive_voltage
(1(ES16.7))
*cfclose
*endif

/post1
/AUTO,1
```

```
/DSCALE,1,OFF
/PLOPTS,INFO,1
/PLOPTS,LEG1,1
/PLOPTS,LEG2,0
/PLOPTS,LEG3,1
/PLOPTS,FRAME,1
/PLOPTS,TITLE,1
/PLOPTS,MINM,1
/PLOPTS,FILE,0
/PLOPTS,WINS,1
/PLOPTS,WP,0
/PLOPTS,DATE,0
/TRIAD,OFF
/RGB,INDEX,100,100,100, 0
/RGB,INDEX, 80, 80, 80,13
/RGB,INDEX, 60, 60, 60,14
/RGB,INDEX, 0, 0, 0,15

allsel
/SHOW,png,,0
cmsel,s,magnet_coil
allsel,below,elem
plnsol,temp
/GFILE,1000,
/REPLOT
/SHOW,CLOSE
/DEVICE,VECTOR,0
/rename,Quench000,png,,STRCAT(
    'Coil-Nodal-Temp',chrval(time_step)),png,
/delete,Quench001,png

*if,electric_analysis,eq,1,then
cmsel,s,magnet_coil
allsel,below,elem
/SHOW,png,,0
plnsol,volt
/GFILE,1000,
/REPLOT
/SHOW,CLOSE
/DEVICE,VECTOR,0
/rename,Quench000,png,,STRCAT(
    'Coil-Nodal-Volt',chrval(time_step)),png,
/delete,Quench001,png
*endif

!!!!!!!!!!!!!!!!!!!!!!!!!!
! Circuit PostProcessing !
!!!!!!!!!!!!!!!!!!!!!!!!!!

/post26
lines,10000     ! avoid repeating headers < 1000 lines
numvar,100      ! set number of variables
ref_number = 10 ! greater than the default of 10
```

```
! Results from the dump resistor
enum = dump_res_elem
esol,ref_number+6,enum,,nmisc,1,pr_dump ! power
esol,ref_number+5,enum,,smisc,1,vr_dump ! voltage
esol,ref_number+4,enum,,smisc,2,ir_dump ! current
/output,sol_dump_resistor,inp
prvar,ir_dump,vr_dump,pr_dump
/output

! Results from the inductor
enum = inductor_elem
esol,ref_number+9,enum,,nmisc,1,pi      ! power
esol,ref_number+8,enum,,smisc,1,vi      ! voltage
esol,ref_number+7,enum,,smisc,2,ii      ! current
/output,sol_inductor,inp
prvar,ii,vi,pi
/output

*dim,process_finished,array,1,1,1
process_finished(1) = 1
*mwrite,process_finished,Process_Finished,txt
(1E20.10)
*del,process_finished

*del,node_Temp
```

Appendix D

Python Implementation

D.1 Compilation of ANSYS APDL with Python

In order to control ANSYS APDL in Python, an external library ansys-corba is used. During the analyses, ANSYS runs in a batch mode referred as "-aas" mode. As illustrated in Fig D.1, in order to launch ANSYS APDL in this mode, one should click on the Customisation window in which an additional parameter "-aas" should be specified. Moreover, the Python script should run in the directory where the numerical simulation is opened in ANSYS APDL.

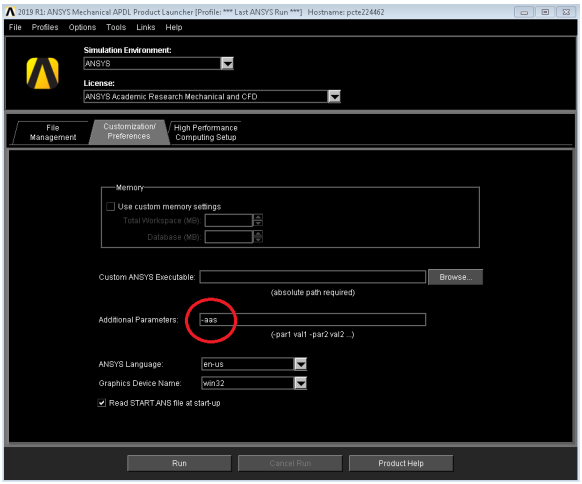

Figure D.1: ANSYS APDL launching scheme.

The representative code for Python compilation with ANSYS environment is presented in Table D.1. By using the methods .executeCommandToString(), one can simply translate ANSYS APDL scripting commands into a Python language.

Table D.1: Compilation of Python with ANSYS environment

from ansys_corba import CORBA	- imports ANSYS library
os.chdir(directory)	- sets analysis directory
with open('aaS_MapdlID.txt', 'r') as f:	
aasMapdlKey = f.read()	- opens APDL analysis
mapdl = CORBA.ORB_init().string_to_object(aasMapdlKey)	- creates mapdl object
mapdl.executeCommandToString("/prep7")	- exemplary command

D.2 Execution Script in Python

```
from source.factory.factory import Factory

# creation of analysis directories
json_file_directory = "C:\\skew_quad_analysis"
json_filename = "input.json"
factory = Factory(json_file_directory, json_filename)

########################################################
# DEFINITION OF INITIAL INSTANCES TO RUN THE PROGRAMME
########################################################

ans = factory.get_ansys_class(factory)
mat = factory.get_material_properties_class(factory)
mag = factory.get_magnetic_map_class(factory)
v_quench = factory.get_quench_velocity_class()

#########################
# GENERAL PRE-PROCESSOR #
#########################

# definition of initial magnetic field,
# initial material properties and coil geometry
preprocessor = factory.get_preprocessor_class(
    mat_props=mat, ansys_commands=ans, factory=factory)
preprocessor.create_ansys_input_variable_file()
preprocessor.define_material_properties(
    magnetic_map=mag.im_short_mag_dict)
preprocessor.define_geometry()
# since geometry is created, Python starts mapping procedure
coil_geo = factory.get_geometry_class(factory)
preprocessor.include_class_geometry_in_class_instance(
    class_geometry=coil_geo)
# input circuit creator
circuit = factory.get_circuit_class(
    ansys_commands=ans, class_geometry=coil_geo, factory=factory)

############################
# INITIAL TIME STEP SOLVER #
############################
```

```
# creation of instances necessary to run the solver
ic_temperature = factory.get_initial_temperature_class(
    ansys_commands=ans, factory=factory,
    class_geometry=coil_geo, mat_props=mat)
solver = factory.get_solver_type(
    mat_props=mat, mag_map=mag,
    ansys_commands=ans, class_geometry=coil_geo,
    circuit=circuit, ic_temperature_class=ic_temperature,
    factory=factory)
# adjustment of resistive material
# properties in initially quenched zone
solver.create_ic_temperature_profile()
postprocessor = factory.get_postprocessor_class(
    class_geometry=coil_geo, ansys_commands=ans,
    v_quench=v_quench, solver=solver, factory=factory)
postprocessor.check_quench_state()
postprocessor.plot_quench_state_in_analysis()
preprocessor.adjust_material_properties_in_quenched_zone(
    postprocessor)
solver.end_of_time_step()

ans.enter_solver()
solver.set_circuit_bcs()
solver.set_initial_temperature()
solver.set_time_step_temperature()
solver.set_solver_boundary_conditions()
solver.enter_solver_settings()
solver.set_time_step()
solver.solve()

###################################
# INITIAL TIME STEP POST-PROCESSOR #
###################################
ans.enter_postprocessor()
# write down temperature profile, plot temperature
postprocessor.get_temperature_profile()
postprocessor.get_current()
postprocessor.check_quench_state_heat_balance()
# estimate resistance in ansys and python, plot resistance
postprocessor.estimate_coil_resistance()
postprocessor.estimate_quench_velocity()
postprocessor.check_quench_state_quench_velocity()
postprocessor.plot_quench_state_in_analysis()
ans.finish()

ans.enter_preprocessor()
postprocessor.update_magnetic_field()
preprocessor.adjust_material_properties_in_quenched_zone(
    postprocessor)
preprocessor.adjust_material_properties_in_non_quenched_zone(
    postprocessor, circuit)
# QDS verifying the quench state
preprocessor.start_discharge_after_qds_switch(
```

```
    circuit, postprocessor)
preprocessor.adjust_nonlinear_inductance(circuit)
solver.end_of_time_step()

############################
# FURTHER TIME STEP SOLVER #
############################
for i in range(2, len(solver.time_step_vector)):
    ans.enter_solver()
    solver.restart_analysis()
    solver.set_time_step()
    solver.set_time_step_temperature()
    solver.solve()

###################################
# FURTHER TIME STEP POST-PROCESSOR #
###################################
    ans.enter_postprocessor()
    # write down temperature profile
    postprocessor.get_temperature_profile()
    postprocessor.get_current()
    postprocessor.check_quench_state_heat_balance()
    # estimate resistance in ansys and python, plot resistance
    postprocessor.estimate_coil_resistance()
    postprocessor.estimate_quench_velocity()
    postprocessor.check_quench_state_quench_velocity()
    postprocessor.plot_quench_state_in_analysis()

    ans.enter_preprocessor()
    postprocessor.update_magnetic_field()
    preprocessor.adjust_material_properties_in_quenched_zone(
        postprocessor)
    preprocessor.adjust_material_properties_in_non_quenched_zone(
        postprocessor, circuit)

    # QDS verifying the quench state
    preprocessor.start_discharge_after_qds_switch(
        circuit, postprocessor)
    preprocessor.adjust_nonlinear_inductance(circuit)
    solver.end_of_time_step()

postprocessor.make_gif()
ans.save_analysis()
ans.terminate_analysis()
factory.copy_ansys_analysis_files_to_output_results_directory()
```

D.3 Quench Front Position Assignment Algorithm

```
class SearchNodes(object):
```

```
@staticmethod
def search_node(position, coil_length, epsilon=0.000001):
    """
    Uses binary search to find initial quench node
    :param position: initial quench position in meters
    :param coil_length: coil length numpy array;
                        1st column - node number,
                        2nd column position in meters
    :param epsilon: searching error as float (optional)
    :return: node number as integer
    """

    # check if position is in the coil region
    if position > coil_length[len(coil_length) - 1, 1]:
        raise Exception("ERROR - search_init_node -
            init. quench position is further than
            the coil length.")
    elif position < coil_length[0, 1]:
        raise Exception("ERROR - search_init_node -
            init. quench position is below the start
            of the coil length.")
    guess_epsilon = None
    left = 0
    right = len(coil_length)-1
    guess = round((left + right)/2)

    while abs(coil_length[guess, 1] - position) > epsilon:

        if guess == guess_epsilon:
            # compare which border is closer
            if abs(coil_length[right, 1] - position) < \
            abs(coil_length[left, 1] - position):
                return right+1
            else:
                return left+1
        if coil_length[guess, 1] < position:
            left = guess
        else:
            right = guess
        guess_epsilon = guess
        guess = round((left + right)/2)

    # counting nodes in ansys starts from 1
    return guess + 1
```

D.4 Quench Detection Algorithm

```
class QuenchDetect(object):

def __init__(self, class_geometry, npoints, mat_props):
    """
```

```
    :param coil_length:
    :param directory: analysis_directory as string
    :param npoints: number of nodes in geometry as integer
    """

    self.mat_props = mat_props
    self.geo = class_geometry
    self.coil_length = self.geo.coil_geometry
    self.npoints = npoints

def detect_quench(self, input_quench_front_vector,
                  temperature_profile, magnetic_field_map):
    """
    Main function for quench detection
    :param input_quench_front_vector:
           list of QuenchFront objects
    :param temperature_profile:
           file with nodal temperature as string
    :param magnetic_field_map:
           magnetic field winding map as dictionary
    :return: list of new quench fronts positions in meters
    """
    input_quench_front_vector_sorted =
        QuenchDetect.sort_input_quench_front_vector(
            input_quench_front_vector)
    temperature_profile_sliced =
        QuenchDetect.slice_temperature_profile(
        input_quench_front_vector=input_quench_front_vector_sorted,
        temperature_profile=temperature_profile)
    temp_profile_windings_sliced =
        self.slice_temperature_profile_with_respect_to_winding_numbers(
        temperature_profile_sliced)
    temp_profile_sliced_quench_detection =
        self.find_quenched_nodes(temp_profile_windings_sliced,
                                 magnetic_field_map)
    new_quenched_fronts_before_rejection =
        QuenchDetect.find_new_quench_fronts(
        quenched_nodes=temp_profile_sliced_quench_detection)
    new_quench_fronts_list =
        QuenchDetect.create_new_quench_fronts_list(
        input_quench_front_vector=input_quench_front_vector_sorted,
        new_quench_fronts=new_quenched_fronts_before_rejection)
    new_quench_fronts_list_without_repetitions =
        QuenchDetect.find_repetitive_fronts(new_quench_fronts_list)
    new_quench_fronts_sorted =
        QuenchDetect.define_final_new_quench_fronts(
        fronts_list=new_quench_fronts_list_without_repetitions,
        new_quench_fronts_nodes=new_quenched_fronts_before_rejection)
    quench_fronts_position =
        QuenchDetect.search_quench_length(
        coil_length=self.coil_length,
        new_quench_fronts_nodes=new_quench_fronts_sorted)
    return quench_fronts_position
```

```
@staticmethod
def sort_input_quench_front_vector(input_quench_front_vector):
    """
    :param input_quench_front_vector: list of QuenchFront objects
    :return: list of QuenchFront objects sorted
             from lowest x_down to highest x_down
    """
    return sorted(input_quench_front_vector,
           key=lambda QuenchFront: QuenchFront.x_down)

@staticmethod
def slice_temperature_profile(
    input_quench_front_vector, temperature_profile):
    """
    :param input_quench_front_vector: list of QuenchFront objects
    :param temperature_profile: temperature profile as numpy array
    :return: list of non-quenched zones as numpy arrays
    """
    node_down_list = []
    for items in input_quench_front_vector:
        node_down_list.append(items.x_down_node)
    node_up_list = []
    for items in input_quench_front_vector:
        node_up_list.append(items.x_up_node)

    sliced_temperature_profile = []
    if len(input_quench_front_vector) != 0:
        for i in range(len(input_quench_front_vector) + 1):
            if i == 0:
                sliced_temperature_profile.append(
                temperature_profile[0:(node_down_list[i]-1), :])
            elif i == len(input_quench_front_vector):
                sliced_temperature_profile.append(
                temperature_profile[
                (node_up_list[i-1]):(len(temperature_profile)), :])
            else:
                sliced_temperature_profile.append(
                temperature_profile[
                (node_up_list[i-1]):(node_down_list[i]-1), :])
        print("Number of zones to check for quench is: {}"
        .format(len(sliced_temperature_profile)))
    else:
        sliced_temperature_profile.append(temperature_profile)
    return sliced_temperature_profile

def slice_temperature_profile_with_respect_to_winding_numbers(
    self, sliced_temp_profile):
    """
    Returns list of dictionaries; each dictionary divides
    a sliced temperature profile into sub-regions
    with respect to winding number
    :param sliced_temp_profile:
```

```
            list of non-quenched zones
            (list of nodes) as numpy arrays
        :return: list of dictionaries;
                 each dictionary assigns node numbers
                 to different winding number
        """
        sliced_temp_wind_profile_list = []
        for slice_pro in sliced_temp_profile:
            if len(slice_pro) != 0:
                dict_slice = {}
                slice_node = slice_pro[:, 0]
                winding_temp_dict = self.geo.
                retrieve_winding_numbers_and_quenched_nodes(
                    x_down_node=slice_node[0],
                    x_up_node=slice_node[len(slice_node)-1])
                for key in winding_temp_dict:
                    value = winding_temp_dict[key]
                    temp_profile_winding = slice_pro[
                    np.where(slice_pro[:, 0] >= int(value[0]))]
                    temp_profile_winding = temp_profile_winding[
                    np.where(temp_profile_winding[:, 0] <= int(value[1]))]
                    dict_slice[key] = temp_profile_winding
                sliced_temp_wind_profile_list.append(dict_slice)
        return sliced_temp_wind_profile_list

    def find_quenched_nodes(
        self, sliced_temp_wind_profile_list, magnetic_map_dict):
        """
        Returns list of quenched nodes by counting
        critical temperature dependent on magnetic field at each winding
        :param sliced_temp_wind_profile_list: list of dictionaries
        :param magnetic_map_dict: magnetic map as dictionary
                assigning different magnetic field to each winding
        :return: list of newly quenched nodes (list of integers)
        """
        quenched_nodes = []
        for item in sliced_temp_wind_profile_list:
            for key in item:
                array = item[key]
                mag_field = magnetic_map_dict[key]
                critic_temp = self.mat_props.
                    critical_current_density.critical_temperature(
                    magnetic_field=mag_field)
                temporary_quenched_nodes = array[
                    np.where(array[:, 1] >= critic_temp)]
                if len(temporary_quenched_nodes) != 0:
                    quenched_nodes.append(temporary_quenched_nodes)
        return quenched_nodes

    @staticmethod
    def find_new_quench_fronts(quenched_nodes):
        """
        :param quenched_nodes: list quenched
```

```
                            nodes as numpy arrays
        :return: list of lists in each of which
                 there is lower and upper node of new quench fronts
        """
        new_quench_fronts = []
        for quenched_nodes_sliced_profiles in quenched_nodes:
            x_up_node = 0
            x_down_node = 0
            while x_up_node < len(quenched_nodes_sliced_profiles):
                while x_up_node < len(quenched_nodes_sliced_profiles)-1
                    and quenched_nodes_sliced_profiles[x_up_node+1][0] - \
                    quenched_nodes_sliced_profiles[x_up_node][0] == 1:
                        x_up_node += 1
                new_quench_fronts.append([
                    int(quenched_nodes_sliced_profiles[x_down_node][0]),
                    int(quenched_nodes_sliced_profiles[x_up_node][0])])
                x_down_node = x_up_node + 1
                x_up_node += 1
        return new_quench_fronts

    @staticmethod
    def create_new_quench_fronts_list(
        input_quench_front_vector, new_quench_fronts):
        """
        :param input_quench_front_vector:
            list of QuenchFront objects
        :param new_quench_fronts: list of lists
            in each of which there is lower
            and upper node of new quench fronts
        :return: list of indices of new quench fronts
            which are directly neighboring with existing quench fronts
        """
        initial_quench_fronts = []
        for items in input_quench_front_vector:
            initial_quench_fronts.append(
                [items.x_down_node, items.x_up_node])
        fronts_to_delete = []
        for i in range(len(initial_quench_fronts)):
            for j in range(len(new_quench_fronts)):
                if abs(initial_quench_fronts[i][0]
                    - new_quench_fronts[j][1]) == 1:
                    fronts_to_delete.append(j)
        for i in range(len(initial_quench_fronts)):
            for j in range(len(new_quench_fronts)):
                if abs(initial_quench_fronts[i][1]
                    - new_quench_fronts[j][0]) == 1:
                    fronts_to_delete.append(j)
        return fronts_to_delete

    @staticmethod
    def find_repetitive_fronts(duplicate):
        """
        Removes repeating indices in the list of indices
```

```
        :param duplicate: list of indices of new quench
            fronts which are directly neighboring
            with existing quench fronts
        :return: list of indices of new quench
            fronts to delete without repetitions
        """
        final_list = []
        for num in duplicate:
            if num not in final_list:
                final_list.append(num)
        return final_list

    @staticmethod
    def define_final_new_quench_fronts(
        fronts_list, new_quench_fronts_nodes):
        """
        :param fronts_list: list of indices of new
            quench fronts to delete without repetitions
        :param new_quench_fronts_nodes: list of lists in each
            of which there is lower and upper node of new quench fronts
        :return: list of new quench fronts without the ones
            neighboring with the existing ones
        """
        fronts_list.sort()
        fronts_list.reverse()
        print("Number of quench fronts faster
            than current quench fronts: {}".format(len(fronts_list)))
        for i in range(len(fronts_list)):
            new_quench_fronts_nodes.remove(
            new_quench_fronts_nodes[fronts_list[i]])
        print("New quench fronts [nodes]:
            {}".format(new_quench_fronts_nodes))
        return new_quench_fronts_nodes

    @staticmethod
    def search_quench_length(coil_length, new_quench_fronts_nodes):
        """
        :param coil_length: length of coil for each node as numpy array
        :param new_quench_fronts_nodes: list of final new quench fronts
        :return: list of position of new quench fronts
        """
        new_quench_node_list = new_quench_fronts_nodes
        new_quench_length_list = []
        for sublist in new_quench_node_list:
            new_x_down = coil_length[sublist[0]-1][1]
            new_x_up = coil_length[sublist[1]-1][1]
            print("New quench zone; x_down= {},
                x_up= {}".format(new_x_down, new_x_up))
            new_quench_length_list.append([new_x_down, new_x_up])
        return new_quench_length_list
```

Bibliography

[1] (28.12.2019). LHC Schematic, [Online]. Available: http://cds.cern.ch/record/842611.

[2] (05.01.2020). CERN, [Online]. Available: https://home.cern/about.

[3] L. Evans, *The Large Hadron Collider: a marvel of technology*, 2nd ed. EPFL Press, 2018.

[4] M. Maciejewski, "Co-simulation of transient effects in superconducting accelerator magnets," Ph.D. dissertation, Lodz University of Technology, 2018.

[5] (20.01.2020). HL-LHC Project, [Online]. Available: https://project-hl-lhc-industry.web.cern.ch/.

[6] K.-H. Mess, P. Schmuser, and S. Wolff, *Superconducting accelerator magnets*. World Scientific, 1996.

[7] (03.01.2020). LHC Machine Outreach, [Online]. Available: https://lhc-machine-outreach.web.cern.ch/lhc-machine-outreach/components/cable.htm.

[8] (28.12.2019). CERN, [Online]. Available: http://cds.cern.ch/record/842611.

[9] T. Salmi, "Optimization of quench protection heater performance in high-field accelerator magnets through computational and experimental analysis," Ph.D. dissertation, Tampere University of Technology, 2015.

[10] E. Ravaioli, "CLIQ. A new quench protection technology for superconducting magnets," Ph.D. dissertation, University of Twente, 2015.

[11] F. Bordry and H. Thiesen. (20.01.2020). LHC inner triple powering strategy, [Online]. Available: https://accelconf.web.cern.ch/AccelConf/p01/PAPERS/ROPB012.PDF.

[12] M. Prioli, T. Salmi, B. Auchmann, L. Bortot, M. Maciejewski, A. Verweij, B. Caiffi, S. Farinon, C. Lorin, M. Segreti, A. M. Fernandez, and J. Munilla, "The CLIQ quench

protection system applied to the 16 T FCC-hh dipole magnets," *IEEE Transactions on Applied Superconductivity*, vol. 29, no. 8, pp. 1–9, Dec 2019.

[13] M. Maciejewski. (04.01.2020). STEAM Architecture, [Online]. Available: https://indico.cern.ch/event/808547/contributions/3367227/.

[14] D.A. May, M. Frehner, M. Afanasiev, P. Sanan. (20.01.2020). Introduction to Finite Element Modelling in Geosciences, [Online]. Available: http://jupiter.ethz.ch/~gfdteaching/femblockcourse/2015/lectures/fem-class-ETHZ2015.pdf.

[15] Y.-A. Cengel, *Heat transfer – a practical approach.* Mc Graw Hill Education, 2002.

[16] G. Manfreda, "Review of ROXIE's material properties database for quench simulation," CERN, Tech. Rep., March 2018.

[17] S. McIntosh, K. Cave-Ayland, A. Holmes, S. Zheng, B. Merrill, C. Jong, N. Mitchell, and K. Hamada, "Qualified analysis of unmitigated quench integrated ANSYS modelling," presented at CERN, April 2017.

[18] K. Hamada, N. Mitchell, A. Foussat, S. McIntosh, A. Holmes, K. Cave-Ayland, A. Ash, F. Domptail, S. Zheng, E. Surrey, and N. Taylor, "Analysis of ITER magnet in safety-related fault condition — case study for PF3," *IEEE Transactions on Applied Superconductivity*, vol. 26, no. 4, pp. 1–5, June 2016.

[19] A. Shaji, "Theory and modelling of quench in cable-in-conduit superconducting magnets," Ph.D. dissertation, Plasma Fusion Center Massachusetts Institute of Technology Cambridge, April 1994.

[20] M. Mentink and M. Maciejewski, *STEAM-BBQ user manual v1.0*, CERN TE-MPE-PE, 2019, EDMS Nr: 2159478.

[21] D. Paudel, "Quench simulation of superconducting magnets with commercial multiphysics software," Master's thesis, Department of Applied Mechanics of Aalto University School of Engineering, June 2015.

[22] G. Apollinari, I. Bejar Alonso, O. Bruning, P. Fessia, M. Lamont, L. Rossi, and L. Tavian, "High-Luminosity Large Hadron Collider technical design report V.0.1," CERN, Tech. Rep., April 2017.

[23] M. Prioli. Private communication, May-December 2019.

[24] S. Mariotto. Private communication, August-December 2019.

[25] A. Davies. (18.11.2019). Material properties data for heat transfer modelling in Nb_3Sn magnets, [Online]. Available: https://www.illinoisacceleratorinstitute.org/2011%20Program/student˙papers/Andrew˙Davies.pdf.

[26] ANSYS Inc., *Element Reference*, 2009.

[27] (24.01.2020). COMSOL Webpage, [Online]. Available: https://www.comsol.com.

[28] ANSYS Inc., *ANSYS Mechanical APDL Command Reference*, 2009.

[29] M. Mentink. Private communication, July 2019.

[30] M.N. Wilson, *Superconducting magnets*. Clarendon Press, 1987.

[31] S. Mcintosh. Private communication, February-May 2019.

[32] M. Wilczek. (15.01.2020). STEAM Ansys Modelling, [Online]. Available: https://gitlab.cern.ch/steam/steam-ansys-modelling.

[33] (16.01.2020). Arisawa Materials, [Online]. Available: http://www.arisawa.co.jp/en/products/r˙m˙pro02.html.

[34] A. Verweij, *CUDI manual*, CERN, 2007.

[35] M. S. Lubell, "Empirical scaling formulas for critical current and critical field for commercial Nb-Ti," *IEEE Transactions on Magnetics*, vol. 19, no. 3, pp. 754–757, May 1983.