Master's thesis
Materials research
Computational materials physics

Machine learning in nondestructive estimation of neutron-induced reactor pressure vessel embrittlement

Sonja Grönroos

November 26, 2020

Supervisors: Antti Kuronen, University Lecturer
Jari Rinta-aho

Examiners: Prof. Flyura Djurabekova
Antti Kuronen, University Lecturer

University of Helsinki
Faculty of Science

P.O. Box 64 (Gustaf Hällströmin katu 2a)
00014 Helsingin yliopisto

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Koulutusohjelma — Utbildningsprogram — Degree programme
Faculty of Science	Materials research Computational materials physics

Tekijä — Författare — Author

Sonja Grönroos

Työn nimi — Arbetets titel — Title

Machine learning in nondestructive estimation of neutron-induced reactor pressure vessel embrittlement

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidantal — Number of pages
Master's thesis	November 26, 2020	72

Tiivistelmä — Referat — Abstract

Several nuclear power plants in the European Union are approaching the ends of their originally planned lifetimes. Extensions to the lifetimes are made to secure the supply of nuclear power in the coming decades. To ensure the safe long-term operation of a nuclear power plant, the neutron-induced embrittlement of the reactor pressure vessel (RPV) must be assessed periodically. The embrittlement of RPV steel alloys is determined by measuring the ductile-to-brittle transition temperature (DBTT) and upper-shelf energy (USE) of the material. Traditionally, a destructive Charpy impact test is used to determine the DBTT and USE.

This thesis contributes to the NOMAD project. The goal of the NOMAD project is to develop a tool that uses nondestructively measured parameters to estimate the DBTT and USE of RPV steel alloys. The NOMAD Database combines data measured using six nondestructive methods with destructively measured DBTT and USE data. Several non-irradiated and irradiated samples made out of four different steel alloys have been measured. As nondestructively measured parameters do not directly describe material embrittlement, their relationship with the DBTT and USE needs to be determined. A machine learning regression algorithm can be used to build a model that describes the relationship.

In this thesis, six models are built using six different algorithms, and their use is studied in predicting the DBTT and USE based on the nondestructively measured parameters in the NOMAD Database. The models estimate the embrittlement with sufficient accuracy. All models predict the DBTT and USE based on unseen input data with mean absolute errors of approximately 20 °C and 10 J, respectively. Two of the models can be used to evaluate the importance of the nondestructively measured parameters. In the future, machine learning algorithms could be used to build a tool that uses nondestructively measured parameters to estimate the neutron-induced embrittlement of RPVs on site.

Avainsanat — Nyckelord — Keywords

Embrittlement, machine learning, nondestructive evaluation, reactor pressure vessel

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — Övriga uppgifter — Additional information

Acknowledgements

I would like to thank my supervisors Jari Rinta-aho and University Lecturer Antti Kuronen for their advice and feedback along the way. I would like to thank my employer VTT Technical Research Centre of Finland Ltd for offering me the chance to study this interesting topic. For the most part, writing this thesis was so engaging it did not feel like working. The Structural Integrity Team offered me an irreplaceable environment to see and learn the ways of research.

I would also like to express my gratitude towards the sponsor of this project. This M.Sc. thesis was supported by the NOMAD project. This project has received funding from the Euratom research and training program 2014–2018 under grant agreement № 755330. Several partners have contributed to the NOMAD project, and for that I would like to thank the Fraunhofer Institute for Nondestructive Testing, the Belgian Nuclear Research Centre, VTT Technical Research Centre of Finland Ltd, the Swiss Association for Technical Inspections Group, European Research and Project Office GmbH, Coventry University, HEPENDIX Technical Service Ltd, the Hungarian Academy of Sciences Centre for Energy Research, the Paul Scherrer Institute and TECNATOM.

I would like to thank all my friends and other fellow students at the Kumpula Campus for creating an inspiring atmosphere to study physics in. Last but not least I thank my family for their endless support and encouragement.

Contents

Abbreviations **ix**

1 Introduction **1**

2 Neutron irradiation **3**
2.1 Nuclear reactors . . . 3
2.1.1 Components and materials . . . 4
2.1.2 Fission and extra neutrons . . . 6
2.2 Embrittlement of reactor pressure vessels . . . 7
2.2.1 Charpy impact test . . . 8
2.2.2 Ductile-to-brittle transition curve . . . 9
2.2.3 Neutron radiation effects in metals . . . 10

3 Nondestructive evaluation **15**
3.1 The NOMAD project . . . 15
3.2 Samples . . . 16
3.3 Methods . . . 17
3.3.1 The Micromagnetic Inductive Response and Barkhausen Emission method . . . 18
3.3.2 Micromagnetic Multiparameter Microstructure and Stress Analysis 20
3.3.3 Magnetic adaptive testing . . . 24
3.3.4 Direct Current-Reversal Potential Drop . . . 26
3.3.5 Thermoelectric power measuring method . . . 27
3.3.6 Ultrasonic method . . . 28

4 Machine learning **30**
4.1 Basics . . . 30
4.2 Supervised machine learning . . . 31
4.2.1 Workflow . . . 31
4.3 Regression algorithms . . . 36

4.3.1 Linear models . . . 36
4.3.2 Training models with gradient descent . . . 38
4.3.3 Support vector machines . . . 39
4.3.4 k-nearest neighbors . . . 41
4.3.5 Ensembles . . . 42
4.3.6 Decision trees and random forests . . . 42
4.3.7 Artificial neural networks . . . 44

5 Models built using the NOMAD Database 48
5.1 Data . . . 48
5.1.1 Preprocessing . . . 49
5.2 Models . . . 50
5.2.1 Ridge regression . . . 50
5.2.2 Support vector regression . . . 51
5.2.3 k-nearest neighbors . . . 51
5.2.4 Boosted decision tree . . . 51
5.2.5 Extra-trees regressor . . . 52
5.2.6 Artificial neural network . . . 52

6 Results 53
6.1 Test accuracy . . . 53
6.2 Model stability . . . 57
6.3 Feature importance . . . 58

7 Conclusions and discussion 59

Appendix A Identification, irradiation and destructively measured parameters in the NOMAD Database 61

Appendix B Nondestructively measured parameters in the NOMAD Database 62

Appendix C Source code of the models 64

References 66

Abbreviations

3MA Micromagnetic Multiparameter Microstructure and stress Analysis

Adam Adaptive moment estimation

***k*NN** k-Nearest Neighbors

ANN Artificial Neural Network

BDT Boosted Decision Tree

CALLISTO Capability for Light-water Irradiation in Steady-state and Transient Operation

CHIVAS Callisto Hot Irradiation of Vessel Alloy Steels

CU Coventry University

DBTT Ductile-to-Brittle Transition Temperature

DCRPD Direct Current-Reversal Potential Drop

dpa displacements per atom

ETR Extra-Trees Regressor

Euratom European Atomic Energy Community

FhG-IZFP Fraunhofer Institute for Nondestructive Testing

LTO Long-Term Operation

MAE Mean Absolute Error

MAT Magnetic Adaptive Testing

MIRBE Micromagnetic Inductive Response and Barkhausen Emission

MSE Mean Squared Error

MTA-EK Hungarian Academy of Sciences Centre for Energy Research

NaN Not a Number

NDE Nondestructive Evaluation

NOMAD Nondestructive Evaluation System for the Inspection of Operation-Induced Material Degradation in Nuclear Power Plants

PKA Primary Knock-on Atom

PSI Paul Scherrer Institute

PWR Pressurized Water Reactor

ReLU Rectified Linear Unit

RPV Reactor Pressure Vessel

RR Ridge Regression

RSS Residual Sum of Squares

SELU Scaled Exponential Linear Unit

SVR Support Vector Regression

TDE Threshold Displacement Energy

TEPMM Thermoelectric Power Measuring Method

USE Upper-Shelf Energy

VTT VTT Technical Research Centre of Finland Ltd

1. Introduction

The European Union has set a goal to be climate-neutral by 2050 [1]. Nuclear energy plays a significant role in reaching this goal and generally in the fight against climate change. In 2018, 26 % of the net electricity and 47 % of the low-carbon electricity generated in the European Union came from nuclear power [2]. The aging of nuclear reactor materials has become a topic of interest, as the originally planned lifetimes of today's nuclear power plants were approximately 40 years, and at the beginning of 2020 the average age of the nuclear power plants in the European Union was close to 40 years. Extensions to the original lifetimes are made in order to ensure a secure low-carbon power supply in the coming decades. This is known as long-term operation.

However, the long-term operation of nuclear power plants is limited by several factors. One of the most limiting factors is the neutron-induced embrittlement of the reactor pressure vessel (RPV). The RPV is exposed to neutron radiation during operation, and the neutrons cause microstructural changes in the material. This leads to changes in mechanical material parameters, such as an increase in brittleness. Destructive methods, like the Charpy impact test, are traditionally used to determine the neutron-induced embrittlement of RPV steel alloys. The embrittlement is characterized by two mechanical parameters: the ductile-to-brittle transition temperature (DBTT) and upper-shelf energy (USE). Destructive methods are well-known and reliable, but the measurements do not account for material inhomogeneities and are not repeatable.

The development of a nondestructive method for evaluating RPV embrittlement has been the goal of numerous studies during the past decades, without notable success. The currently ongoing NOMAD project, launched in 2017, shares this goal [3]. The NOMAD project strives to develop a software-based tool that can be used to estimate the embrittlement of operating RPVs nondestructively and thus to improve the safety and reliability of aging nuclear power plants. The database used in this thesis has been constructed as a part of the NOMAD project. This NOMAD Database combines destructively measured DBTT and USE data with several nondestructively measured parameters. The nondestructive methods are based on relationships between changes in material microstructure, mechanical properties, and magnetic, electric and elastic material properties. The measurements have been taken from samples made out of four different RPV steel alloys with varying levels of embrittlement.

A machine learning regression algorithm can be used to find the relationship between the embrittlement, which is described by the DBTT and USE data, and non-destructively measured parameters. The algorithm automatically builds a model based on a given training data set, and the model can then be used to predict output values for new, unseen input values. The goal of this thesis is to study how different kinds of machine learning regression algorithms can be used to predict the embrittlement of RPV steel alloys based on the nondestructively measured parameters in the NOMAD Database. Six models are trained with data from the NOMAD Database, and their use in predicting the DBTT and USE is evaluated.

The structure of this thesis is as follows. Chapter 2 explains the structure of a nuclear reactor and the origins and effects of the neutron radiation that targets the RPV. Additionally, the destructive Charpy impact test and the DBTT and USE are introduced. Chapter 3 presents the NOMAD project and the six nondestructive methods that have been used to compose the NOMAD Database. After this, Chapter 4 describes the basic principles of supervised machine learning and introduces six machine learning regression algorithms. Chapter 5 presents the models trained with data from the NOMAD Database. Chapter 6 reports the results of this study, and Chapter 7 finalizes this thesis with conclusions and discussion.

2. Neutron irradiation

This chapter begins with an overview of the main components that make up a nuclear reactor. Additionally, the fission reaction and its products, including the neutrons that are responsible for the embrittlement of the reactor pressure vessel (RPV), are introduced. This is followed by a description of the Charpy impact test and how it is used to determine the embrittlement of a material destructively. The final section of this chapter covers the microstructural changes in metals that are caused by neutron irradiation and which explain the embrittlement.

2.1 Nuclear reactors

Numerous nuclear power plants of different designs are in operation around the globe. Nuclear reactors can be categorized into generations based on their age and type [4]. Now mainly decommissioned Generation I reactors are early prototype reactors, which were built in the 1950s−1970s. Generation II reactors were built in the 1970s−1990s and consist mostly of boiling and pressurized water reactors (PWRs). The reactors built during the 1990s−2010s use reactor designs with some improvements to Generation II reactors and are categorized as Generation III reactors. Newer reactor designs are categorized as either Generation III+ or Generation IV reactors.

As of 2020, there are over a hundred operating nuclear reactors in the European Union, and a majority of these reactors were built in the 1970s or 1980s [5]. This means that most of the nuclear reactors operating in the European Union are Generation II reactors. Because Generation II reactors were generally designed to have a lifetime of 40 years, many of the reactors in the European Union will approach the ends of their initially planned lifetimes during the 2020s. If requirements for safe operation are met, the lifetimes of operating reactors can be prolonged beyond their initially planned lifetimes. A common extension to the lifetime is 20 years. When a nuclear power plant operates longer than originally was planned, it is in long-term operation (LTO). Keeping the already existing nuclear power plants in operation is economically more preferable than the construction of new power plants. The LTO of nuclear power plants is regulated and involves extensive safety assessments [6].

Without the LTO of Generation II reactors, the nuclear power capacity of the European Union would strikingly decrease in the coming years [6]. Many European countries have accepted LTO as the general strategy to maintain their nuclear electricity production rate. A nuclear power plant in LTO in Finland is used as an example in the next section. The Loviisa power plant is operated by Fortum Oyj and produces electricity using two VVER-440/213 type PWR units, Loviisa 1 and Loviisa 2. The reactors are of Soviet design, and Loviisa 1 has been in operation since 1977 and Loviisa 2 since 1980. The two units are thus categorized as Generation II reactors. Originally, the reactors had operating licenses for 30 years, but the operating licenses of both units were extended by 20 years in 2006. Thus, Loviisa 1 and Loviisa 2 have operating licences until 2027 and 2030, respectively [7].

2.1.1 Components and materials

Different types of nuclear reactors are distinguished by the moderator they use. All operating nuclear reactors in the European Union are light-water reactors, which use ordinary water as both moderator and coolant. PWRs are a subtype of light-water reactors, and the majority of the nuclear power plants in operation in the European Union produce electricity using PWRs. The basic structure of a PWR, which is illustrated in Figure 2.1, consists of two distinct loops: the primary loop and the secondary loop. The coolant, pressurized water, circulates through the primary loop and carries the heat away from the reactor core. The water in the primary loop becomes moderately radioactive and is not in direct contact with the water in the secondary loop at any point.

The main component of the primary loop is the RPV, which contains the reactor core. Different types of PWRs have different fuel configurations and operating parameters, and the following values are valid for VVER-440 reactors. The basic design features of VVER reactors are introduced in Ref. [8]. The nuclear fuel, low enriched uranium dioxide, is manufactured into fuel pellets. The pellets are inserted into fuel rods, which are then assembled into fuel assemblies. One fuel assembly contains 126 fuel rods. The cladding of the fuel assemblies is made of zirconium-niobium alloy. Control rods, which can be inserted into the reactor core from above, are used to control the reactivity of the reactor. The control rods consist of neutron-absorbing boron steel and have fuel extensions. The reactor core contains 349 elements, which consist of 312 fuel assemblies and 37 control rods. The water in the primary loop is kept at a high pressure, close to 123 bar, so that it does not boil. As the pressurized water flows through the reactor core, it is heated by the nuclear reaction. Under normal operating conditions, the coolant heats up to 300 °C. The water is then pumped to a steam generator, where a part of the energy produced by the nuclear reaction is transferred

to the secondary loop. The water in the secondary loop is at a lower pressure, and the heat from the primary loop vaporizes the water in the secondary loop. The steam is fed through a steam turbine, which drives an electrical generator that produces electricity. The steam is then condensed back to a liquid and fed again to the steam generator, where it starts the secondary loop again. Respectively, the cooled water in the primary loop enters the reactor core again. The inlet temperature of the coolant is 270 °C.

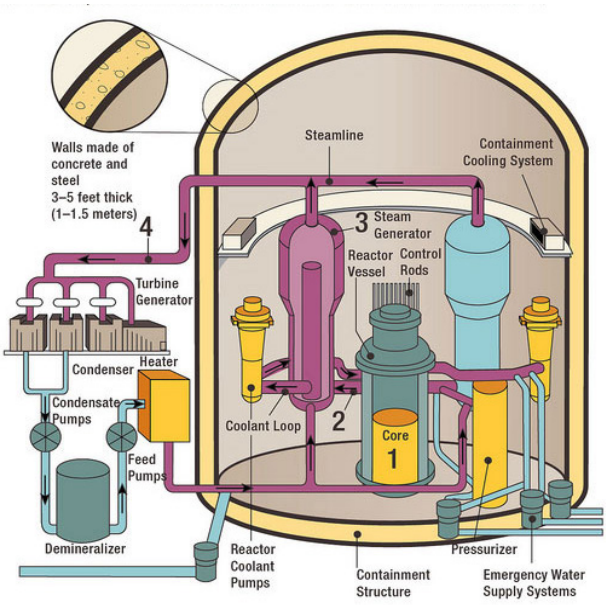

Figure 2.1: A schematic drawing of the basic structure of a pressurized water reactor. The reactor pressure vessel contains the reactor core, which contains the nuclear fuel. A large containment structure is built to enclose the components that could release radioactive substances into the surrounding environment in accidental events. The containment structure also protects the crucial components from external forces. Figure taken from Ref. [9].

Reactor pressure vessels are large cylindrical steel structures with top and bottom heads that are typically hemispherical or rounded. Commonly, RPV walls are composed of forged rings that are joined together by radial welds, but also axial welds can be used [10]. Weld materials have been shown to be more sensitive to neutron irradiation than base materials, and the welds nearest to the beltline region of the RPV are recognized as its weak points, as they are nearest to the reactor core and thus receive the greatest amount of neutron radiation. The RPVs of VVER-440 reactors stand 12 meters tall, and have a diameter of 4 meters and a wall thickness of 15 cm. The RPVs are made out of low-alloy ferritic chromium-molybdenum-vanadium steel (15Ch2MFA) and usually have an austenitic stainless steel cladding on the inside, which protects the RPV from corrosion.

RPVs are practically irreplaceable, so they determine the lifetime of the entire power plant. This means that RPVs must be able to operate safely for a long time period. The integrity of the RPV cannot be compromised during either normal op-

eration or potential accident event conditions. The RPVs of both Loviisa units have separate operating licenses, which have been extended to match the current 50-year operating licenses of the units. The integrity of the RPVs is assessed periodically, and the neutron flux to both RPVs has been reduced by replacing some fuel elements with steel "dummy" elements at the peripheries of the cores, for example [7, 11].

2.1.2 Fission and extra neutrons

A common nuclear fuel is uranium-235. Natural uranium contains approximately 0.7 % of uranium-235, 99 % of uranium-238 and a minuscule amount of uranium-234. In most cases, natural uranium needs to be enriched so that the concentration of uranium-235 becomes higher before it is used as a nuclear fuel. Pressurized heavy water reactors, which moderate neutrons with heavy water, can use natural uranium as fuel. A commonly used fuel in PWRs is uranium dioxide that contains 3−5 % of uranium-235. When the uranium-235 nucleus is hit by a thermal neutron, the neutron is absorbed by the nucleus, which then undergoes neutron-induced fission. Neutrons are classified according to their kinetic energy: a thermal neutron has a kinetic energy of the order of 0.025 eV. Heavier atoms split into lighter elements during fission, and because the lighter daughter nuclei prefer a smaller neutron to proton ratio than the original nucleus, extra neutrons are released during the reaction [12]. A fission reaction of uranium-235 is depicted in Figure 2.2.

The total mean fission energy of uranium-235 is 203 MeV, and it is distributed among the kinetic energy of the daughter nuclei (166 MeV), neutrons (5 MeV), gammas (8 MeV), and the energy of the betas, gammas and anti-neutrinos that are born when the daughter nuclei decay (24 MeV in total) [13]. On average, 2.4 neutrons are produced per uranium-235 fission reaction, so the mean energy of a neutron is ~ 2 MeV. Therefore, the neutrons are fast neutrons, which are neutrons with a kinetic energy greater than 1 MeV.

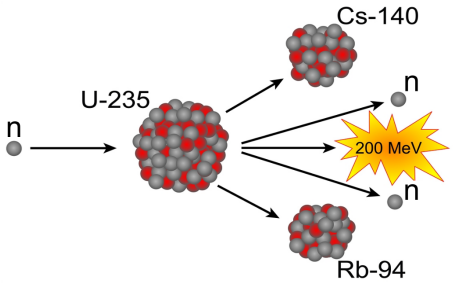

Figure 2.2: When a uranium-235 (U-235) nucleus is hit by a neutron (n), it undergoes neutron-induced fission. The reaction depicted here produces two daughter nuclei, caesium-140 (Cs-140) and rubidium-94 (Rb-94), two neutrons and energy. The neutrons can subsequently induce new fission reactions. Figure adapted from Ref. [14].

The neutrons that are born in the fission reaction can induce new fission reactions and sustain the nuclear chain reaction. The cross-section of the neutron-induced fission of uranium-235 is significantly larger at the energies of thermal neutrons than at the energies of fast neutrons, so a slower thermal neutron is more likely to cause the fission reaction of a uranium-235 atom than a fast neutron. Thus, the fast neutrons born in the fission reactions need to be slowed down. When the fast neutrons collide with light atoms of a medium, such as the hydrogen atoms of water molecules, their kinetic energy is reduced efficiently. This process is called moderation, and this is why the water in the primary loop is known as the moderator. If the reactivity of the reactor is kept constant, only one of the extra neutrons from each fission reaction induces a new reaction.

The use of pressurized water as a moderator causes PWRs to self-regulate and makes them stable. As the nuclear reaction becomes more rapid, the temperature of the water increases and its density decreases. Less dense water slows the neutrons down less efficiently than denser water, and this reduces the reactivity of the reactor. In order to make the reactivity and power of the reactor more controllable, boron is added to the coolant water, often in the form of water-soluble boric acid. Boron is an efficient neutron absorber. The boron concentration of the water is monitored and can be adjusted in response to changes in the state of the reactor. Other elements inside the reactor core also absorb neutrons, such as the uranium-238 atoms in the fuel. This reaction produces plutonium-239, which can undergo neutron-induced fission and act as fuel. However, some neutrons are not absorbed inside the reactor core, and some neutrons will hit the RPV.

2.2 Embrittlement of reactor pressure vessels

The neutrons that hit the RPV cause changes in its microstructure, which lead to changes in mechanical material properties, such as an increase in brittleness. If the RPV material becomes sufficiently brittle, intense loading conditions can lead to fracturing. A particular concern is pressurized thermal shock, when a thermal load is generated on the RPV under pressurized conditions, for instance due to emergency cooling. The thermal shock could initiate the propagation of a pre-existing flaw, which could have unfortunate consequences. The neutron-induced embrittlement of the RPV is one of the most important safety issues that concern nuclear power plants, and the RPV embrittlement must be monitored closely throughout the entire lifespan of a power plant [15].

2.2.1 Charpy impact test

Several destructive methods have been developed to assess material properties and evaluate material embrittlement. Traditional destructive methods include tensile, fatigue, creep, and impact tests. The Charpy impact test or Charpy V-notch test is used to measure the energy that a material absorbs during fracturing, referred to as the absorbed energy [16]. If the absorbed energy of the material is relatively low, the material is brittle. Vice versa, a larger amount of energy absorbed during fracturing indicates that the material is more ductile. Figure 2.3 depicts a schematic diagram of a standard Charpy impact test machine and a sample.

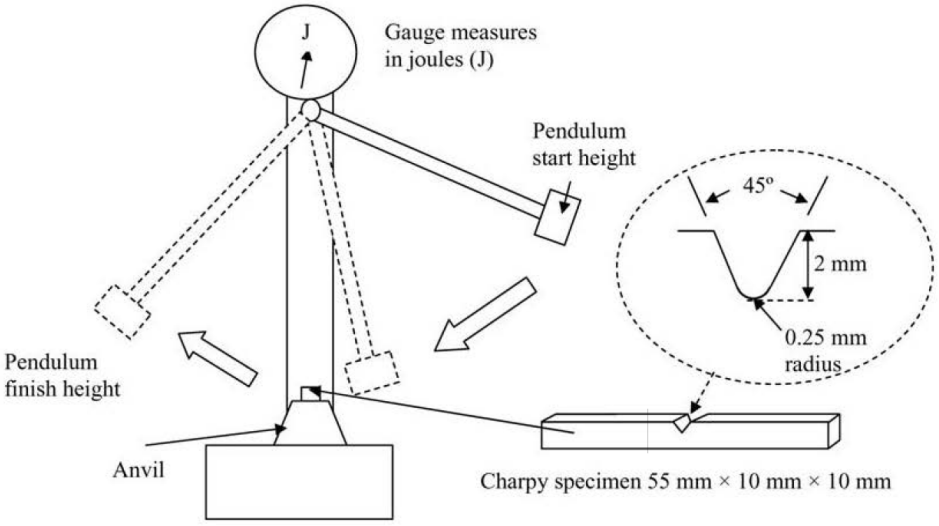

Figure 2.3: A Charpy impact test machine and a standard sample with a V-notch. The sample lies on an anvil at the equilibrium position of the pendulum. The pendulum strikes the sample on the opposite side of the V-notch. The pendulum's finishing height determines the energy absorbed by the sample during impact. The gauge shows the absorbed energy in joules. Figure taken from Ref. [16].

A Charpy impact test machine has a weighted pendulum, which is released from a set start height h. The sample lies at the equilibrium position of the pendulum on an anvil. Standard Charpy samples are bar-shaped and have a standardized notch on one of the larger sides of the sample. The pendulum strikes the sample on the side that is opposite to the side with the notch. The notch acts as a stress concentration zone. According to ISO 148 standard, the dimensions of a Charpy sample should be 55 mm × 10 mm × 10 mm, and a V-shaped notch should be 2 mm deep with a 45-degree angle and a 0.25 mm radius along the base [17]. Also U-shaped or keyhole notches can be used. During impact, the sample absorbs a part of the energy of the pendulum. After impact, the pendulum travels to a finish height $h' < h$. The difference between h and h' equates the absorbed energy.

The Charpy impact test is a popular technique thanks to its standardization, efficiency, and affordability. The test is a regular part of the periodic safety reviews of nuclear power plants. The test is easy and fast to perform, and the sample preparation is straightforward as it follows set standards. On the other hand, the results of the test depend on the dimensions of the sample, such as its thickness. The obvious main drawback of the Charpy impact test is its destructiveness, which motivates the development of alternative, nondestructive methods that could be used to evaluate embrittlement.

2.2.2 Ductile-to-brittle transition curve

The energy that metals with a body-centered cubic structure, such as common RPV metals iron, manganese, and chromium absorb during fracture depends on the temperature. At higher temperatures, these metals are more ductile, meaning that the material is more workable and can absorb more energy without breaking. The absorbed energy of a metal in a ductile state is characterized by the upper-shelf energy (USE). When the metal is cooled to temperatures below a certain threshold, which is defined as the ductile-to-brittle transition temperature (DBTT), it turns brittle. The transition from a ductile state to a brittle state involves a danger: whereas ductile materials fracture slowly and with prior signs of the impending failure, brittle materials fail suddenly with no prior plastic deformation if sufficient stress is applied [18].

A ductile-to-brittle transition curve is obtained by measuring the absorbed energy of identical samples at different temperatures, as illustrated in Figure 2.4. The transition from ductility to brittleness is not instantaneous, but there is a visible transition region between the two states. The DBTT is determined from the transition curve by finding the temperature where the absorbed energy is at a certain predetermined value, which is typically 41 J in the case of RPV steel alloys. The USE is the absorbed energy at temperatures higher than the DBTT, when the material is in a ductile state. At least a dozen measurements, and thus identical samples, are required to plot the transition curve corresponding to one value of radiation exposure. Two transition curves are plotted in Figure 2.4. The first curve has been plotted based on non-irradiated samples, and the second curve has been plotted based on irradiated samples. Both the DBTT and USE have changed due to irradiation. The DBTT is considerably higher and the USE is lower after irradiation. This means that the material has become brittle at higher temperatures and less energy is absorbed during fracture in the ductile state.

The DBTT of the RPV steel alloy is one of the most crucial parameters that are determined during a periodic safety review of a nuclear power plant [10]. The embrittlement of an RPV in an operating reactor is characterized using surveillance

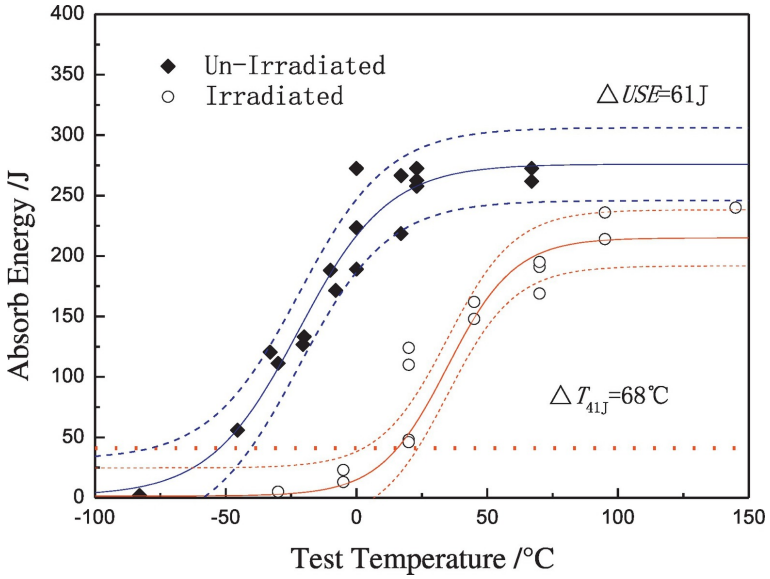

Figure 2.4: The absorbed energy of non-irradiated (diamond-shaped data points) and irradiated (circle-shaped data points) samples made out of A508-3 steel measured at different temperatures using a Charpy impact test. Two ductile-to-brittle transition curves have been plotted based on the measurements. The temperature where the transition curve meets the dotted line, which marks 41 J, is the ductile-to-brittle transition temperature, abbreviated here as $T_{41\mathrm{J}}$. The upper-shelf energy (USE) is the absorbed energy in the ductile state, at temperatures above the $T_{41\mathrm{J}}$. The $T_{41\mathrm{J}}$ increased by 68 °C and the USE decreased by 61 J due to irradiation. Figure taken from Ref. [19].

samples that have been placed inside the RPV before the start up of the reactor. The surveillance samples are located in the beltline region of the RPV, and the samples are made out of the same materials as the base and weld of the RPV in question. The surveillance samples receive radiation during operation, and a fraction of them are taken out to be measured approximately every ten years during annual maintenance outages. Because the location of the surveillance samples receives more neutron radiation than the RPV, the surveillance samples can be used to conservatively evaluate the DBTT and USE of the RPV material. If the DBTT rises above a certain threshold, the RPV and thus the whole power plant has come to the end of its lifetime. On the other hand, if the DBTT is low enough at the end of the originally planned lifetime of the power plant, an extension to the lifetime can be considered.

2.2.3 Neutron radiation effects in metals

Changes in the microstructure of the material are responsible for the observable changes in the mechanical properties, DBTT and USE, of RPVs. The primary cause of these changes is neutron irradiation. Neutron radiation is called indirectly ionizing radiation because neutrons have a neutral charge but are highly penetrating and cause ionization

effects through secondary ionizations. The various effects of neutron irradiation depend on the irradiation conditions, such as irradiation temperature, neutron flux and neutron fluence.

When the neutron flux to the RPV is determined, only the neutrons with energy greater than a specific threshold energy are considered to be sources of damage. A typical threshold energy for RPVs of Western light-water reactors is the energy of fast neutrons, 1 MeV, while the threshold energy is 0.5 MeV for VVER-type reactors due to decades-old differences in Western and Eastern standards [10]. In Western light-water reactors, the fast neutron flux is in the order of 10^{10} neutrons/cm^2/s at the beltline region of the RPV [10]. The neutron flux at the RPV is greater in PWRs than in boiling water reactors because of the smaller distance between the reactor core and the RPV wall. This distance is even smaller than typical in both Loviisa units because the RPVs needed to be transportable by railways.

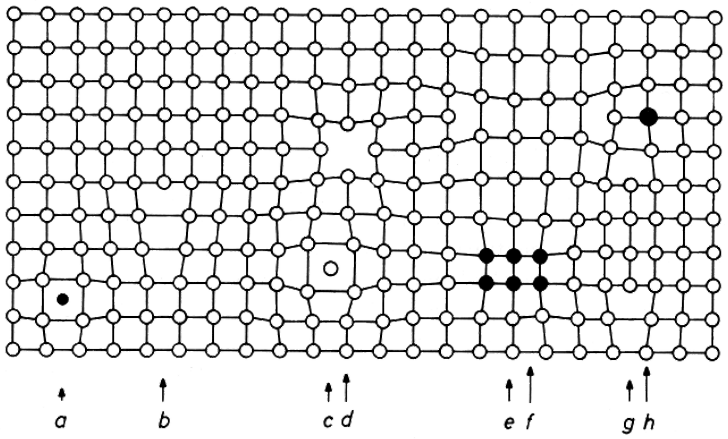

Figure 2.5: An illustration of a lattice structure with different types of defects: a) an interstitial impurity atom, b) an edge dislocation, c) a self interstitial atom, d) a vacancy, e) a precipitate of impurity atoms, f) a vacancy type dislocation loop, g) an interstitial type dislocation loop, and h) a substitutional impurity atom. Figure taken from Ref. [20].

Microstructural effects

The microstructure of metals is normally arranged in a crystal structure, in which each atom occupies a specific site. No perfect crystal lattice exists in nature, and the lattice always contains some defects and solute atoms, which affect the material properties. A lattice structure with multiple kinds of defects is visualized in Figure 2.5. The lattice atoms are bound to each other by electrostatic forces, and neutron radiation induces defects in the microstructure by displacing the atoms. This and the following section are heavily based on Ref. [18], which offers an extensive coverage of the mechanisms of radiation damage in metals. The threshold displacement energy (TDE) is the minimum kinetic energy that is required to be transferred to a lattice

atom in order to displace the atom from its site in the crystal lattice. When a fast neutron interacts with the nucleus of a lattice atom, the two main types of interaction are scattering and absorption.

An elastic scattering between the neutron and the nucleus can be thought of as a collision between two billiard balls, where their total energy is conserved. The neutron transfers a fraction of its kinetic energy to the nucleus. If the transferred energy exceeds the TDE of the atom, the atom is displaced from its position in the lattice. The displaced atom is called a primary knock-on atom (PKA), and it leaves behind a vacancy in the crystal lattice. If the energy of the PKA is large enough, the PKA can displace other atoms and create secondary knock-on atoms. As an example, a 1 MeV neutron can transfer up to approximately 70 keV to a PKA in iron [21]. The average TDE of iron atoms is approximately 40 eV, and as illustrated in Figure 2.6, one energetic neutron can induce a collision cascade in which several atoms are displaced by secondary knock-on atoms. If the energy of the PKA is not large enough to displace other atoms, the PKA becomes an interstitial atom. Generally, a collision cascade results in a region rich with vacancies that is surrounded by a region rich with interstitials. The pairs of vacancies and interstitials are called Frenkel pairs. The damage caused by neutron irradiation can be measured as displacements per atom (dpa), which is the average number of times an atom is displaced during an irradiation period. A typical value for an RPV of a Western light-water reactor at the end of its lifetime is approximately 0.05 dpa [10, 21].

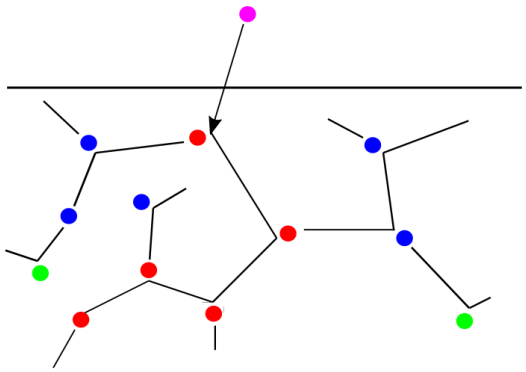

Figure 2.6: An illustration of a collision cascade. An energetic neutron (magenta) collides with an atom, which becomes a primary knock-on atom (red). In turn, the primary knock-on atom creates secondary knock-on atoms (blue), which create tertiary knock-on atoms (green).

Inelastic scattering is similar to elastic scattering, except that the nucleus is left in an excited state and the kinetic energy is not conserved. Excitation is followed by a gamma-ray emission as the excited atom returns to its ground state. The emitted gamma-ray might be energetic enough to subsequently produce more defects. Neutron absorption can lead to the emission of gammas, neutrons, protons, or alpha particles

from the nucleus of the atom. This reaction, in which the neutron or proton number of the atom changes due to neutron absorption, is called transmutation. Neutron-induced transmutation reactions can form impurity gases in the material. For example, the transmutation of boron creates helium. If the transmuted element is radioactive, the process is known as neutron activation. A prominent neutron activation reaction in RPV steel alloys is the transmutation of iron-58 into radioactive cobalt-60.

Most of the point defects annihilate each other at the end of the cascade, and the migration of the surviving fraction of point defects causes changes in material properties. Thermal agitation causes the movement and diffusion of the point defects, and at the operating temperatures of nuclear reactors, which are roughly 300 °C, the point defects can migrate relatively far. In metals, interstitials move faster through the lattice and thus diffuse faster than vacancies. Over time, the neutron-atom interactions lead to the accumulation of point defects in the material.

Dislocations are one-dimensional line defects where the lattice atoms are not aligned in their proper positions. The two main types of dislocations are edge and screw dislocations, and they can move through the lattice by gliding or climbing. Grain boundaries are two-dimensional planar defects that separate the grains of the material. The dislocations and grain boundaries act as sinks in the crystal lattice that attract point defects. The point defects can interact with each other and are annihilated once they interact with a point defect of an opposing type or with a sink. Interstitials are more readily attracted to sinks than vacancies, so irradiation leads to the accumulation of vacancies.

If two point defects of the same kind interact, they will cluster together. There are multiple kinds of clusters, and they vary in size, stability, and mobility. One- or two-dimensional clusters of defects can grow identical to dislocation lines or planar defects known as dislocation loops, and three-dimensional clusters of vacancies form voids in the lattice structure. If a void is filled with the gas atoms born in transmutation, it is known as a bubble.

All metal alloys contain solute atoms, such as copper, nickel, and phosphorus atoms. These solutes can be integrated into the lattice structure as either substitutional or interstitial atoms. The excess number of vacancies in the lattice structure induces the radiation-enhanced diffusion of solutes, most importantly the copper atoms in RPV steel alloys [22]. The copper atoms segregate to voids and form copper-rich precipitates. The other solutes, such as phosphorus, segregate to grain boundaries or surfaces. The enrichment and depletion of solutes in certain areas lead to changes in the local chemistry. In order to control the effects of solutes on the material properties, limitations have been set to the impurity atom contents of RPV materials. For example, the copper content of RPV steel alloys is limited to a maximum of 0.1 wt% [10].

Mechanical effects

The main mechanical effect of neutron irradiation is embrittlement, which is primarily caused by irradiation hardening [21]. The primary irradiation-induced lattice defects that lead to hardening are the point defects, their clusters, and solute precipitates, most dominantly the copper-rich precipitates. Also irradiation-induced phosphorus segregation can cause embrittlement, but it is not a hardening mechanism.

Dislocations in the lattice structure allow rows of atoms to glide over each other and act as carriers of plastic deformation. At higher temperatures the dislocations in metals, especially in metals with a body-centered cubic structure, move with ease and cause the metal to be ductile. These metals are brittle at lower temperatures, where the movement of the dislocations requires more energy. The movement of the dislocations is hindered by the lattice defects, which act as obstacles or pinning sites. This causes an increase in the yield strength of the material. The increase in the yield strength is proportional to the increase in the DBTT [23]. Concurrently, the absorbed energy in the ductile state, USE, decreases.

Neutron irradiation has other effects besides the embrittlement. Voids in the lattice structure can lead to macroscopic volume expansion of the material, which is known as swelling. Not all materials exhibit swelling, and the steel alloys used in nuclear reactors are considered to exhibit minor to no swelling at temperatures below $\sim$ 350 °C. As mechanical stress is applied to the material under irradiation, the dimensions of the material can permanently evolve. The material grows or changes in size depending on the direction of the stress and does not return to its original size or shape after the stress is removed. This is known as irradiation creep. Void swelling and irradiation creep can lead to visible changes in irradiated reactor components, such as the fuel cladding tubes, if they are made out of materials prone to these effects [24].

A combination of sufficient embrittlement and stress could lead to fractures. Embrittled steel alloys commonly fracture transgranularly, but due to the solute segregation to grain boundaries, intergranular fracturing can also occur. Post-irradiation thermal annealing can be used to mitigate the effects of neutron irradiation and to restore the mechanical properties of the material [25]. During annealing, the material is heated up to a temperature higher than operating temperatures for a specific amount of time, commonly several days, and then cooled controllably. Annealing prolongs the lifetime of an RPV, but the embrittlement rate during re-irradiation after annealing does not necessarily behave identically to the original embrittlement rate. Annealing of the RPVs in PWRs is not always possible because of the plant design, but for example, the RPV of Loviisa 1 was successfully annealed in 1996 after 19 years of operation [26].

3. Nondestructive evaluation

The destructive Charpy impact test is a reliable and well-known technique to evaluate RPV embrittlement, as it has been in use for several decades. However, destructive tests combined with the limited number of surveillance samples and LTO can lead to problems. If the surveillance samples run out, the required safety assessments cannot be performed. Destructive tests are not repeatable and assume that the RPV is homogeneous, even though that might not be the case for such a large structure.

As the name suggests, nondestructive evaluation (NDE) means assessing material properties without destroying the evaluated material. Several NDE methods have been developed to serve different purposes, including the characterization of the neutron-induced embrittlement of RPV steel alloys. If a reliable and verifiable relationship between nondestructively measured material properties and embrittlement is found, it could be used to monitor the embrittlement in place of or alongside the traditional destructive tests. This is the idea behind the NOMAD project, which is introduced at the beginning of this chapter and followed by a description of the NDE methods that have been used in the project.

3.1 The NOMAD project

The NOMAD (Nondestructive Evaluation System for the Inspection of Operation-Induced Material Degradation in Nuclear Power Plants) project was launched in 2017 and has received funding from Euratom (European Atomic Energy Community) [3]. The project is multidisciplinary and has ten partners from seven countries. The goal of the NOMAD project is to ensure the safe and reliable LTO of nuclear power plants by developing, demonstrating, and validating a hybrid method for the NDE of RPV material degradation. Six NDE methods have been chosen for the project.

The project is described by three goals. The first goal is to develop a tool that can characterize RPV embrittlement based on nondestructively measured parameters, referred to as the NDE parameters. This NOMAD Tool will be based on supervised machine learning and will be written in open-source Python code. The NOMAD Tool will account for material inhomogeneities and should be able to characterize the whole RPV. Current surveillance procedures do neither. The second goal is to create a multi-

parametric database that combines the data from the NDE measurements with mechanical properties and describes the relationships between the NDE parameters, material microstructure, and mechanical properties. The database will also contain the uncertainties of the measurements. The third goal of the project is the application of the NOMAD Tool to actual cladded RPV materials, simulating a real-life surveillance scenario. Thus, the NOMAD project strives to upgrade the already existing surveillance programs of Generation II and III reactors.

Because nondestructive tests do not directly measure material embrittlement, the NDE measurements need to be calibrated by comparing them to standardized destructively measured reference values. Multiple non-irradiated and irradiated samples have been analyzed using the six NDE methods, and reference values that describe the embrittlement have been obtained by measuring the corresponding DBTT and USE destructively with the Charpy impact test. These NDE and destructively measured parameters have been combined into a single database, referred to as the NOMAD Database, and the relationships between individual NDE parameters and embrittlement have been studied. The next task is to develop a tool that combines the NDE parameters. The final NOMAD Tool will combine different methods so that they optimally complement each other and provide better results than individual methods would.

3.2 Samples

The samples used in the NOMAD project are ISO-standard Charpy V-notch samples made out of four steel alloys, which are used in Eastern and Western RPVs as base and weld materials [27]. The materials are 18MND5-W (weld), 22NiMoCr37 (base), A508-B (base) and HSST-03 (base). The chemical compositions of the materials are listed in Table 3.1. The samples have been irradiated at the Belgian Nuclear Research Centre in the CALLISTO (Capability for Light-water Irradiation in Steady-state and Transient Operation) loop of the materials testing Belgian Reactor 2 reactor. The CALLISTO loop emulates PWR operating conditions [28].

Table 3.1: The chemical compositions (wt%) of the studied Charpy samples. The chemical compositions have been determined by optical emission spectroscopy. Table contents taken from Ref. [27].

Material	**C**	**Si**	**Mn**	**P**	**S**	**Cr**	**Mo**	**Ni**	**Cu**
18MND5-W	0.09	0.23	1.21	0.018	0.009	0.12	0.49	0.96	0.13
22NiMoCr37	0.20	0.25	0.87	0.009	0.007	0.39	0.49	0.85	0.06
A508-B	0.20	0.31	1.40	0.010	0.008	0.10	0.45	0.74	0.06
HSST-03	0.25	0.27	1.42	0.013	0.011	0.13	0.48	0.62	0.12

Samples made out of the same material were divided into groups. One of the groups was not irradiated and was measured as-received, while the remaining two or three groups were irradiated under different controlled CHIVAS (Callisto Hot Irradiation of Vessel Alloy Steels) irradiation programs [27]. The irradiation programs are defined by the irradiation temperature T_Irrad and neutron fluence F_Irrad. The three irradiation temperatures were 150 °C, 260 °C and 305 °C, and the neutron fluence ranged between 2.66–9.36 · 10^{19} neutrons/cm^2. The number of samples in each group and the irradiation conditions are summarized in Table 3.2. The irradiation made the samples radioactive, so post-irradiation measurements had to be conducted in hot cell laboratories. The NOMAD Database used in this thesis contains 89 samples and covers 12 different values of DBTT and 13 different values of USE. More samples will be measured during 2020–2021, meaning that the NOMAD Database will be extended in the future.

Table 3.2: The number of samples in each group and the irradiation conditions, irradiation temperature T_Irrad and neutron fluence F_Irrad, of each group. Some of the samples were not irradiated and were measured as-received. The number of samples in the NOMAD Database used in this thesis is 89.

Material	**# of samples**	**T_Irrad (°C)**	**F_Irrad (n/cm^2)**
18MND5-W	5	–	–
	9	150	$3.48 \cdot 10^{19}$
	3	260	$4.84 \cdot 10^{19}$
	6	260	$9.36 \cdot 10^{19}$
22NiMoCr37	3	–	–
	9	260	$3.37 \cdot 10^{19}$
	9	260	$5.73 \cdot 10^{19}$
A508-B	5	–	–
	9	150	$4.14 \cdot 10^{19}$
	8	305	$4.19 \cdot 10^{19}$
HSST-03	5	–	–
	9	150	$2.66 \cdot 10^{19}$
	9	305	$3.06 \cdot 10^{19}$

3.3 Methods

The basis of the NDE methods is that both the mechanical properties and the NDE parameters depend on the material microstructure. The NOMAD methods consist of electrical, micromagnetic and ultrasonic methods, and in total 28 NDE parame-

ters have been measured. Given that the goal is to find the optimal combination of the NDE parameters, the final tool will not necessarily include all of these NDE parameters. The NDE measurements have been performed by the following NOMAD partners: Coventry University (CU), Fraunhofer Institute for Nondestructive Testing (FhG-IZFP), Hungarian Academy of Sciences Centre for Energy Research (MTA-EK), VTT Technical Research Centre of Finland Ltd (VTT) and Paul Scherrer Institute (PSI).

The first three methods that are described in the following sections, the Micromagnetic Inductive Response and Barkhausen Emission (MIRBE) method, Micromagnetic Multiparameter Microstructure and Stress Analysis (3MA), and Magnetic adaptive testing (MAT), are micromagnetic methods that rely on the magnetic properties of ferromagnetic materials. By adjusting the parameters governing the magnetization process, the analyzed sample volume can be controlled. The next two methods described, the Direct Current-Reversal Potential Drop (DCRPD) and the Thermoelectric power measuring method (TEPMM), are electrical methods that require that the sample material is electrically conductive. Even though these methods cannot be applied to all materials, they are suitable for evaluating common RPV steel alloys. The last method measures the ultrasonic velocity in the sample. Ultrasonic testing does not set as strict requirements for the sample materials as the other methods, but it requires that the sample material is a solid.

Each NDE method is described in two parts. Firstly, the physical basis of each method is introduced in order to highlight the dependencies between the NDE parameters, material microstructure, and embrittlement. Secondly, the experiment and the parameters acquired via each method characteristic to the NOMAD project are outlined. See Appendices A and B for tables that list the parameters in the NOMAD Database. The database also contains the standard deviations of the DBTT, USE and NDE measurements. The standard deviations have been determined by repeated measurements.

3.3.1 The Micromagnetic Inductive Response and Barkhausen Emission method

Barkhausen effect

MIRBE is a nondestructive technique based on the Barkhausen effect [29]. According to domain theory, ferromagnetic materials consist of a multitude of domains that act like individual magnets [30]. These domains are separated by regions known as the

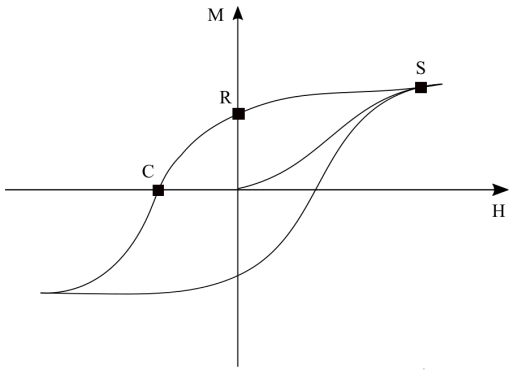

Figure 3.1: The major magnetic hysteresis loop plots the magnetization M as a function of the magnetic field strength H. The initial magnetization curve starts from the origin. Saturation point (S) is achieved when the Bloch walls cannot move any more.

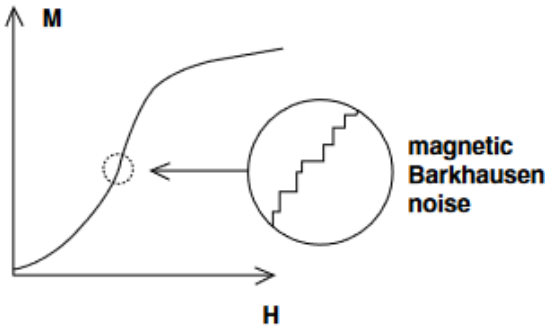

Figure 3.2: A closer look at the initial magnetization curve. The magnetization M contains discontinuous noise as a function of the external magnetic field strength H. This noise is known as magnetic Barkhousen noise. Figure taken from Ref. [29].

Bloch walls. When a ferromagnetic sample is demagnetized, the magnetizations of the individual domains point in random directions so that the net magnetization of the sample is negligible.

When an external magnetic field is applied, the sample becomes magnetized as the individual domains strive to become oriented in parallel to the external field. The domains that already had a close orientation with the external field grow in size at the cost of other, less-oriented, domains. The major magnetic hysteresis loop, which is visualized in Figure 3.1, plots the magnetization of the sample M as a function of the external magnetic field strength H. As the initial magnetization curve, which starts from the origin, reaches the saturation point (S), the Bloch walls cannot move any more and the magnetization will not increase. As the external field strength decreases back to zero, the magnetization behaves differently to the initial curve, and the beginning of the major hysteresis loop is produced. Hysteresis is a word for "lag behind", and in magnetics, it refers to the phenomenon where magnetization lags behind the applied field strength.

The magnetization curve is not continuous as can be seen from Figure 3.2, where the curve is amplified to reveal its discontinuity. The Bloch walls get stuck to or snagged on pinning sites in the lattice structure of the material. These pinning sites include dislocations, precipitates, and other lattice defects. If the external magnetic field is strong enough, the Bloch walls irreversibly jump over the pinning sites and create discontinuities in the magnetization curve. During the irreversible jumps, the Bloch walls lose energy. This causes the hysteresis of the magnetization curve. The noise in the magnetization curve is known as Barkhousen noise. Barkhausen noise gives

information about the interactions between the Bloch walls and the microstructure of the material. The defects created by neutron radiation act as pinning sites, so Barkhousen noise can be used to characterize neutron-induced embrittlement.

As a NOMAD parameter

The MIRBE measurements were performed by CU [31]. During the experiment, an external magnetic field is applied to a sample using an electromagnetic yoke, which consists of a U-shaped core made out of soft ferromagnetic material that has a magnetizing coil wound around its base. The magnetization curve is measured with a pick-up coil. A fixture that holds the sample, yoke, and the pick-up coil in place was used to make repeated measurements easier to perform.

The signal from the magnetic Barkhousen noise consists of several voltage pulses. The signal has a stochastic nature and a wide power range and usually needs to be high and low pass filtered [32]. The magnetization and analyzing frequencies determine the penetration depth. High frequencies correspond to near-surface emissions and low frequencies to deeper emissions. A single value, known as the magnetoelastic parameter MBN_RMS, is calculated as the root mean square of individual Barkhousen noise signal bursts as

$$\text{MBN\_RMS} = \sqrt{\frac{\sum_{i=1}^{n} y_i^2}{n}}, \tag{3.1}$$

where n is the number of Barkhausen noise signals in a certain analyzing frequency range and y_i is the amplitude of a single signal pulse. The bandwidth of the analyzing frequency range was set to 70–200 kHz in the NOMAD experiment [31].

The magnetizing frequency and voltage are adjustable experiment parameters. Whereas the magnetizing frequency affects the penetration depth, the voltage determines the magnitude of the external magnetic field. The optimal magnetizing frequency and voltage were found by utilizing a sweeping method, where a range of values is swept through. The optimal magnetizing voltage was 2 V, and the magnetizing frequency was 120 Hz. The magnetoelastic parameter is abbreviated as MBN_RMS in the NOMAD Database [33].

3.3.2 Micromagnetic Multiparameter Microstructure and Stress Analysis

3MA, which was performed by FhG-IZFP, measures several NDE parameters by combining three methods [34, 35]. The sample is magnetized periodically, and the eddy current impedance, harmonics in the magnetizing current, and incremental permeability are measured to obtain a so-called “magnetic fingerprint” of the sample.

Eddy currents

According to Faraday's law of induction, closed current loops known as eddy currents are induced within a conducting material when it is subjected to a changing magnetic field. The primary magnetic field is induced by a transmitter coil, which is fed a sinusoidal high frequency and low amplitude current. The eddy currents are perpendicular to the plane of the primary magnetic field and create a secondary magnetic field, which opposes the primary magnetic field. The intensity of the eddy currents decreases as the penetration depth into the sample increases. This is known as the skin effect. A higher magnetizing frequency produces signals from near the surface, and lowering the magnetizing frequency increases the penetration depth.

The eddy currents depend on the magnetic permeability and electrical conductivity of the material, which are affected by lattice defects. This means that eddy currents can be used to measure the neutron-induced embrittlement of a material. The eddy currents are measured as the change in the impedance Z of a receiver coil. Impedance is a complex number with the real part defined by resistance and the imaginary part by reactance. An eddy current impedance loop is plotted on an impedance plane, as illustrated in Figure 3.4, and several parameters are obtained from its droplet-like shape.

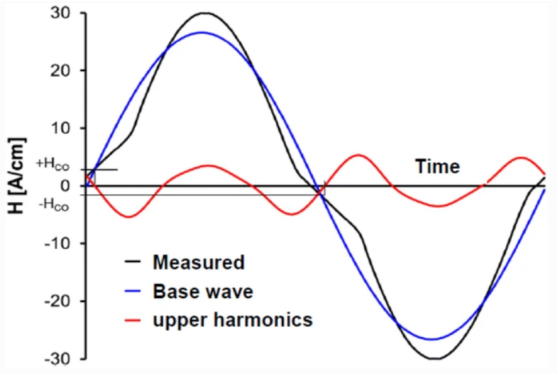

Figure 3.3: An illustration of the harmonics in the tangential magnetic field strength H. The measured signal is shown in black. The base wave (blue) and the third upper harmonic component (red) have been separated from the signal. Similar harmonics occur in the magnetizing current. The harmonics are caused by changes in coil impedance, which in turn are caused by the magnetic hysteresis of the sample. Figure taken from Ref. [36].

Harmonic analysis

During harmonic analysis, the major magnetic hysteresis loop is traversed once per magnetization period. The sample is magnetized with a magnetic yoke that has a magnetizing coil wound around it. The coil is supplied with a sinusoidal low frequency

voltage, so the magnetic flux density also changes sinusoidally. However, the magnetizing current and magnetic field strength tangential to the sample surface are not sinusoidal but contain harmonics. The harmonics occur because the non-linear magnetic hysteresis behavior of the sample disturbs the current in the magnetizing coil by changing its impedance.

According to Fourier's theorem, a periodic signal can be expressed as the sum of sine terms. Fast Fourier Transform is used to process the signal and measure the harmonic components, as illustrated in Figure 3.3. The magnetic hysteresis of a ferromagnetic material is affected by lattice defects, so the harmonic analysis gives information about the embrittlement of the material.

Incremental permeability

In order to measure the incremental permeability of the sample independently from the electrical conductivity, two simultaneously exciting sources are required. A high amplitude and low frequency source, which consists of a magnetic yoke and a coil, generates major magnetic hysteresis loops in the sample similarly to the harmonic analysis. At the same time, a transmitter coil with low amplitude and high frequency voltage generates minor hysteresis loops similarly to eddy current impedance analysis. The minor loops are superimposed on the major loop. This means that an eddy current analysis is performed and the impedance is measured at several points of the hysteresis cycle.

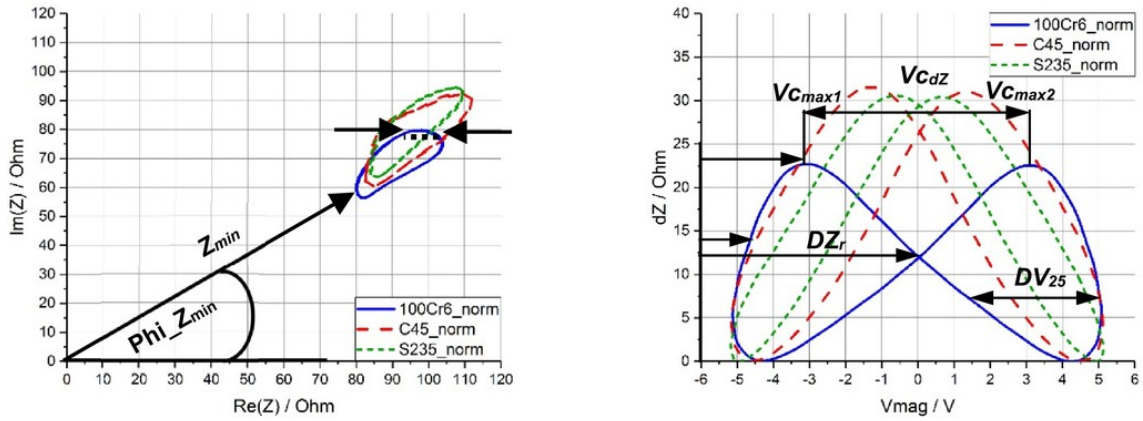

Figure 3.4: The eddy current impedance loops (on the left) and the incremental permeability curves (on the right) of three materials. The eddy current impedance loops are plotted on a complex impedance plane. The incremental permeability curves are obtained by plotting the change of the eddy current coil impedance dZ as a function of the magnetizing voltage Vmag. The shapes of the plots are used to extract several NDE parameters, some of which have been visualized in this figure. Figure taken from Ref. [37].

The slope of the magnetization curve describes the change in permeability that happens during magnetization. Thus, the slopes of the superimposed inner minor hysteresis loops are proportional to the incremental permeability. An incremental permeability curve is obtained by plotting the amplitude change of the eddy current impedance loop, which corresponds to the imaginary component of the impedance loop, as a function of magnetizing voltage. The incremental permeability curve represents the change in impedance at different parts of the hysteresis cycle. The incremental permeability curves of three samples can be seen in Figure 3.4, where some of the several parameters that stem from the shape of the incremental permeability loop are also illustrated.

As NOMAD parameters

The setup of the 3MA experiment consists of a 3MA device and a probe. The probe head and its components are illustrated in Figure 3.5. The probe combines the components needed in the three individual measurements and it measures the 3MA parameters quasi-simultaneously. The device generates a sinusoidal voltage that drives the magnetizing coil in the probe. The coil is wound around a magnetic yoke, and this electromagnetic yoke excites a magnetic field in the sample. A second magnetic field can be excited in the sample with the transmitter coil. The magnetic response of the sample is picked up by a receiver coil, and a Hall sensor measures the magnetic field.

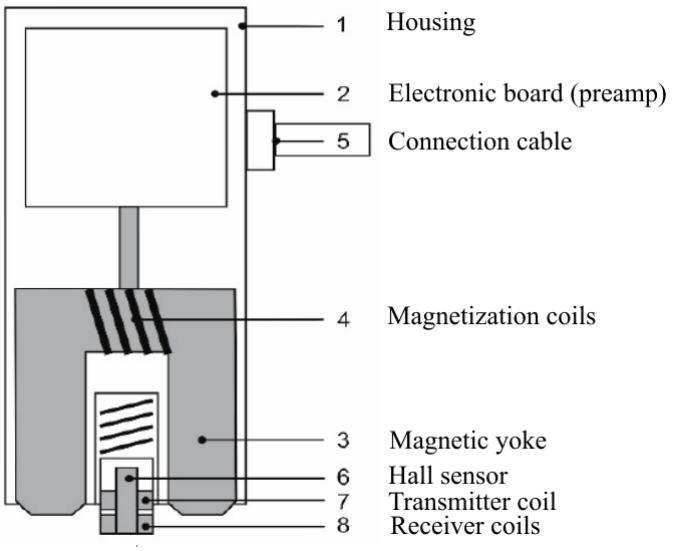

Figure 3.5: The probe head of a Micromagnetic Multiparametric Microstructure and Stress Analyzer. The separate components of the probe are shown in the figure. During a measurement, the probe is attached to the sample so that the magnetic yoke and the sample form a closed magnetic circuit. Figure adapted from Ref. [34].

All 3MA parameters were measured from the side of the Charpy samples that is opposite the V-notch [31]. Two experiment parameters, the magnetizing frequency and amplitude of the magnetizing voltage, can be changed by the user. In the NOMAD experiment, the parameters were selected using the sweep method. The experiment parameters were selected so that they provided an optimal distinction between samples with different irradiation levels. The magnetizing frequency used was 50 Hz, and the voltage was 2 V. In order to characterize the mechanical properties, the 3MA-device needs to be calibrated. This is done using a well-known calibration data set, which contains both the measurements and reference target values, and a calibration function that describes the relationship between the target parameter and the 3MA parameters. The calibration function can be obtained via regression analysis or a pattern recognition software. A regression analysis was used in the NOMAD project.

In total, 20 NDE parameters were obtained from the 3MA experiments [33]. The eddy current impedance loop and some of the several parameters obtained from its shape are shown in Figure 3.4. The parameters and their abbreviations in the NOMAD Database are the minimum, maximum and mean values of the impedance loop (abbreviated as Zmin, Zmax and Zmean), the corresponding phase angles (Phizmin, Phizmax and Phizmean), and the width of the impedance loop at 3 % and 10 % of maximum (W3Z and W10Z). The parameters derived from the harmonic analysis are the amplitude of magnetizing voltage (Vmag), the amplitude of magnetizing current (Imag), the distortion factor (K), the amplitude of the third harmonic (A3), and the phase shift of the third harmonic relative to the fundamental wave (P3). The parameters derived from the shape of the incremental permeability curve illustrated in Figure 3.4 are the maximum and mean amplitude of the incremental permeability curve (DZmax and DZmean), the magnetizing voltage at the incremental permeability curve maximum (Ucdz), the value of the curve at the remanence (DZr), and the width of the incremental permeability curve at 75 %, 50 % and 25 % of the maximum amplitude (DU75dz, DU50dz and DU25dz).

3.3.3 Magnetic adaptive testing

Magnetic permeability

MAT is an NDE technique that systematically measures the minor magnetic hysteresis loops of a sample [38]. Minor magnetic hysteresis loops differ from the major magnetic hysteresis loop, which is depicted in Figure 3.1, by not reaching the saturation point. More data can be gathered by measuring a series of minor hysteresis loops instead of a single major hysteresis loop.

Magnetic permeability measures the resistance of a material against the magnetization induced by an external magnetic field, and permeability curves can be plotted based on the minor hysteresis loops. Lattice defects affect the magnetic behavior of the material and, subsequently, the permeability curves. Thus, analyzing the differential permeability μ of the material provides information about neutron-induced embrittlement. A closely linear correlation has been reported between the MAT parameters and DBTT [39].

As NOMAD parameters

The MAT measurements were conducted by MTA-EK. The samples were demagnetized before the experiments with a coil that was fed a decreasing current with a triangular waveform. During the experiment, an electromagnetic yoke was attached to the surface of the sample. The sample was magnetized by feeding the electromagnetic yoke a triangular waveform current with an increasing amplitude and a fixed slope in each triangle. This induces a sweeping and gradually growing magnetic field, H, into the sample. A pick-up coil collects a voltage signal U of the form

$$U(\mathrm{d}H/\mathrm{d}t, H, A_j) = K \cdot \mu(\mathrm{d}H/\mathrm{d}t, H, A_j) \cdot \mathrm{d}H/\mathrm{d}t, \tag{3.2}$$

where A_j is the amplitude of the jth loop, t is time and K is a constant [38]. As can be seen from Equation (3.2), as long as the slope of the applied magnetic field $|\mathrm{d}H/\mathrm{d}t|$ stays constant, the voltage signal is proportional to the differential permeability of the sample.

The MAT measurements collect a family of permeability loops, like the ones in Figure 3.6, for each sample. Instead of keeping the data in this continuous form, a large set of data points are obtained by evaluating the loops in discrete points. This yields a permeability matrix $\mu(H, A_j)$. Usually, the magnetic field strength in the sample and amplitude are used as the coordinates of the matrix. However, if the magnetic circuit is not perfectly closed and the magnetic field is scattered as a result, the exact magnetic field in the sample is not known. In cases like this, the corresponding magnetizing current is used instead of the magnetic field. This was the case in the NOMAD experiment [31].

The most sensitive matrix element was chosen for each sample with the help of sensitivity matrices. This element is known as the optimally chosen descriptor, and the process is described with more detail in Ref. [38]. In the NOMAD Database, MAT_1 values are optimally chosen descriptors that have been normalized by dividing each value by a corresponding value measured from a non-irradiated reference sample [33]. The optimally chosen descriptors that have not been normalized are abbreviated as MAT_2.

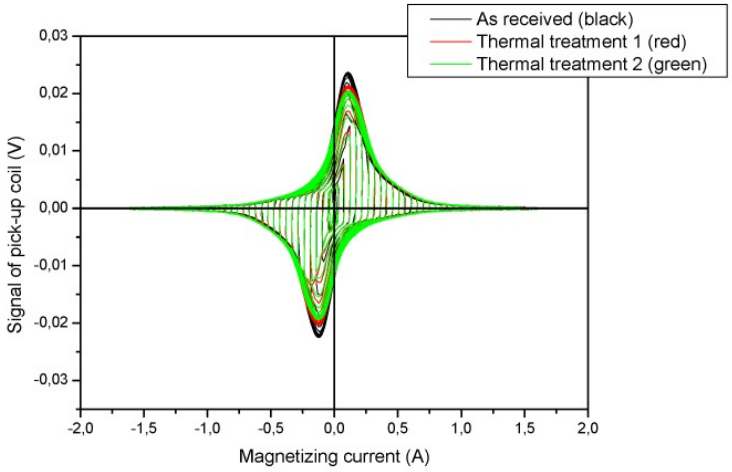

Figure 3.6: The permeability loops of three samples. One of the samples was measured as-received, and two of the samples were thermally treated before measurements. The thermal treatments caused visible changes to the permeability loops. Figure taken from Ref. [39].

3.3.4 Direct Current-Reversal Potential Drop

Electrical resistivity

DCRPD measures the electrical resistivity of the sample [40]. The resistivity of metals is explained through their microstructure by the free electron model [30]. The model assumes that the conduction electrons in metals act like free particles and ignores the repulsion between the electrons. According to the free electron model, the dependency between resistivity ρ and the mean free path of electrons l_{avg} is given by

$$\rho = \frac{m_e v_F}{n_e e^2 l_{\text{avg}}}, \tag{3.3}$$

where m_e is the mass on an electron, v_F is the Fermi velocity, e is the elementary charge, and n_e is the electron density.

The mean free path is the average distance a conduction electron travels between interactions. For instance, the dependency between temperature and resistivity in metals can be explained with the free electron model: an increase in temperature increases the thermal motion of atoms so that the lattice structure of the metal becomes more irregular and the mean free path decreases. This causes the resistivity to increase. Equivalently, lattice defects reduce the mean free paths of electrons. This indicates that resistivity can be used to estimate neutron-induced embrittlement.

As NOMAD parameters

The resistance of the samples was measured by VTT with a four-point probe [31]. A well-known constant direct current was fed through a sample using two spring-loaded

needle probes. A second pair of similar needle probes was used to measure the potential difference. Resistance is obtained according to Ohm's law by dividing the potential difference by the current running through the sample. Because the samples are assumed to be homogeneous, the measured resistance R can be expressed as a function of the resistivity as

$$R = \rho \frac{l}{A}, \tag{3.4}$$

where A is the cross-sectional area of the sample and l the distance between the two measurement points. If the measurement geometry does not change, observed changes in resistance are caused by changes in resistivity.

The setup was calibrated using several pure metals. The resistance of each sample was measured in three different configurations and each measurement was repeated 10,000 times. The first configuration measured the resistance of the whole sample in the longitudinal direction, over the V-notch. These measurements are abbreviated as Resistance_1 in the NOMAD Database [33]. The second and third configurations measured the resistance from the two halves of the Charpy sample that are separated by the V-notch, also in the longitudinal direction. These results are abbreviated as Resistance_2 and Resistance_3.

3.3.5 Thermoelectric power measuring method

Thermoelectric power

TEPMM is based on the Seebeck effect [41]. In electrically conducting materials, a temperature gradient is accompanied by the build-up of electric potential. The Seebeck effect is explained by the coupling between thermal and electric currents in metals. Whereas in insulators heat is carried by phonons, in metals both heat and electric charge are carried by electrons [30]. If a thermal gradient is present, the electrons will diffuse from the hotter part to the colder part. A charge builds up in the colder part, and a voltage is induced between the two parts. This voltage, known as thermoelectric power, induces a small current over the temperature gradient. The thermoelectric power U between two points is determined by the mean Seebeck coefficient $\overline{S}$ and the difference in temperature T according to

$$U = \overline{S}(T_1 - T_0). \tag{3.5}$$

The Seebeck coefficient is affected by several factors other than neutron irradiation, such as heat treatments, but TEPMM has been shown to be a suitable technique to estimate neutron-induced embrittlement. According to measurements by PSI, there is a linear dependency between the Seebeck coefficient and neutron-induced embrittlement of RPV steel alloys [41].

As a NOMAD parameter

The Seebeck coefficient was measured from the samples in longitudinal and transversal directions [31]. Each sample was held between two copper blocks that were heated by flowing water, and the temperatures of the two water sources were 20 °C and 40 °C. The measuring time was 10 minutes to ensure that the temperature gradient was stabilized. The voltage was measured from the copper blocks, as they acted as electrodes. The temperature was measured with thermocouples, and the Seebeck coefficient was determined according to Equation (3.5) by dividing the voltage by the temperature difference. The average values of the last 50 temperature and voltage measurements were used to calculate the Seebeck coefficient. The Seebeck coefficient is abbreviated as SC in the NOMAD Database [33].

3.3.6 Ultrasonic method

Piezoelectric ultrasonic velocity

Ultrasonic waves are sound waves that have frequencies higher than 20 kHz and are thus inaudible to the human ear. Sound waves propagate through a medium as atoms in the lattice vibrate around their equilibrium positions. Ultrasonic testing commonly uses the longitudinal and transversal modes of sound wave propagation in solids. In the longitudinal mode, the atoms oscillate in the same direction as the waves propagate. In transversal mode, the atoms oscillate in directions that are transverse to the direction of wave propagation.

Piezoelectric materials are used to produce and receive ultrasonic waves [42, 43]. When a material that exhibits piezoelectricity is subjected to mechanical stress, it accumulates electric charge. This is the piezoelectric effect. If the material is contracted and expanded, a pulsed electric signal is produced. The effect is also reversible: if a piezoelectric material is subjected to an external electric field, it contracts or expands in response to the direction of the electric field. This is the inverse piezoelectric effect. If the applied electric field is alternating, the material will vibrate and create sound waves in the surrounding medium.

Sound velocity v for longitudinal waves in a homogeneous and isotropic solid is a function of the density ρ and the elastic properties, Young's modulus E and Poisson's ratio ν, of the material according to

$$v = \sqrt{\frac{E(1-\nu)}{\rho(1+\nu)(1-2\nu)}}. \tag{3.6}$$

If the effects of neutron-induced swelling can be ignored, the density of a solid does not change under irradiation, and the elastic properties E and ν must be responsible

for changes in sound velocity. On a microscale, the elastic properties are tied to the interactions between atoms, which in turn are affected by neutron irradiation. This means that the ultrasonic sound velocity can be used to assess the embrittlement caused by neutron irradiation.

As a NOMAD parameter

A piezoelectric transducer uses the indirect piezoelectric effect to transmit ultrasonic waves and the direct effect to receive the waves. The transducer transforms incoming ultrasonic waves into voltage signals. A coupling agent is needed in between the sample and the transducer to transmit the waves between them, as illustrated in Figure 3.7. The ultrasonic experiments for the NOMAD project were performed by FhG-IZFP, who used longitudinal wave transducers with a peak frequency of 15 MHz [31]. A fixture was designed to hold the ultrasonic probe and the sample, and the measurements were taken through the thickness of the sample d. The time-of-flight t is defined as the time in which the ultrasonic wave travels from the transducer to the receiver and is measured as the time difference between two successive back wall echoes. The time difference can be determined accurately by fitting a function to the voltage signal. The ultrasonic velocity c_{us} in the sample is defined as

$$c_{\mathrm{us}} = \frac{2d}{t}. \tag{3.7}$$

In addition to the accuracy of the measured time-of-flight and sample thickness, the coupling conditions between the transducer and the sample affect the accuracy of the measurements. An estimation of the coupling accuracy and the optimal measuring parameters were determined via repeated measurements. The ultrasonic velocity is abbreviated as c_us in the NOMAD Database [33].

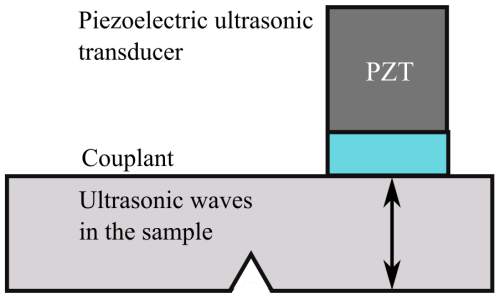

Figure 3.7: A piezoelectric ultrasonic transducer uses a piezoelectric material to excite ultrasonic waves. A commonly used piezoelectric material is lead zirconate titanate (PZT). A couplant transfers the waves from the transducer into the sample and vice versa. The waves are reflected from the back wall of the sample and received in the transducer, which turns them into voltage signals.

4. Machine learning

The goal of the NOMAD project is to develop a software-based tool that uses the NDE parameters to estimate the degradation of RPV steel alloys. Supervised machine learning algorithms, being the current state-of-the-art method for regression analysis, are a natural choice as the basis of this tool. This chapter covers the basics of machine learning, concentrating on supervised machine learning and its use in solving regression tasks. An overview of six different machine learning regression algorithms is provided.

4.1 Basics

The frameworks and algorithms of machine learning have been developed since the 1950s. As computers have advanced over the past few decades, machine learning has become a state-of-the-art tool for solving a broad range of tasks. Machine learning is a subset of artificial intelligence, and the goal of a machine learning algorithm is to learn how to execute a task without an explicit set of instructions, or an algorithm, to follow. Thus, machine learning algorithms can tackle tasks that require an algorithm that would be difficult to develop manually by a human.

The three main categories of machine learning are supervised learning, unsupervised learning, and reinforcement learning. A supervised machine learning algorithm builds a model that predicts outputs to given inputs. The traditional approach to building such prediction models would be to write an explicit program based on a particular function that calculates an output from a given input. The machine learning approach is to take a data set that contains both the inputs and the corresponding correct outputs, write a general framework, and then let a computer to deduce the explicit model based on the data.

In unsupervised learning, the algorithm is given data without the correct outputs, and the algorithm tries to find hidden patterns from the data independently. Unsupervised learning is used in clustering tasks, for example, where a set of input data needs to be grouped. A reinforcement learning algorithm learns through a sequence of decisions. The algorithm interacts with an environment, receives a reward when it makes a good decision, and learns by striving to maximize the rewards. A real-life example of reinforced learning is an algorithm that can play computer games and has been able to beat human players.

4.2 Supervised machine learning

In its most basic form, the model built by a supervised machine learning algorithm predicts an output variable, known as a target, based on an input variable, which is known as a feature. Supervised learning is used in solving classification and regression problems. In classification, the output values are discrete. A classic example of a classification task is a model that classifies inputs as either cats or dogs. In regression, the output is a continuous quantity. A commonly used example of a regression task is a model that predicts house prices based on the features of the houses, such as their size and age. Given the aim of this thesis, another case of regression would be a model that predicts the DBTT and USE of RPV steel alloys based on their nondestructively measured features, such as resistance and ultrasonic velocity.

Usually, the input data consists of several features, in which case the process is called a multivariate classification or regression. When an output vector, instead of a scalar, is predicted for each sample based on an input feature vector, the process is known as a multi-label multivariate classification or multi-target multivariate regression. The desired result is to develop a model that generalizes well, meaning that it can predict accurate outputs for unseen input data. Unseen data means data that was not used to train the model. A thorough coverage of supervised machine learning and individual algorithms can be found in Refs. [44, 45], which have been used as the basis of the following sections of this chapter unless otherwise indicated.

4.2.1 Workflow

The process of building a supervised machine learning model follows a certain workflow. In order to train a model and assess its generalization ability, a data set that contains both the inputs and the corresponding correct target outputs is split into training and test sets. Commonly, the size of the training set is 80 % of the whole data set and the remaining 20 % constitutes the test set. The model is trained using the training set, which contains the feature vector and the true target of each training sample. Training means the process where a machine learning algorithm is supplied with the training set, which it then uses to build a model that describes the relationship between the features and the target.

Fitting a regression model to a set of training data means finding a balance between underfitting and overfitting. Figure 4.1 visualizes underfitting and overfitting models. If the model overfits the training data, the model is too complex. Even though an overfitting model describes the training data extremely well, the model will generalize poorly when given unseen input data because it has learned from overly fine details of the training data. Bias measures the difference between the predictions and

the true targets, and variance measures the sensitivity of the model to variations in the training data. An overfitting model will have a low bias and high variance. If the model underfits, the model is too simple and will not be able to adapt to unseen data. Underfitting models have a high bias and low variance.

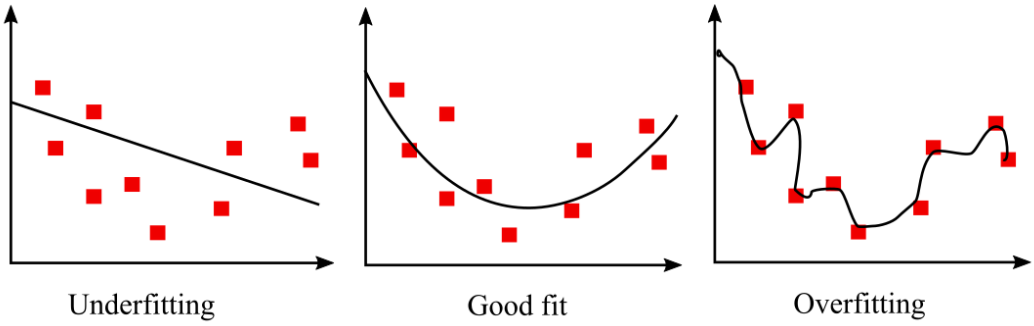

Figure 4.1: Visualizations of underfitting and overfitting regression models. The underfitting model does not describe the training data with sufficient accuracy, while the overfitting model fits the noise in the training data. An example of a good fit is in the middle.

The generalization ability of the trained model is tested by feeding the test set to the model without its targets. The model makes predictions based on the input, and metric functions are used to measure the test accuracy of the model by comparing the predicted targets with the targets of the test set. For classification tasks, a common metric is accuracy, which returns the number of correctly classified samples. For regression tasks, common metrics are the coefficient of determination R^2, mean absolute error (MAE) and mean squared error (MSE).

Because the training set practically shapes the entire model, it plays a crucial role in determining the accuracy of the model. Machine learning algorithms do not distinguish bad data from good data, and biased or low quality training data will lead to a biased or low quality model. However, one cannot select the best data points for training, as the training and test sets should represent the whole data set equally well. Otherwise, the test results will not describe the true generalization ability of the model. Another significant factor that contributes to model performance alongside data quality is data quantity. A smaller number of total samples means fewer training samples, which will produce a model with a worse generalization ability. On the other hand, a larger data set does not necessarily lead to better model performance, because an excessive number of features increases the complexity and computation time of the model.

Handling missing values

Raw data can rarely be directly fed to a machine learning algorithm, so several data preprocessing steps need to be taken before the machine learning algorithm can be applied to the data. Although the preprocessing steps depend on the initial state of the data, the basic formula is the same.

The first step in data preprocessing is to detect and handle the possible missing values, known as NaN (Not a Number) values, in the data set. NaN values can be handled by either removing the samples or features containing NaN values altogether or by replacing the NaN values with some predetermined value, often the mean value of the feature in question. Alternatively, algorithms that automatically fill missing values can be used, such as the k-nearest neighbors algorithm (see section 4.3.4).

Encoding categorical variables

The second step in data preprocessing is to encode categorical variables, which are variables that are in string format. An example of a categorical variable is a variable categorizing samples as either "cats" or "dogs". In order for a machine learning algorithm to be able to process categorical variables, the variables need to be encoded into a numerical format.

A simple way to encode categorical variables is to give each category a corresponding numerical value, for example by replacing the categorical variable with a variable where the samples that are classified as "cats" are marked with zero and "dogs" with one. This works with binary classification tasks as long as each sample belongs to either class. However, it makes the model non-recyclable because new categories cannot easily be added to the model. In addition, the categories now have irrelevant orders because the numerical values imply that dogs are more important than cats. One-hot encoding is used to prevent irrelevant orders in categorical variables. This "dummy" encoding scheme creates a binary variable for each of the categories. The variables of each sample then contain a binary array where the number one represents the category to which the sample belongs.

Scaling

The third step is to scale the features, as most machine learning algorithms benefit from normalized or standardized input data. If the features have multiple different scales or are not normally distributed, training can take excessive amounts of time or the model cannot be fully trained at all. A value x is normalized and standardized according to

$$x_{\mathrm{n}} = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{4.1}$$

$$x_{\mathrm{s}} = \frac{x - \mu}{\sigma}, \tag{4.2}$$

where x_{n} is the normalized value and x_{s} the standardized value. Normalization scales a feature to a range between zero and one as expressed in Equation (4.1) using the maximum and minimum values of the feature $x_{\max}$ and $x_{\min}$. It does not touch the distribution of the data, so the distributions of different features might still differ substantially after normalization. Normalization is sensitive to outliers because exceptionally large or small values distort the scaling of other values. Standardization scales the features according to Equation (4.2) using their mean μ and standard deviation σ so that the features have a mean of zero and unit variance. As standardization has no bounds, it is less sensitive to outliers. It is essential to split the data into training and test sets before scaling and scale the features based on the statistical parameters of the training set. Otherwise, information from the test set leaks to the model. This leakage can cause overfitting to go unnoticed.

Feature selection

In order to optimize the performance of a model, the least relevant features can be removed. If too many features with a weak correlation with the target variable are included in the analysis, they reduce the computational efficiency, make the model more complex, and make the model harder to interpret. Increased model complexity increases the probability of overfitting. A large number of features prompts a problem known as the curse of dimensionality: when the number of dimensions increases, the volume of the feature space increases, and the data becomes more sparse. As the number of features increases, the number of samples required to maintain accuracy grows exponentially.

Several methods have been developed to perform feature selection, and different methods might result in different subsets of features. In this case, user knowledge of the data can be used to choose the best subset of features, or the user might need to turn to brute force to assess all alternative options. A feature selection algorithm that utilizes brute force tests all possible subsets of the features and returns the combination that leads to the best results. A downside of any brute force method is their copious use of computing power.

Cross-validation

After preprocessing, the training set can be supplied to the algorithm. The test set is used to assess the fully trained model, but say, one wants to study if removing different features improves model performance. The different subsets of features cannot be

evaluated using the test set, because then the model would end up fitting to the test set, and the test set would no longer measure the generalization ability of the model. This problem is solved with a third subset, known as a validation set, which is extracted from the training set and used to assess the model during the training process by feeding it to the model without its targets. Commonly, the size of the validation set is 20 % of the training set. This process is visualized in Figure 4.2.

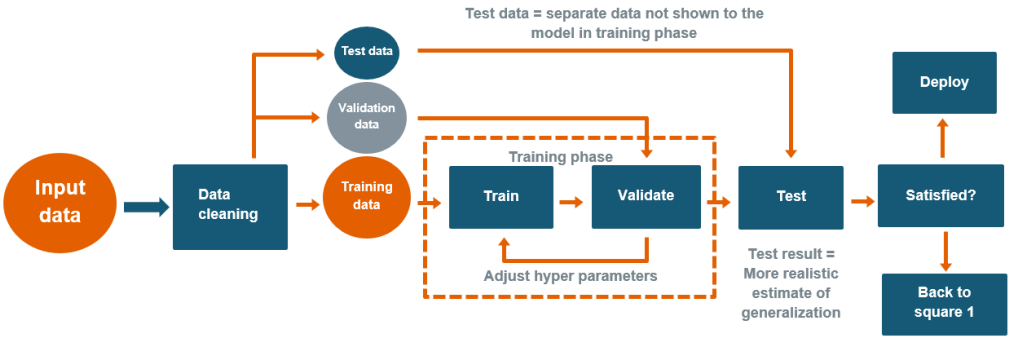

Figure 4.2: A workflow of developing a supervised machine learning model. The data is split into three parts: training, validation, and test sets. The model is trained using the training set and fine-tuned during the training phase with the help of the validation set. The generalization ability of the trained model is evaluated with the test set.

A less biased alternative for the split into training, validation, and test sets is k-fold cross-validation [46]. In this method the training set is split into k groups known as folds, and one by one, each fold is used as a validation set while the other $k-1$ folds are used to train the model. This is repeated until k scores have been obtained and can be averaged into a single cross-validation score. Commonly used numbers of folds are 5 and 10. This method is especially useful when the size of the data set is small, as it does not necessarily reduce the amount of training data as much as the extraction of a traditional validation set. In an extreme case, the number of folds can be set to equal the number of training samples. The size of the training set is reduced by only one in each fold. This is known as leave-one-out cross-validation.

Hyperparameter optimization

During the training process, the model learns its intrinsic model parameters based on the training data. Models also contain parameters that are not learned during training and are controlled by the user, known as hyperparameters. An example of a hyperparameter is the degree of the polynomial in polynomial models. The model is fine-tuned to its best performance by optimizing its hyperparameters. The user can tune the hyperparameters by hand until a satisfactory cross-validation score is achieved, but this is often impractical, as most machine learning algorithms have multiple hyperparameters.

An algorithm that combines a grid search and cross-validation can be used to find the optimal hyperparameters. An exhaustive grid search goes through all the possible hyperparameter combinations from a user-defined parameter grid and builds a model based on each combination. All of the models are evaluated with cross-validation, and the model with the best score is chosen. In order to reduce the required computing power, especially if the hyperparameter ranges are large, a randomized search can be performed instead of an exhaustive search.

Algorithm selection

Similarly to how different sets of hyperparameters of the same algorithm produce different models, different machine learning algorithms build different models based on the same training set. There is no way to see which algorithm will yield the best model in advance, so the process of selecting the optimal algorithm contains some trial and error. The performance of an algorithm depends on the problem and data at hand. Brute force can be utilized by building several models using different algorithms, comparing their performance using cross-validation, and choosing the one with the best accuracy. However, in addition to accuracy, the simplicity and efficiency of the algorithm might be attributes worth taking into account.

4.3 Regression algorithms

Several different kinds of supervised machine learning algorithms have been developed in order to efficiently solve different kinds of regression tasks. The following sections introduce six regression algorithms that are quite distinct in nature.

4.3.1 Linear models

Linear regression fits a linear equation to the training data. If the training data contains N samples described by d independent features, the predicted target for the ith training sample $\hat{y}_i$ is written as

$$\hat{y}_i = w_0 + w_1 x_{i1} + ... + w_d x_{id}, \tag{4.3}$$

where w_0 is a constant bias term and x_{ij} is the jth feature for the ith sample. Each feature is multiplied by a weight w_j. If the weight vector is written as $\boldsymbol{w} = (w_0, w_1, ..., w_d)^\mathsf{T}$ and each feature vector is written as $\boldsymbol{x}_i = (1, x_{i1}, ..., x_{id})$, the model can be expressed in matrix notation as

$$\hat{\boldsymbol{y}} = \boldsymbol{X}\boldsymbol{w}, \tag{4.4}$$

where $\hat{\boldsymbol{y}}$ is a $N \times 1$ vector that contains the predictions for all training samples and $\boldsymbol{X}$ is a $N \times (d+1)$ feature matrix with each row a feature vector. In the case of multi-target regression with k target variables, $\hat{\boldsymbol{y}}$ would be a matrix with dimensions of $N \times k$ and $\boldsymbol{w}$ would have dimensions of $(d+1) \times k$. The number of systems of linear equations would increase from one to k. This means that linear regression algorithms support multi-target regression intrinsically.

The model learns the weights from the training data by minimizing a cost function. A loss function calculates the difference between the predicted target $\hat{y}_i$ and the true target y_i of a single training sample, and a common loss function is the squared loss. The cost function calculates the sum of losses over all training samples. The least squares method minimizes the residual sum of squares (RSS):

$$\mathrm{RSS}(\boldsymbol{w}) = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2, \tag{4.5}$$

which can be written in matrix notation as

$$\mathrm{RSS}(\boldsymbol{w}) = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w})^{\mathsf{T}} (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}), \tag{4.6}$$

where $\boldsymbol{y}$ is a $N \times 1$ vector that contains the targets of the training set. Differentiating Equation (4.6) with respect to $\boldsymbol{w}$ and setting the first derivative to zero leads to

$$\boldsymbol{X}^{\mathsf{T}} (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}) = 0. \tag{4.7}$$

Now the optimal weights that can be used to make predictions are given by

$$\boldsymbol{w} = (\boldsymbol{X}^{\mathsf{T}} \boldsymbol{X})^{-1} \boldsymbol{X}^{\mathsf{T}} \boldsymbol{y}. \tag{4.8}$$

A regularization term can be added to the cost function in Equation (4.5) to set constraints to the weights. Ridge regression [47], also known as L2 regularization, adds a regularization term to the cost function that forces the model to keep the squared sum of the weights as small as possible. Hyperparameter λ controls the amount of regularization. The cost function is expressed as

$$\text{Ridge: } J(\boldsymbol{w}) = \mathrm{RSS}(\boldsymbol{w}) + \lambda \sum_{j=1}^{d} w_j^2. \tag{4.9}$$

Lasso regression [48], also known as L1 regularization, strives to flatten the function by seeking small weights, which is done by a regularization term that minimizes the absolute values of the weights. Lasso regularization can minimize the weights of some features to zero and thus performs intrinsic feature selection. The cost function is

$$\text{Lasso: } J(\boldsymbol{w}) = \mathrm{RSS}(\boldsymbol{w}) + \lambda \sum_{j=1}^{d} |w_j|. \tag{4.10}$$

Elastic net combines lasso and ridge regularizations [49]:

$$\text{Elastic net: } J(\boldsymbol{w}) = \text{RSS}(\boldsymbol{w}) + \lambda_1 \sum_{j=1}^{d} w_j^2 + \lambda_2 \sum_{j=1}^{d} |w_j|. \tag{4.11}$$

Despite their name, linear regression models are not constrained to predicting only linear relationships between input and output variables. What the name "linear" refers to, is that the model is linear in the weights, meaning that all terms are either a constant or a weight multiplied by an independent feature. The features can be transformed with a non-linear mapping function in order to implement a non-linear fit. These kinds of models are called kernelized regression models. For example, a polynomial kernel function maps the feature vectors onto a polynomial space. The model is still linear in the weights, and the degree of the polynomial is a new hyperparameter.

4.3.2 Training models with gradient descent

Before the rest of the regression algorithms are introduced, it is worthwhile to shortly discuss how many machine learning algorithms are trained. Finding the optimal model parameters that minimize the cost function is essentially the same as training the model. The process of fine-tuning the hyperparameters is also a part of the training, but it is not an intrinsic optimization process. A common optimization algorithm used to optimize the model parameters is the gradient descent algorithm [50]. An example of these intrinsic model parameters are the weights of the linear model. In the previous section, the linear model was optimized via the least squares method, but it could have as well been optimized using the gradient descent algorithm.

The gradient descent algorithm finds the model parameters $\boldsymbol{w}$ that minimize the cost function $J(\boldsymbol{w})$ by calculating the gradient of the cost function stepwise. The step length is known as the learning rate η of the algorithm and is defined by the user. Initially, the parameters are set to zero or some random values. At each step, the algorithm updates the model parameters so that the next step will be taken in the opposite direction of the gradient vector. Eventually, the algorithm will settle into a local minimum of the cost function if the learning rate is a suitable step length. The updated model parameters $\boldsymbol{w}_{n+1}$ are computed from the model parameters at the previous step $\boldsymbol{w}_n$ as

$$\boldsymbol{w}_{n+1} = \boldsymbol{w}_n - \eta \nabla_{\boldsymbol{w}} J(\boldsymbol{w}_n). \tag{4.12}$$

There are three versions of the gradient descent. Standard, or batch, gradient descent uses the whole training set to compute the gradient of the cost function before it updates the model parameters. This makes it the slowest of the three. Stochastic gradient descent calculates the gradient and updates the parameters one training sample at a time and is thus faster but, in contrast, fluctuates more than batch gradient

descent. If multiple iterations over the training set are performed, the training data is shuffled in between iterations. Mini-batch gradient descent combines the two prior versions. It divides the training set into mini-batches that contain more than one sample, and the mini-batches are then used one by one to calculate the gradient and update the parameters. Again, if the training set is iterated multiple times, it is shuffled in between.

A challenge concerning these gradient descent algorithms is the requirement to set a constant learning rate. If the learning rate is too large, the algorithm might jump over the minimum. If the learning rate is too small, the training process becomes computationally expensive. This problem is solved with a learning rate that changes as the training process progresses, which can be implemented with a pre-defined learning rate schedule or an adaptive learning rate. Learning rate schedules contain hyperparameters that must be set by the user, and their performance depends on the task at hand. Therefore, algorithms that utilize an adaptive learning rate are preferred. Adaptive learning rates adjust the learning rate in response to the performance of the model. Several adaptive gradient descent algorithms have been developed, such as the adaptive moment estimation (*Adam*) [51].

4.3.3 Support vector machines

The intuition behind support vector machines [52] can be explained through classification. A simple classification task consists of a training data set that can be divided into two classes. The support vector machine finds the hyperplane that separates the two classes so that the margin between the hyperplane and the nearest data points is at its maximum. This means that only a subset of the data points, known as the support vectors, are used to build the model. If the margin is hard, all the data points must be outside the margin. If the margin is soft, some violations to the margin are allowed. The simplest way to separate two classes from each other is with a linear function, but in trickier cases, a linear function is not enough. The feature vectors can be mapped individually to another space where they are linearly separable: $\boldsymbol{x_i} \rightarrow \Phi(\boldsymbol{x_i})$, where $i = 1, 2, ..., N$ and N is the number of training samples. The mapping function Φ is known as a kernel.

When used for regression, the algorithm is called support vector regression (SVR) [53]. In contrast to classification, in regression, the objective is to fit as many data points as possible inside the margin. The most distant data points act as support vectors. An SVR algorithm with a hard margin demands that all points must be inside the margin of width ε, and an SVR with a soft margin includes slack variables ξ_i and ξ_i^* for each data point that allow some points to be outside the margin. The slack

variables measure how much the data points that are outside the margin deviate from the margin. An illustration of an SVR model with a soft margin is depicted in Figure 4.3. The goal is to find a function $f(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b$, where $\boldsymbol{w}$ is a weight vector, $\boldsymbol{x}$ an input feature vector and b is bias, that does not deviate from the support vectors more than the hard margin ε or soft margin $\varepsilon + \xi_i$ allows. The cost function minimizes the norm of $\boldsymbol{w}$ and the slack variables according to

$$J(\boldsymbol{w}) = \frac{1}{2} \|\boldsymbol{w}\|^2 + C \sum_{i=1}^{N} (\xi_i + \xi_i^*) \tag{4.13}$$

subject to

$$\begin{cases} y_i - \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle - b \leq \varepsilon + \xi_i \\ \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b - y_i \leq \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \geq 0, \end{cases}$$

where y_i is the target of the ith training sample and $C > 0$ is a regularization hyperparameter that prevents overfitting by defining the ratio between the flatness of the function and the deviations over ε. Equation (4.13) is known as the primal formula.

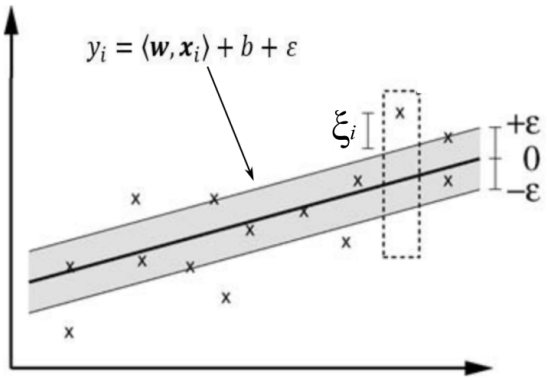

Figure 4.3: An illustration of a linear support vector regression model with a soft margin. The most distant data points act as support vectors. A slack variable ξ_i has been illustrated for one data point. Figure adapted from Ref. [54].

A dual formulation is obtained by constructing a Lagrangian function of the primal formula. The optimal solution (i.e., the weights) of the primal formula is given by the solution to the dual formula. Non-negative multipliers α, α^* are added for each feature vector $\boldsymbol{x}_i$ and the Lagrangian yields an optimization problem that is to maximize

$$L(\boldsymbol{\alpha}) = -\frac{1}{2} \sum_{i,j=1}^{N} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle - \varepsilon \sum_{i=1}^{N} (\alpha_i + \alpha_i^*) + \sum_{i=1}^{N} y_i (\alpha_i - \alpha_i^*) \tag{4.14}$$

subject to

$$\begin{cases} \sum_{i=1}^{N}(\alpha_i - \alpha_i^*) = 0 \\ 0 \leq \alpha_i, \alpha_i^* \leq C. \end{cases}$$

The weights are solved by $\boldsymbol{w} = \sum_{i=1}^{N}(\alpha_i - \alpha_i^*)\boldsymbol{x_i}$, and predictions are given by $f(\boldsymbol{x}) = \sum_{i=1}^{N}(\alpha_i - \alpha_i^*)\langle \boldsymbol{x}_i, \boldsymbol{x}\rangle + b$. A detailed derivation of the SVR algorithm and how b is solved can be found in Ref. [55].

The algorithm is now written in terms of the inner products between input features. This is the formula for a linear SVR, and it can be mapped into a non-linear space using a kernel trick. Instead of calculating the mapping of each feature vector $\Phi(\boldsymbol{x}_i)$ and calculating their inner product, the inner products can be replaced with a kernel function $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}_j)\rangle$. An example of a kernel function is the polynomial kernel function, which is defined as

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\gamma\langle \boldsymbol{x}_i, \boldsymbol{x}_j\rangle + r)^d, \tag{4.15}$$

where d is the degree, γ is a kernel parameter and r is an independent term. Mapping individual feature vectors requires more computing power than calculating the kernel function, so the kernel trick makes the computation more efficient. Traditional SVR does not support multi-target regression, but this is solved by creating an individual SVR model for each target variable.

4.3.4 k-nearest neighbors

The k-nearest neighbors algorithm [56] is a simple machine learning algorithm, where k stands for the number of neighbors. The algorithm predicts a target for a data point based on its k nearest neighbors in the training set. Standard k-nearest neighbors regression algorithm determines a target for a data point by calculating the mean value of the targets of the nearest neighbors. The nearness of a neighbor is commonly determined by their Euclidean distance from the data point. Each neighbor can have the same weight, or the neighbors can be weighted based on their distance from the data point. The k-nearest neighbors algorithm supports multi-target regression intrinsically.

Small values of k create a model with a high variance and a small bias, and larger values of k produce a model with a higher bias and a lower variance. As the distances between all data points need to be calculated, nearest neighbor models can be computationally quite expensive to train. However, this is not a problem for relatively small data sets. In addition to traditional regression tasks, a k-nearest neighbors algorithm can be used to fill NaN values in data [57]. A NaN value in a feature is filled with a value determined based on the k nearest neighbors that have a value for that feature.

4.3.5 Ensembles

Ensembles of models are used to combat problems of individual models. Ensembles are divided into two categories: averaging and boosting ensembles. If the models in the ensemble are different, use different subsets of the data, or are initialized randomly, their results can be averaged. A common averaging ensemble is a bagging ensemble [58]. The name comes from "bootstrap aggregating". In bagging, multiple models are built based on bootstrapped subsets of the training data. Bootstrapping means that the subsets are randomly sampled from the training data with replacement, so the same sample might appear in two or more subsets. The results of the individual models are then aggregated, by for example averaging them, to produce the final predictions. Bagging is a simple technique that is used to reduce the variance of otherwise strong models without increasing their bias.

An ensemble can also be built incrementally by training each new model so that it emphasizes the training samples that were not predicted well by the previous model. These kinds of ensembles are known as boosting ensembles. A common boosting algorithm is the Adaptive Boosting algorithm, commonly known as the AdaBoost algorithm [59]. The model is built based on a weak base estimator, such as a small decision tree. In the beginning, all of the training samples are weighted equally. The AdaBoost iteratively applies the base estimator to the training data, and the weights of the training samples are adjusted between each iteration. The weights of the training samples that were predicted less accurately during the previous iteration are increased. Correspondingly, the weights of the training samples that were predicted more accurately are decreased. Final predictions are made based on the weighted average of the estimators. Boosting is used not only to decrease the variance of a model but also to decrease its bias as it concentrates on the samples that are harder to predict.

4.3.6 Decision trees and random forests

Classification and regression trees are machine learning algorithms with a tree-like structure [60]. Decision trees consists of nodes that recursively split the feature space into subsets using "if-else" statements. The objective of a regression tree is to partition the training samples into M sets $S_1, S_2, ..., S_M$ so that they minimize

$$RSS = \sum_{m=1}^{M} \sum_{i \in S_m} (y_i - \hat{y}_{S_m})^2, \tag{4.16}$$

where y_i is the target of the ith training sample and $\hat{y}_{S_m}$ is the mean of the targets of the training samples in set S_m. Figure 4.4 visualizes a regression tree, where $M = 5$. Because it would be computationally impossible to evaluate every possible split across

an entire tree, the splits are determined in a greedy manner, meaning that the algorithm always chooses the locally best split at each node. At the beginning of training, or "growing" a regression tree, all of the samples in the training set belong to the same set. At each node, a set of data $\boldsymbol{X}$ is split into two subsets S_1 and S_2 based on a feature j and a threshold value t for that feature. The algorithm finds the splitting parameters j and t that minimize the cost function

$$J(\boldsymbol{X}, j, t) = \sum_{i:\, \boldsymbol{x}_i \in S_1(j,t)} (y_i - \hat{y}_{S_1})^2 + \sum_{i:\, \boldsymbol{x}_i \in S_2(j,t)} (y_i - \hat{y}_{S_2})^2, \tag{4.17}$$

where y_i is the target corresponding to the ith feature vector $\boldsymbol{x}_i$ in the training set, $\hat{y}_{S_1}$ is the mean target of the training samples in set S_1, and $\hat{y}_{S_2}$ is the mean target of the training samples in set S_2. After the split, the same process is applied to the two new branches, and the tree is built downwards. Eventually, a terminal node known as a "leaf", which contains the final output, is reached. A node becomes terminal if a user-defined maximum number of nodes is reached, a user-defined minimum node size is reached, or a new split would no longer improve the accuracy. The final prediction is the mean target of the training samples in the leaf.

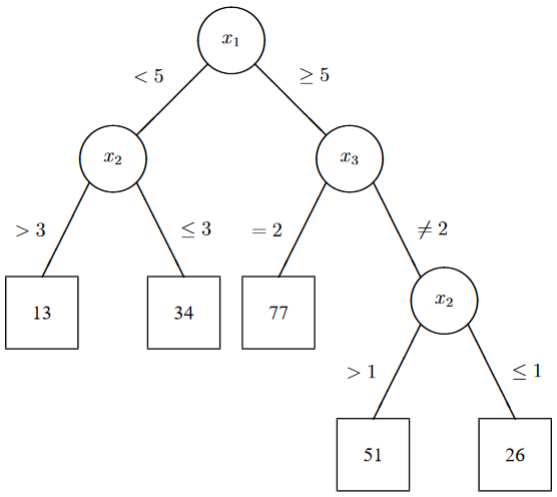

Figure 4.4: An illustration of a regression tree. Each node, represented by a circle, finds the split that minimizes the cost function. The splits are made by selecting the optimal feature j and threshold value t at each node. The first node splits the data based on the feature x_1, and the threshold value is 5. The final decisions are represented by squares. Figure taken from Ref. [60].

Regression trees can be extended to support multi-target regression. One tree, known as a multi-target regression tree, can predict multiple outputs at the same time. Separate trees could be built for each target variable, but multi-target regression trees are smaller in size and take into account the correlations between target variables [61]. A single-target tree is expanded into a multi-target tree by making two changes. The

leaves need to be expanded to contain output vectors instead of scalars, and the cost function needs to be adjusted so that it is summed over the target variables.

Single decision trees typically suffer from sensitiveness to small variations in the input data. A tree also tends to overfit the training data, especially if it has many nodes. One way to battle this is to set constraints to the size of the tree. However, if the growing process is cut too early based on one locally bad split, the following, possibly important splits are never reached. This problem is solved by first building a large tree and then pruning it. The minimal cost-complexity pruning algorithm removes the weakest subtrees [60]. Each node starts a new subtree, so pruning strives to minimize the number of leaves while simultaneously minimizing the error of the tree. A pruning hyperparameter, known as the complexity parameter α, determines the ratio between the two terms that are minimized. The larger the α, the more the tree is pruned, and the more the error increases.

Random forests are bagged ensembles of decision trees [62]. The trees in random forests are trained with subsets sampled from the training set with replacement, and the split at each node is determined based on a random subset of features. Extra-tree ensembles [63] are similar to random forests but contain more randomness. Extra-tree ensembles sample the data without replacement and the best split at each node is selected from a group of random splits determined based on randomly selected features. Random forests reduce the overfitting done by individual decision trees. When multiple different decision trees are collected into an ensemble of trees that overfit in different ways, the collective overfitting is reduced. Random forests can be extended to predict multiple targets like single decision trees.

The depth of a decision tree, where a feature is used to define a split, defines the importance of that feature. If the feature is used at the top of the tree, it affects the final decisions more than the features used at the nodes closer to the leaves. Single decision trees might produce significantly different results, but averaging the feature importance values of trees in an ensemble produces robust values corresponding to feature importance. Thus, ensembles of decision trees act as feature selection algorithms. However, the method is not foolproof: the feature importance is determined based on the training set only, and features with many unique values can bias the results [64].

4.3.7 Artificial neural networks

Artificial neural networks (ANNs) bear a resemblance to the brain. A human brain consists of billions of neurons that are connected with nerve cells. In turn, neural networks consist of layered artificial neurons that are connected with weights. The first and the simplest artificial neuron is known as the perceptron [65]. A perceptron

takes numerical values as input, calculates the weighted sum of the inputs, applies a step function to the sum, and then outputs the result. The perceptron is a supervised learning algorithm and can perform binary classification tasks. When three or more layers of perceptrons are connected, they form a multi-layer perceptron.

Since the 1950s, modern multi-layer neural networks have been developed based on the perceptron. Just like their biological counterpart, artificial neural networks are powerful tools that can be used to solve a wide range of problems. In a multi-layer neural network, the layers between the input and output layers are called hidden layers. If a neural network has two or more hidden layers, it is called a deep neural network. The number of layers and their sizes are hyperparameters. Multi-target regression is implemented by setting the size of the output layer to the number of target variables. The structure of a deep neural network is illustrated in Figure 4.5. A comprehensive coverage of deep neural networks can be found in Ref. [66].

A neuron j takes the outputs from the previous layer as an input vector $\boldsymbol{x}$, where $x_0 = 1$, and the output of the neuron is given by

$$f(\boldsymbol{x}) = \phi(\boldsymbol{x}^{\mathsf{T}}\boldsymbol{w}_j), \tag{4.18}$$

where ϕ is a layer-specific activation function, which is usually non-linear, and $\boldsymbol{w}_j$ is the weight vector, which includes the bias as $w_0 = b$. The output can then be given as an input to the neurons of the following layer. A layer is called a dense layer if each neuron receives an input from all the neurons in the previous layer. This is known as the forward phase because the neural network is traversed in the forward direction, from the input layer towards the output layer. If the input data travels only in the forward direction, the neural network is a feedforward neural network.

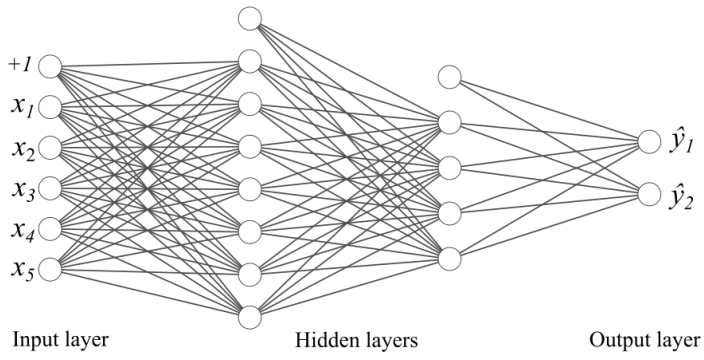

Figure 4.5: An illustration of a feedforward neural network with dense layers. The neural network has two hidden layers. The number of layers and their sizes are defined by the user. This neural network takes five inputs and gives two outputs. The neurons that do not receive any input are bias neurons. Figure is adapted from a schematic created with the online tool in Ref. [67].

Neural networks are trained by a backpropagation algorithm [68]. An optimization algorithm, such as the adaptive gradient descent algorithm *Adam*, is used to find the weights that minimize the cost function. Common cost functions in regression are MSE and MAE. The gradient is calculated backwards through the layers of a network, starting from the output layer and propagating through the network until the input layer is reached. The cost function of a layer k depends on the cost function of the next layer in the forward direction $k+1$, so the gradient calculations that were made at layer $k+1$ can be used in the calculations of layer k. For this reason, using backpropagation is more efficient than calculating the gradient separately for each layer in the forward direction. The backpropagation step is known as the backward phase, and it is performed after the forward phase to complete one iteration of the gradient descent.

Three hyperparameters define the training process: the learning rate of the optimizer, the batch size, and the number of epochs. The batch size determines the number of samples the gradient descent algorithm uses to update the model parameters during one iteration. The number of epochs is the number of times the algorithm runs through the whole training set. The number of epochs should be large enough so that the model converges to its optimum but small enough so that the model does not overfit.

Regularization is used to prevent overfitting. A regularization penalty can be added to the cost function of a layer via a regularization term that strives to minimize the sum of the absolute values of the weights (L1 regularization) or the sum of the squared values of the weights (L2 regularization). The regularization strength can be adjusted through a hyperparameter λ, known as the regularization factor. Another effective method for regularization is dropout [69]. During training, a dropout layer keeps a neuron active with a user-defined probability p and drops out a neuron by setting it to zero with a probability $1-p$. This means that at each training iteration, a random part of the neural network is ignored. The dropout is not applied during testing, and an inverted dropout scales the active neurons by $1/p$ during training so that the amount that each neuron contributes to the output stays the same at training and test phases.

The unit step function is not differentiable at zero and its derivative is zero elsewhere, so the step function in perceptrons had to be replaced with a different activation function in order to enable gradient-based learning. Two commonly used activation functions are the rectified linear unit (ReLU) function and the logistic sigmoid (σ) function, which are defined as

$$\mathrm{ReLU}(z) = \max(0, z) \tag{4.19}$$

$$\sigma(z) = \frac{1}{1+\exp(-z)}. \tag{4.20}$$

A newer addition to the family of activation functions is the scaled exponential linear unit (SELU) function, which is expressed as

$$\text{SELU}(z) = \lambda \begin{cases} \alpha(e^z - 1) \text{ if } z \leq 0 \\ z \qquad\qquad \text{if } z > 0, \end{cases} \tag{4.21}$$

where λ and α are pre-defined constants, $\lambda \approx 1.0507$ and $\alpha \approx 1.67326$. SELUs form self-normalizing neural networks and combat problems encountered when working with ReLUs, which include them dying and constantly outputting zero [70]. The activation function is chosen depending on the task at hand at each layer. Because the ReLU function returns values in the range $[0, \infty)$, it is commonly used in the hidden layers and the input layer. The SELU function returns normalized values, meaning that the outputs have a mean of zero and unit variance. The sigmoid function returns values in the range (0,1) and hence is used in the output layer in classification tasks, where the output should be a probability. No activation function is needed in the output layer in regression tasks.

5. Models built using the NOMAD Database

This thesis studies how models built by six different machine learning regression algorithms can be used in predicting the DBTT and USE based on the NDE parameters in the NOMAD Database. The following six models, which are identified by the algorithm that was used to build them, are evaluated:

- Ridge regression (RR)
- Support vector regression (SVR)
- k-nearest neighbors (kNN)
- Boosted decision tree (BDT)
- Extra-trees regressor (ETR)
- Artificial neural network (ANN)

The first section of this chapter describes the data set that was used to build the models and the general data preprocessing steps taken before the algorithms were applied to the data. The second section describes the data preprocessing steps characteristic to each model and the optimized form of each model.

5.1 Data

The version of the NOMAD Database used in this thesis contains 89 samples, which are described by 69 variables. Nine of the variables are used to identify the samples and describe the irradiation conditions of the samples. The rest of the variables consist of the 28 NDE parameters, their experimentally determined standard deviations, the DBTT and USE, and their standard deviations. It must be pointed out that while the parameters in the NOMAD Database are named "parameters" in the sense that they measure the characteristics of a sample, in the machine learning context the word "parameter" means the model parameters and hyperparameters.

5.1.1 Preprocessing

A data set was created by extracting the NDE parameters and the DBTT and USE data from the NOMAD Database. Additionally, two of the identifying parameters in the NOMAD Database, Base_Weld and Material_ID, were included in the data set because different materials have different characteristic DBTT and USE values. These are categorical variables, so they were one-hot encoded. The data set was split into targets and features. The targets consisted of the DBTT and USE, and the features consisted of the 28 NDE parameters and the one-hot encoded variables for Base_Weld and Material_ID. None of the NDE features were removed from the analysis at this point because feature selection will be studied during the development of the final NOMAD Tool when more data is available.

One of the one-hot encoded features can be removed. For example, in the case of one-hot encoded features for the base and weld, the category of a sample can be deduced using only one of the two encoded features. Removing one of the one-hot encoded features can improve the performance of some algorithms. This means that the number of one-hot encoded features was either four or six, depending on whether one of the encoded features was left out or not.

The NOMAD Database contains several NaN values because some samples have not been measured with every NDE method. All samples that contained over six NaN NDE features were removed from the data set, and one sample was removed because its NDE measurements are considered to contain errors. This reduced the number of samples by 16, so the number of samples used in the analysis was 73. The rest of the NaN values were filled with a k-nearest neighbors algorithm. The number of neighbors k was set to 2, and the distances of the neighbors were used as weights. The filled values should not be included in the test set, as the test set should consist of original data points. A temporary binary column was added to the data set to indicate if a sample contained filled values or not. The number of samples that contained filled values was 35 out of 73.

The data set was split into training and test sets so that the test set was randomly sampled from the data points that did not contain filled values. The size of the test set was 20 % of the size of the data set, and the rest 80 % of the data constituted the training set. The training set contained 59 samples, and the test set contained 14 samples. The standard deviations in the NOMAD Database were not included in the data set, but the standard deviations of the NDE parameters were saved for later use in studying the stability of the models.

5.2 Models

After preprocessing, the training set was used to train the six models. All of the models except the ANN were implemented using scikit-learn [71], which is an open-source Python library. Additionally, the k-nearest neighbors algorithm that was used to fill the NaN values was implemented with scikit-learn using its KNNImputer class. Scikit-learn provides many machine learning algorithms with an intuitive and straightforward interface. The algorithms in scikit-learn come with default hyperparameters that can, and should, be fine-tuned by the user.

The ANN was implemented using open-source TensorFlow 2 [72] and Keras [73] frameworks. TensorFlow 2 is a machine learning library released by Google and is commonly used for neural network applications. It executes machine learning tasks using high-performance C++ but uses user-friendly Python as an application programming interface. Keras is a neural network library written in Python, and it enables the fast and modular implementation of neural networks by running on top of TensorFlow 2. Keras provides a high-level Python application programming interface for TensorFlow 2 and makes building neural networks more user-friendly.

The characteristic data preprocessing procedures and the model hyperparameters were optimized by performing a 10-fold cross-validation with the training set. Different preprocessing choices, specifically how the features were scaled and if one of the one-hot encoded features was removed or not, and hyperparameter combinations were evaluated using a cross-validation score MAE, and the best combination was implemented. An exhaustive grid search was utilized to find the optimal hyperparameter ranges when using scikit-learn. The hyperparameters of the ANN were fine-tuned manually, i.e., by trial and error. The source code of the optimized models can be found in Appendix C.

5.2.1 Ridge regression

All one-hot encoded features were included in the analysis. The features were normalized by scaling them between zero and one according to Equation (4.1). The model is a kernelized regression model, and it replaces the original feature vectors with vectors that consist of the polynomial combinations of the features with a degree of two or less. As an example, a feature vector of the ith sample consisting of two features $(x_{i,1}, x_{i,2})$ would be transformed according to

$$(x_{i,1}, x_{i,2}) \rightarrow (1, x_{i,1}, x_{i,2}, x_{i,1}^2, x_{i,1}x_{i,2}, x_{i,2}^2). \tag{5.1}$$

The hyperparameter λ of Equation (4.9), which is denoted as "alpha" in the source code and determines the strength of the ridge regularization, was set to 1.

5.2.2 Support vector regression

All one-hot encoded features were included in the analysis. The features were normalized by scaling them between zero and one according to Equation (4.1). Because the SVR algorithm in scikit-learn does not intrinsically support multi-target regression, two SVR models with identical hyperparameters were combined to predict the DBTT and USE simultaneously. The model uses a polynomial kernel, which is defined in Equation (4.15). The kernel parameter γ is defined by the number of features n_{features} and the variance of the feature matrix $\text{Var}(\boldsymbol{X})$ according to

$$\gamma = \frac{1}{n_{\text{features}} \text{Var}(\boldsymbol{X})}. \tag{5.2}$$

The degree of the kernel function was set to $d = 2$, and the independent term of the kernel function was set to $r = 0.7$. The regularization parameter was set to $C = 100$. The width of the margin was set to $\varepsilon = 10$.

5.2.3 k-nearest neighbors

All one-hot encoded features were included in the analysis. The features were normalized by scaling them between zero and one according to Equation (4.1). The number of neighbors was set to $k = 2$. The distance between neighbors was calculated as the Euclidean distance, and the model uses brute force to compute the distances between data points. The neighbors were weighted based on their distance. Because the data set was small, the model was not particularly slow to train.

5.2.4 Boosted decision tree

All one-hot encoded features were included in the analysis. The features were standardized by scaling them according to Equation (4.2) so that they had a mean of zero and unit variance. The boosting ensemble uses the AdaBoost algorithm. Two AdaBoost models with identical hyperparameters were combined to predict the DBTT and USE simultaneously. The model was used to assess the importance of the features, and five least important features were removed from the analysis. The following NDE features were removed: K, DZmax, DZmean, DU25dz and Phizmean.

A simple regression tree stump with maximum depth set to 3 was used as the base estimator. The maximum number of estimators was set to 300, so the number of boosts was 299. The learning rate was set to 0.1. The model uses the squared loss to calculate the cost function.

5.2.5 Extra-trees regressor

One of the one-hot encoded features of both categorical features was removed. The features were normalized by scaling them between zero and one according to Equation (4.1). The model was used to assess the importance of the features, and five least important features were removed from the analysis. The following NDE features were removed: Resistance_3, P3, A3, K and DU75dz.

The maximum depth of a tree was set to 9. The maximum number of features that the model uses to determine a split was set to the number of features, 27. The complexity parameter α, which controls the pruning process, was set to 3. The number of trees in the ensemble was set to 100.

5.2.6 Artificial neural network

All one-hot encoded features were included in the analysis. The features were standardized by scaling them according to Equation (4.2) so that they had a mean of zero and unit variance. The neural network has two dense hidden layers in addition to the input and output layers. The size of the input layer was set to the number of features, 32, and the sizes of the hidden layers were set to 25 and 22. The size of the output layer was set to 2. Both hidden layers use the SELU activation function, which is defined in Equation (4.21), and the output layer has no activation function. The hidden layers were L2 regularized, and the regularization factors of the layers were set to 1 and 0.01. An inverted dropout layer follows both hidden layers. The dropout frequencies, which determine the fractions of neurons to drop, of the two dropout layers were set to 0.15 and 0.1. *Adam* with a learning rate of 0.001 was used as an optimizer, and the model uses the MAE loss. The number of epochs was set to 600, and the batch size was set to 10.

6. Results

6.1 Test accuracy

The accuracies are reported as MAEs between the predicted targets and the true targets. Once the models had been trained, their generalization ability was studied with the test set. Figures 6.1–6.6 illustrate the training and test accuracies of the models. The predicted targets are plotted as a function of the corresponding true targets as scatter plots separately for the DBTT and USE. The training and test accuracies are reported in the figure headings.

Each model predicted the DBTT and USE simultaneously with a satisfactory degree of accuracy. The performance of the models was quite consistent: all test accuracies were smaller than 22 °C for the DBTT and smaller than 14 J for the USE. The training accuracy is reported, even though it does not measure the generalization ability of a model, because it gives information about the possibility that the model was overfitting. A model with a small training error is likely to be overfitting and will not generalize well. For example, the training accuracy of the DBTT was worse for the BDT than for the ETR as can be seen from Figures 6.4 and 6.5, but the corresponding test accuracy of the BDT was better than the test accuracy of the ETR. On the other hand, the training error of the kNN was zero, but the model generalized well when tested with the test set, as shown in Figure 6.3.

Figures 6.1–6.6 show that the test accuracy is better at higher values of DBTT and at lower values of USE for all models. This means that the models predict the embrittlement of more irradiated samples more accurately than the embrittlement of non-irradiated samples or less irradiated samples. This is a consequence of the split between the training and test sets. The randomly sampled training set contained fewer samples with low embrittlement levels.

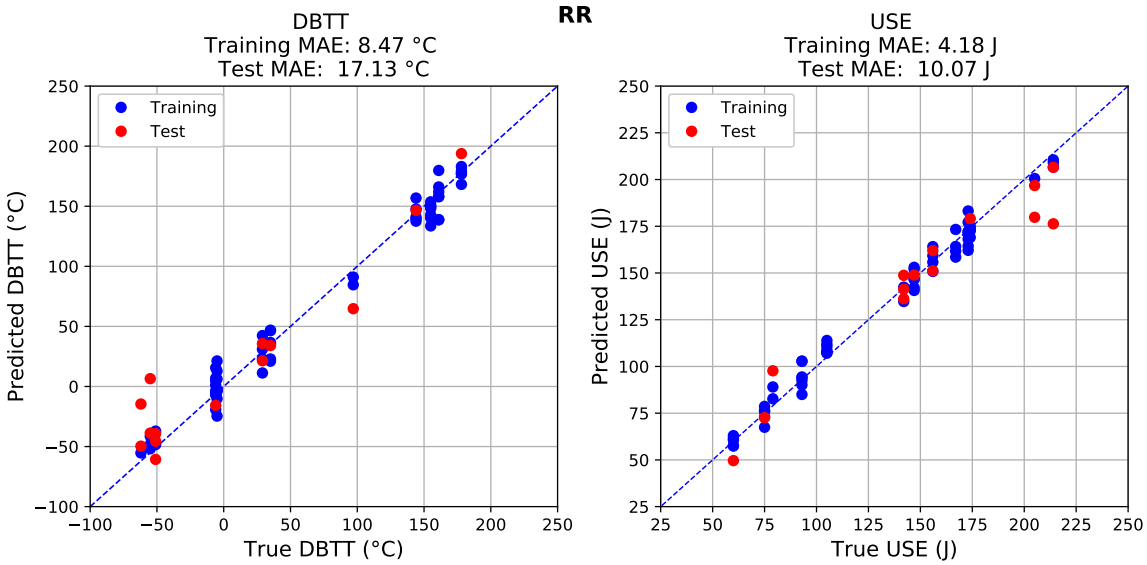

Figure 6.1: Scatter plots of the targets predicted by the model built with the ridge regression algorithm as a function of the true targets. The values of the ductile-to-brittle transition temperature (DBTT) are plotted on the left and the values of the upper-shelf energy (USE) on the right. The blue dots represent the training set, and the red dots represent the test set. The diagonal lines represent perfect fits. The training and test accuracies, measured as mean absolute error (MAE), are reported in the headings rounded to two decimal places.

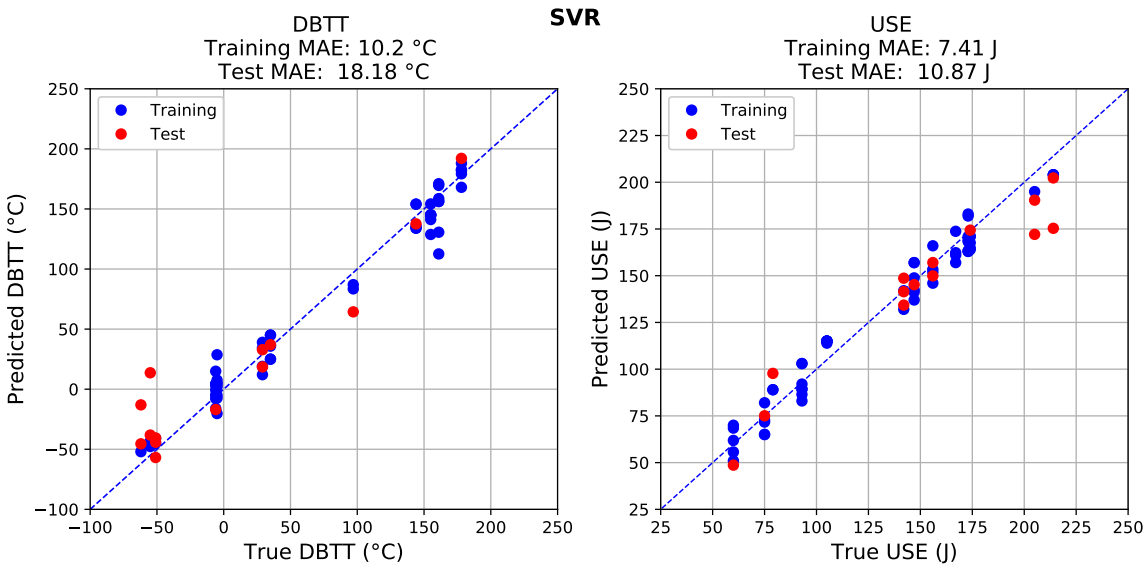

Figure 6.2: As for Figure 6.1 but for the support vector regression algorithm.

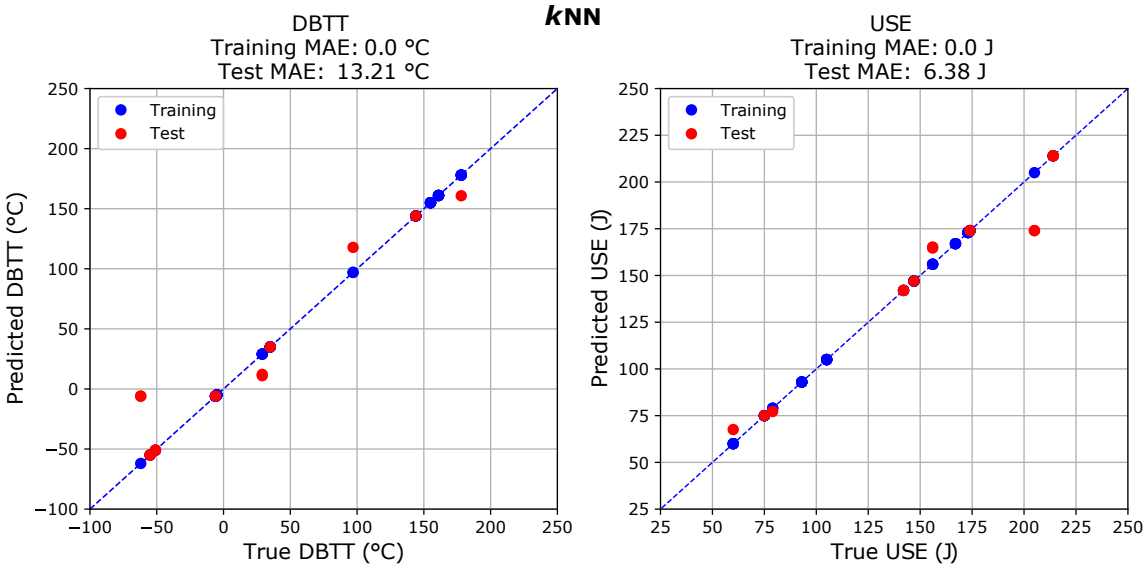

Figure 6.3: As for Figure 6.1 but for the k-nearest neighbors algorithm.

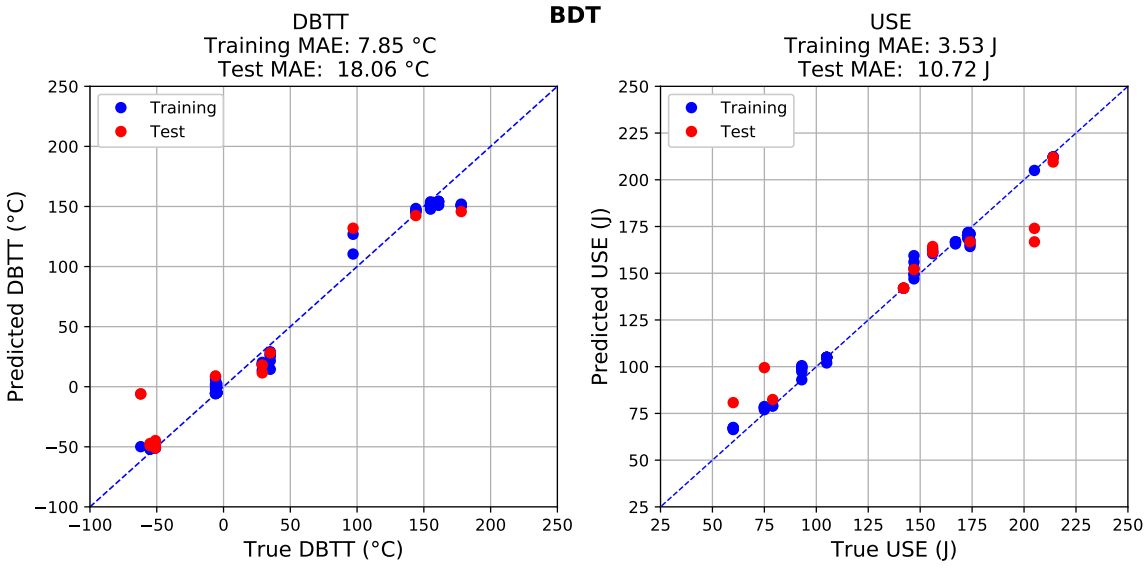

Figure 6.4: As for Figure 6.1 but for the boosted decision tree algorithm.

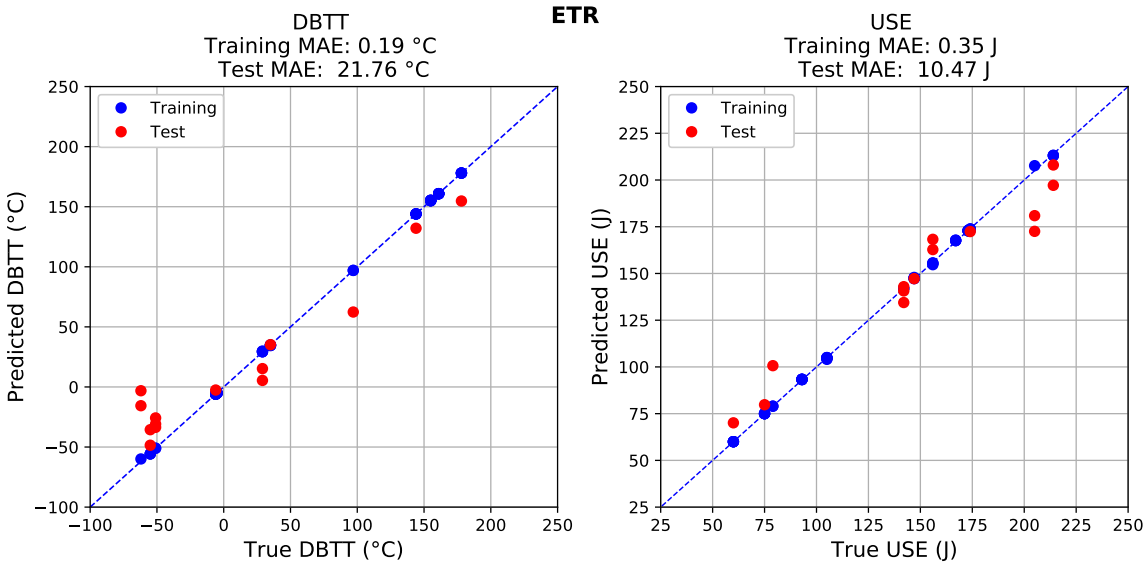

Figure 6.5: As for Figure 6.1 but for the extra-trees regressor algorithm.

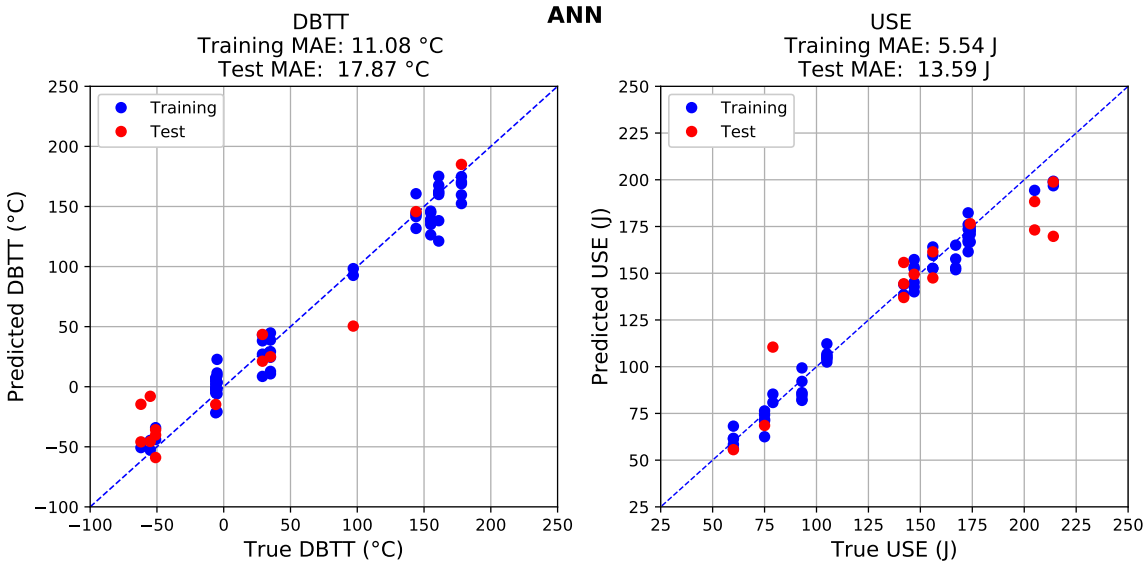

Figure 6.6: As for Figure 6.1 but for the artificial neural network algorithm.

6.2 Model stability

The stability of the models was studied by adding random noise to the features of the training and test sets. The motive for this procedure was to estimate how the models would perform in real-life applications, where small changes in the input data due to, e.g., measurement uncertainty are often unavoidable. Each NDE feature of each sample was replaced with a value that was randomly sampled from a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$. The mean μ was set to the value of the feature, and the standard deviation σ was set to the experimentally determined standard deviation of the NDE measurement in question. These new noisy training and test features were then used as inputs to the models that had already been trained with the original training set. This was repeated one thousand times with different random noise at each iteration. To make the results comparable across the models, the thousand iterations of random noise were the same for all models. The mean values and standard deviations of the training and test accuracies are reported in Table 6.1.

Table 6.1: Model stability was evaluated by giving noisy input data to the models that had been trained with the original training set. This was repeated one thousand times with different noise. The mean values and standard deviations of the training and test accuracies, measured as mean absolute error (MAE), are presented in this table separately for the ductile-to-brittle transition temperature (DBTT) and upper-shelf energy (USE). The values are rounded to two decimal places.

Model	**Training MAE**		**Test MAE**	
	DBTT (°C)	USE (J)	DBTT (°C)	USE (J)
RR	13.35 ± 1.22	6.77 ± 0.64	19.32 ± 2.79	10.97 ± 1.46
SVR	13.87 ± 1.21	8.5 ± 0.66	20.04 ± 2.73	12.15 ± 1.37
kNN	1.65 ± 0.47	0.7 ± 0.22	15.3 ± 1.77	7.1 ± 0.67
BDT	9.92 ± 0.84	5.08 ± 0.54	19.64 ± 1.53	11.75 ± 1.21
ETR	5.75 ± 0.92	2.67 ± 0.43	22.97 ± 1.51	11.23 ± 0.82
ANN	15.87 ± 1.51	8.24 ± 0.74	20.88 ± 3.72	14.77 ± 2.19

All models respond in a similar way to the noisy input data. Compared to the accuracies calculated based on the original data reported in the headings of Figures 6.1–6.6, the training accuracy increases on average by 4 °C for the DBTT and by 2 J for the USE. The test accuracy increases on average by 2 °C for the DBTT and by 1 J for the USE. Out of all analyzed models, the ANN has the largest standard deviations. The differences between the two decision tree-based models are subtle, but overall, the BDT has a slightly larger standard deviation than the ETR. In general, the performance of the models trained with the original data does not substantially change when small amounts of noise are added to the input data.

6.3 Feature importance

Ensembles of decision trees can be used to study the importance of the input features as was outlined in section 4.3.6, so the BDT and ETR can be used for this purpose. Before any of the features were removed from the analysis of the BDT and ETR, the importance of the features was studied. Figure 6.7 shows the importance of the NDE and material features as bar plots. The results given by the two models are somewhat different, but some features stand out clearly. For example, among the less important NDE features are the K, DU75dz and Phizmean. According to both models, the most important NDE feature is the SC. However, the results given by this method are not absolute: the purpose is to show that machine learning algorithms can be used to make a distinction between better and worse NDE features and compare the features simultaneously. Decision tree ensembles could be utilized in removing the least important NDE parameters from the NOMAD Database during the development stages of the NOMAD Tool.

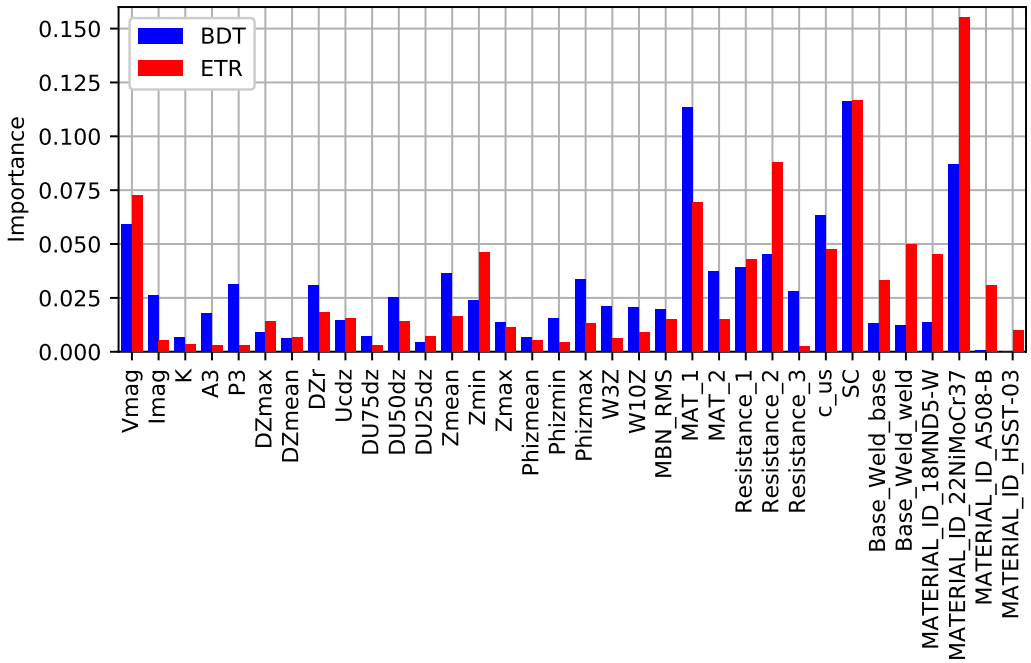

Figure 6.7: The relative importance of the features evaluated by the models built with the boosted decision tree (BDT) algorithm and the extra-trees regressor (ETR) algorithm plotted as bar plots. Because the BDT model consists of two models that estimate the ductile-to-brittle transition temperature and upper-shelf energy separately, the values given by these two models have been averaged in this plot.

7. Conclusions and discussion

The objective of this thesis was to study how different kinds of machine learning regression algorithms could be utilized in developing the NOMAD Tool. Six machine learning regression algorithms were applied to data in the NOMAD Database and used to build models that predict the embrittlement of RPV steel alloys based on NDE parameters. The six resulting models predicted the DBTT and USE with test accuracies, which are measured as MAEs, of approximately 20 °C and 10 J, respectively. The stability of the models was studied by adding noise that emulates experimental uncertainty to the input data. The models were stable even though manipulating the input data made the accuracies of the models slightly worse.

The test accuracies of the models are satisfactory, as the ranges of the DBTT and USE are quite large, and their destructively measured values have uncertainties of the same scale. The experimentally determined standard deviations of the DBTT and USE in the NOMAD Database are on average 12 °C and 10 J, respectively.

The models performed quite uniformly, but if one algorithm had to be selected based on this study to act as the basis of the NOMAD Tool, the SVR algorithm would be a robust choice. A major advantage of the SVR is that the model was significantly simpler to train than the decision tree-based models and the ANN, which required more hyperparameters to be fine-tuned.

Furthermore, feature selection was studied through the decision tree-based models. The NOMAD Tool will use the combination of NDE features that yields the best performance, so the tool might not include all of the NDE parameters that are in the NOMAD Database. Based on the results of this thesis, decision tree ensembles could be used to select the most important NDE features for the NOMAD Tool, but this requires further research. Other aspects are worth considering in addition to feature importance when choosing the NDE features for the NOMAD Tool, such as the practicality of each NDE technique in field.

It should be noted that because the processes of training the models contained varying amounts of trial and error, making comparisons between the different algorithms is unfair. Although a grid search was utilized, it is not certain that the most optimal hyperparameters were selected for every model. The coincidental nature of

fine-tuning the hyperparameters was pronounced in the case of the ANN, which was fine-tuned entirely manually. It is speculated that the hyperparameters of the ANN could have been tuned more, as the model proved to be somewhat more unstable than the other models. In addition to the hyperparameters, some of the data preprocessing steps were optimized through trial and error. The performance of the decision tree ensembles might have been better if a different data preprocessing path had been taken. However, these are just speculations made afterwards and could as well have made the models worse by overfitting or underfitting the data.

A major drawback of this study is the uncompleted status of the NOMAD project. According to plans, feature selection will be performed in the future after all measurements have been finalized. For this reason none of the NDE features were removed from the analysis of this thesis, which made the data sparse. Additionally, due to the incomplete measurements, the NOMAD Database used in this thesis is very small: it contains only 89 samples and covers only 13 different embrittlement levels. This undoubtedly set strict limits for the performance of the models. Although no lower limit has been specified, data sets that are used to train machine learning models commonly contain thousands of samples. Considering the amount of data available, the models performed surprisingly well.

Lastly, it must be pointed out that the NOMAD Database hardly represents the real-life application of the NOMAD Tool. In reality, the NDE measurements would be made through the cladding of the RPV and would not be affected by the size of a Charpy sample. It remains unclear how the cladding will affect the measurements and if the results from Charpy samples can be extrapolated to actual RPV walls. Thus, the idea of using the final version of the NOMAD Tool for monitoring RPV embrittlement in realistic conditions is still a thing of the future. Meanwhile, a beta version of the NOMAD Tool or even an individual NDE method could be applied to surveillance Charpy samples during annual maintenance outages along with the usual surveillance procedures. This would collect additional information that could be used to describe material degradation or study the performance of the NDE method.

In conclusion, machine learning regression algorithms have been proven to be able to produce models that estimate the embrittlement of RPV steel alloys based on the NDE parameters in the NOMAD Database with sufficient accuracy. Machine learning algorithms were also useful in evaluating the importance of the input features. This research will continue in the framework of the NOMAD project and take a forward step when the NOMAD Database is extended with more measurements. The increased amount of data, especially if it includes new irradiation levels, will allow further development of the models.

Appendix A. Identification, irradiation and destructively measured parameters in the NOMAD Database

Table A.1: The identification, irradiation and destructively measured parameters in the NOMAD Database. Table contents taken from Ref. [33].

Abbreviation	Full name or description	Type or unit
Sample and material parameters		
SQL_ID	Line ID for SQL database	Running integer
Sample_ID	ID carved into the sample	String
Material_ID	Sample material	String
Track_Number_ID	ID in NOMAD tracking system	Integer
Base_Weld	Is the sample material base or weld	"Base" or "Weld"
Irrad_program	Irradiation program	CHIVAS-X
Irradiation parameters		
T_Irrad	Irradiation temperature	°C
F_Irrad	Fluence of fast (energy >1 MeV) neutrons	n/cm^2
Destructively measured parameters		
DBTT	Ductile-to-brittle transition temperature	°C
USE	Upper-shelf energy	J

Appendix B. Nondestructively measured parameters in the NOMAD Database

Table B.1: The nondestructively measured parameters in the NOMAD Database, which have been obtained using six different nondestructive evaluation techniques. Table contents taken from Ref. [33].

Abbreviation	Full name	Unit
MIRBE		
MBN_RMS	Magnetoelastic parameter	Arbitrary
3MA, harmonic analysis		
Vmag	Amplitude of the magnetizing voltage	V
Imag	Amplitude of the magnetizing current	A
K	Distortion factor	%
A3	Amplitude of the third harmonic	%
P3	Phase shift of the third harmonic in relation to the fundamental wave	rad
3MA, incremental permeability		
DZmax	Maximum amplitude of the incremental permeability curve	Ω
DZmean	Mean value of the incremental permeability curve	Ω
DZr	Value of the incremental permeability curve in remanence	Ω
Ucdz	Excitation voltage at the incremental permeability curve maximum	V
DU75dz	Width of the incremental permeability curve at 75 % of the maximum amplitude	V
DU50dz	Width of the incremental permeability curve at 50 % of the maximum amplitude	V
DU25dz	Width of the incremental permeability curve at 25 % of the maximum amplitude	V

Continued on the next page

Abbreviation	Full name	Unit
3MA, eddy current impedance		
Zmean	Mean impedance of the impedance loop	Ω
Zmin	Minimum impedance of the impedance loop	Ω
Zmax	Maximum impedance of the impedance loop	Ω
Phizmean	Phase angle of the mean impedance	rad
Phizmin	Phase angle of the minimum impedance	rad
Phizmax	Phase angle of the maximum impedance	rad
W3Z	Width of the impedance loop at 3 % of the maximum	Ω
W10Z	Width of the impedance loop at 10 % of the maximum	Ω
MAT		
MAT_1	Magnetic permeability 1: normalized	Arbitrary
MAT_2	Magnetic permeability 2: not normalized	Arbitrary
DCRPD		
Resistance_1	Resistance measured with sequence 1: whole sample	Ω
Resistance_2	Resistance measured with sequence 2: half sample	Ω
Resistance_3	Resistance measured with sequence 3: half sample	Ω
TEPMM		
SC	Seebeck coefficient	nV/K
Ultrasonic method		
c_us	Ultrasonic velocity	m/s

Appendix C. Source code of the models

The ridge regression implemented with scikit-learn:

```
make_pipeline(PolynomialFeatures(degree=2, include_bias=True,
                                 interaction_only=False, order='C'),
              Ridge(alpha=1.0, copy_X=True, fit_intercept=True,
                    max_iter=None, normalize=False, random_state=None,
                    solver='auto', tol=0.001))
```

The support vector regression implemented with scikit-learn:

```
MultiOutputRegressor(SVR(C=100, cache_size=200,
                         coef0=0.7, degree=2, epsilon=10,
                         gamma='scale', kernel='poly',
                         max_iter=-1, shrinking=True,
                         tol=0.001, verbose=False))
```

The boosted decision tree implemented with scikit-learn:

```
MultiOutputRegressor(AdaBoostRegressor(
                     base_estimator=DecisionTreeRegressor(ccp_alpha=0.0,
                             criterion='mse',
                             max_depth=3,
                             max_features=None,
                             max_leaf_nodes=None,
                             min_impurity_decrease=0.0,
                             min_impurity_split=None,
                             min_samples_leaf=1,
                             min_samples_split=2,
                             min_weight_fraction_leaf=0.0,
                             presort='deprecated',
                             random_state=None,
                             splitter='best' ),
                     n_estimators=300,
                     learning_rate=0.1,
                     loss='square',
                     random_state=42))
```

The extra-trees regressor implemented with scikit-learn:

```
ExtraTreesRegressor(bootstrap=False, ccp_alpha=3,
                    criterion='mse', max_depth=9,
                    max_features='auto',
                    max_leaf_nodes=None, max_samples=None,
                    min_impurity_decrease=0.0,
                    min_impurity_split=None,
                    min_samples_leaf=1, min_samples_split=2,
                    min_weight_fraction_leaf=0.0,
                    n_estimators=100, n_jobs=None,
                    oob_score=False, random_state=42,
                    verbose=0, warm_start=False)
```

The *k*-nearest neighbors implemented with scikit-learn:

```
KNeighborsRegressor(algorithm='brute', leaf_size=30, metric='minkowski',
                    metric_params=None, n_jobs=None, n_neighbors=2, p=2,
                    weights='distance')
```

The neural network implemented with TensorFlow 2 and Keras:

```
def ANN(input_dim):
    activation = "selu"
    inputs = keras.Input(shape=(input_dim))
    dense = layers.Dense(25, activation=activation,
                         kernel_regularizer=regularizers.l2(1))
    x = dense(inputs)
    layer = layers.Dropout(.15)
    x = layer(x)
    x = layers.Dense(22, activation=activation,
                     kernel_regularizer=regularizers.l2(0.01))(x)
    layer = layers.Dropout(.1)
    x = layer(x)
    outputs = layers.Dense(2)(x)
    model = keras.Model(inputs=inputs, outputs=outputs)
    optimizer = Adam(lr=0.001)
    model.compile(loss="mae" , optimizer=optimizer, metrics=["mae"])
    return model
epochs = 600
batch_size = 10
```

References

[1] European Council. European Council meeting (12 December 2019) – Conclusions, 2019.

[2] Eurostat. Electricity production, consumption and market overview. /https://ec.europa.eu/eurostat/statistics-explained/index.php/Electricity_production,_consumption_and_market_overview#Electricity_generation, June 2020. Accessed 31.7.2020.

[3] The NOMAD Consortium. NOMAD Non-destructive Evaluation (NDE) System for the Inspection of Operation-Induced Material Degradation in Nuclear Power Plants STATUS REPORT June 2020. /https://www.nomad-horizon2020.eu/news/nomad-status-report-available/NOMAD_status%20report_2020.pdf. Accessed 11.9.2020.

[4] Ph. G. Tipping. *Understanding and mitigating ageing in nuclear power plants: Materials and operational aspects of plant life management (PLiM), Chapter 2: Key elements and principles of nuclear power plant life management (PLiM) for current and long-term operation.* Woodhead Publishing Series in Energy, Woodhead Publishing Limited, 2010.

[5] IAEA. PRIS Power Reactor Information System. /https://pris.iaea.org/PRIS/home.aspx, 2020. Accessed 14.5.2020.

[6] INTERNATIONAL ATOMIC ENERGY AGENCY. *Safe Long Term Operation of Nuclear Power Plants.* Number 57 in Safety Reports Series. 2008.

[7] Martti Antila and Tuukka Lahtinen. Recent Core Design and Operating Experience in Loviisa NPP. *Fortum Nuclear Services Ltd, Espoo, Finland.* PDF. Accessed 14.5.2020.

[8] Z. Hozer, K. Trambauer, and J. Duspiva. VVER-specific features regarding core degradation - Status Report (NEA-CSNI-R–1998-20), 1999.

[9] The U.S. Nuclear Regulatory Commission (NRC). Pressurized Water Reactors. /https://www.nrc.gov/reactors/pwrs.html, Last updated January 15, 2015. Accessed 13.5.2020.

[10] *Integrity of Reactor Pressure Vessels in Nuclear Power Plants: Assessment of Irradiation Embrittlement Effects in Reactor Pressure Vessel Steels.* Number NP-T-3.11 in Nuclear Energy Series. INTERNATIONAL ATOMIC ENERGY AGENCY, Vienna, 2009.

[11] STUK. An assessment by the Radiation and Nuclear Safety Authority on the periodic safety review of Loviisa NPP. *Safety assessment PSR2015*, 2017.

[12] E. De Sanctis, Stefano Monti, and Marco Ripani. *Energy From Nuclear Fission: An Introduction.* Springer, 2016.

[13] Juhani Hyvärinen. LUT Lecture notes. Ydinturvallisuus- ja ydinjätehuoltokurssi: Osa 3 Ydinreaktorien toimintaperiaatteet, luento 3.1 Reaktorifysiikka ja lämmönsiirto reaktorin sydämessä, 23.10.2018. Notes are in Finnish.

[14] Theresa Busch. IAEA Department of Nuclear Sciences and Applications. Tackling the Challenges of Nuclear Data in the Future. /https://www.iaea.org/newscenter/news/tackling-the-challenges-of-nuclear-data-in-the-future, 2017. Accessed 14.7.2020.

[15] STUK. Guide YVL E.4 Strength analyses of nuclear power plant pressure equipment, 2020. Translation, original text in Finnish.

[16] S.E. Hughes. *A Quick Guide to Welding and Weld Inspection.* Quick Guides (ASME Press) Series. Matthews Engineering Training, 2009.

[17] ISO 148-1:2016 Metallic materials – Charpy pendulum impact test - Part 1: Test method.

[18] G. S Was. *Fundamentals of radiation materials science: metals and alloys.* Springer, 2017.

[19] Yun Lin, Wen Yang, Zhenfeng Tong, Changyi Zhang, and Guangsheng Ning. Charpy impact test on A508-3 steel after neutron irradiation. *Engineering Failure Analysis*, 82:733 – 740, 2017.

[20] Von Helmut Föll and Bernd Kolbesen. Agglomerate von Zwischengitteratomen (Swirl-Defekte) in Silizium - ihre Bedeutung für Grundlagenforschung und Technologie. *Jahrbuch der Akademie der Wissenschaften in Göttingen*, 1976. Träger des Akademie-Preises für Physik 1976.

[21] G.R. Odette and G. Lucas. Embrittlement of nuclear reactor pressure vessels. *JOM*, 53:18–22, 01 2001.

[22] W.J. Phythian and C.A. English. Microstructural evolution in reactor pressure vessel steels. *Journal of Nuclear Materials*, 205:162 – 177, 1993.

[23] G. R. Odette and G. E. Lucas. Recent progress in understanding reactor pressure vessel steel embrittlement. *Radiation Effects and Defects in Solids*, 144(1-4):189–231, 1998.

[24] Frank Garner. Void swelling and irradiation creep of austenitic stainless steels and its consequences on performance of stainless steels in nuclear reactors. 04 2007.

[25] Amir D Amayev, Alexander M Kryukov, and Mikhail A Sokolov. Recovery of the transition temperature of irradiated WWER-440 vessel metal by annealing. In *Radiation Embrittlement of Nuclear Reactor Pressure Vessel Steels: An International Review (Fourth Volume)*. ASTM International, 1993.

[26] Ralf Ahlstrand and Jyrki Kohopää. Annealing of the RPV of unit 1 in Loviisa 1996. *IVO Power Engineering LTD*, 01019 IVO8, 1997.

[27] I. Uytdenhouwen, R. Chaouadi, I. Szenthe, and Gillemot. F. NOMAD: Non-Destructive Evaluation (NDE) System for the Inspection of Operation-Induced Material Degradation in Nuclear Power Plants Deliverable D1.2 Progress report on results of the reference materials and preparation irradiated materials. 2019. Source derived from the Intranet of the NOMAD project. Unpublished.

[28] SCK CEN Belgian Nuclear Research Center. Belgian Reactor 2. /https://science.sckcen.be/en/Institutes/NMS/BR2, 2020. Accessed 27.5.2020.

[29] C.-G Stefanita. Barkhausen noise as a magnetic nondestructive testing technique. *From Bulk to Nano: The Many Sides of Magnetism*, pages 19–40, 01 2008.

[30] J. R. Hook and H. E. Hall. *Solid State Physics*. Second edition. Springer New York Inc., Chichester, West Sussex, England ; New York: J. Wiley & Sons, 1991.

[31] M. Rabung, M. Kopp, M. Kurras, P. Lappalainen, E. Leskelä, J. Rinta-aho, A. Gasparics, G. Vertesy, M. Niffenegger, H. Kottmann, J. Mathew, and J. Griffin. NOMAD: Non-Destructive Evaluation (NDE) System for the Inspection of Operation-Induced Material Degradation in Nuclear Power Plants Deliverable D2.1 Progress report regarding NDE results on Charpy samples, 2019. Source derived from the Intranet of the NOMAD project. Unpublished.

[32] Murat Deveci. STRESSTECH BULLETIN 2 The Properties of Barkhausen Noise. /https://www.stresstech.com/stresstech-bulletin-2-the-properities-of-barkhausen-noise/. Accessed 20.7.2020.

[33] Jari Rinta-aho and Ari Koskinen. NOMAD: Non-Destructive Evaluation (NDE) System for the Inspection of Operation-Induced Material Degradation in Nuclear Power Plants Deliverable D3.1 Report on method classification and database, 2019. Source derived from the Intranet of the NOMAD project. Unpublished.

[34] Bernd Wolter, Yasmine Gabi, and Christian Conrad. Nondestructive Testing with 3MA – An Overview of Principles and Applications. *Applied Sciences*, 9:1068, 03 2019.

[35] Gerd Dobmann. Physical basics and industrial applications of 3MA – micromagnetic multiparameter microstructure and stress analysis. *Fraunhofer IZFP, Saarbrcken, Germany*, pages 1–17, 2007.

[36] Samfaß, Lisa and Baak, Nikolas and Meya, Rickmer and Hering, Oliver and Tekkaya, A Erman and Walther, Frank. Micro-magnetic damage characterization of bent and cold forged parts. *Production Engineering*, 14(1):77–85, 2020.

[37] Sargon Youssef and Cyril Zimmer and Klaus Szielasko and Andreas Schütze. Bewertung subjektiver und automatisierter Merkmalsextraktion periodischer Zeitsignale am Beispiel des 3MA-X8-Verfahrens. *tm - Technisches Messen*, 86(5):267 – 277, 2019.

[38] Ivan Tomáš and Gábor Vértesy. Magnetic adaptive testing. *Nondestructive testing methods and new applications*, (3):145–186, 2012.

[39] G. Vértesy, A. Gasparics, I. Szenthe, F. Gillemot, and I. Uytdenhouwen. Inspection of Reactor Steel Degradation by Magnetic Adaptive Testing. *Materials (Basel)*, 2019. 12(6);963.

[40] Jari Rinta-aho, Petteri Lappalainen, and Esa Leskelä. Non-destructive evaluation of material degradation of nuclear reactor pressure vessels. In *Baltica XI*, Finland, 2019. VTT Technical Research Centre of Finland. Baltica XI 2019 : International Conference on Life Management and Maintenance for Power Plants ; Conference date: 11-06-2019 Through 13-06-2019.

[41] M. Niffenegger and H.J. Leber. Monitoring the embrittlement of reactor pressure vessel steels by using the Seebeck coefficient. *Journal of Nuclear Materials*, 389(1):62 – 67, 2009. Thermochemistry and Thermophysics of Nuclear Materials.

[42] Nathan Ida and Norbert Meyendorf. *Handbook of Advanced Nondestructive Evaluation*. Springer, Cham, 01 2019.

[43] Lester Schmerr and Jung-Sin Song. *Ultrasonic nondestructive evaluation systems*. Springer, 2007.

[44] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning, Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[45] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning: With Applications in R*. Springer Publishing Company, Incorporated, 2014.

[46] Ron Kohavi. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'95, pages 1137–1143, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[47] Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[48] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[49] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, 67(2):301–320, 2005.

[50] Sebastian Ruder. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747, 2016.

[51] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization, 2014.

[52] Corinna Cortes and Vladimir Vapnik. Support-Vector Networks. In *Machine Learning*, pages 273–297, 1995.

[53] Harris Drucker, Christopher JC Burges, Linda Kaufman, Alex J Smola, and Vladimir Vapnik. Support vector regression machines. In *Advances in neural information processing systems*, pages 155–161, 1997.

[54] Bernhard Schölkopf and Alexander Smola. *Learning with Kernels*. 2002.

[55] Alex J. Smola and Bernhard Schölkopf. A tutorial on support vector regression, 2004.

[56] Gregory Shakhnarovich, Trevor Darrell, and Piotr Indyk. Nearest-Neighbor Methods in Learning and Vision: Theory and Practice (Neural Information Processing). 2005.

[57] Olga G. Troyanskaya, Michael N. Cantor, Gavin Sherlock, Patrick O. Brown, Trevor J. Hastie, Robert Tibshirani, David Botstein, and Russ B. Altman. Missing value estimation methods for DNA microarrays. *Bioinformatics*, 17 6:520–5, 2001.

[58] Leo Breiman. Bagging Predictors. *Mach. Learn.*, 24(2):123–140, August 1996.

[59] Harris Drucker. Improving Regressors Using Boosting Techniques. *Proceedings of the 14th International Conference on Machine Learning*, 08 1997.

[60] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA, 1984.

[61] Jan Struyf and Sašo Džeroski. Constraint Based Induction of Multi-objective Regression Trees. In Francesco Bonchi and Jean-François Boulicaut, editors, *Knowledge Discovery in Inductive Databases*, pages 222–233, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[62] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE, 1995.

[63] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely Randomized Trees. *Mach. Learn.*, 63(1):3–42, April 2006.

[64] Gilles Louppe. Understanding Random Forests: From Theory to Practice, 2014.

[65] F. Rosenblatt. The Perceptron: A Probabilistic Model for Information Storage and Organization in The Brain. *Psychological Review*, pages 65–386, 1958.

[66] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. /http://www.deeplearningbook.org.

[67] Alex Lenail. NN-SVG: Publication-ready NN-architecture schematics. /http://alexlenail.me/NN-SVG/. Accessed 6.5.2020.

[68] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[69] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. 15(1):1929–1958, 2014.

[70] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. In *Advances in neural information processing systems*, pages 971–980, 2017.

[71] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[72] Martín Abadi et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, 2015. Software available from tensorflow.org.

[73] François Chollet et al. Keras. /https://keras.io, 2015.