UCLouvain
Institute for Information
and Communication Technologies,
Electronics and Applied Mathematics

Data-driven learning and optimization approaches for proton therapy

Valentin Hamaide

Thesis submitted in partial fulfillment
of the requirements for the degree of
Docteur en sciences de l'ingénieur

Dissertation committee:

Prof. François Glineur (UCLouvain, advisor)
Prof. Benoît Macq (UCLouvain, advisor)
Prof. John Lee (UCLouvain)
Prof. Jean-Philippe Thiran (EPFL, Switzerland)
Dr. Guillaume Janssens (Ion Beam Applications, Belgium)
Prof. Christophe Craeye (UCLouvain, chair)

October 2022.

Abstract

Proton therapy is a type of radiation therapy that uses a beam of protons to irradiate cancerous tissues. In principle, it offers a physical advantage over conventional radiotherapy due to the very localized dose deposition of protons in the body, which can be exploited to decrease the dose received by healthy tissues, leading to fewer complications. However, this unique characteristic comes at the cost of high vulnerability to uncertainties, requiring extremely precise machinery from beam production to treatment delivery. Moreover, accelerating protons requires a heavier infrastructure than conventional radiotherapy, which increases the cost of proton therapy over photon therapy.

This thesis aims at improving the accessibility, cost-effectiveness, and treatment quality of proton therapy by using data-driven approaches wherever they can bring added value.

In the first part of this work, with the aim of improving equipment maintenance, we develop predictive maintenance solutions based on machine learning to detect incoming failures and decrease the overall system downtime. In the second part of the thesis, we concentrate on reducing the time needed to install a new proton therapy system by developing an automatic procedure based on mathematical optimization to speed up the calibration of a proton therapy beamline. The third and final part of the thesis is devoted to improving the treatment quality of mobile tumors in proton therapy. To achieve this goal, we develop a method based on a library of treatment plans that uses real-time information about the patient's anatomy via the acquisition of images to guide the treatment delivery.

Acknowledgments

First of all, I would like to sincerely thank my two advisors, Prof. François Glineur and Benoît Macq. François Glineur gave me the opportunity to do the PhD that I aspire to: an applied thesis in partnership with an industry that can have a direct impact on society. Throughout the thesis, François was always there to support me, give me feedback, sharing ideas and advice that helped me to arrive at this point. Thanks to his expertise, I gained valuable knowledge in a variety of domains. When my first project came to an end, I reached out to Benoît in pursuit of funding that would continue the collaboration with IBA I started in the previous project. I am very thankful to him for accepting and proposing an extremely interesting topic on mobile tumors that has the potential to have an important impact on treating cancer. Thanks to this new funding, I got to work with a team of very nice and bright people.

I would like to thank my thesis jury composed of Prof. John Lee, Prof. Jean-Phillipe Thiran, Prof. Christophe Craeye, and Dr. Guillaume Janssens. They gave me valuable feedback and we shared very interesting discussions during my defense that helped improve this manuscript. I would also like to thank Prof. Raphaël Jungers for being part of my accompanying committee and giving me feedback on the second part of the thesis.

I would like to thank my colleagues at IBA, with whom I worked throughout the thesis. During my first project, I want to especially thank Lauriane Castin, who welcomed me at IBA, helped me feel integrated into the team, and worked towards integrating my work within the IBA system; but also Kurt Verduyck for the trust he gave me, Flavien for his immense knowledge of data science tools, Corentin, Dat, Pierre-Yves, Martin, Gery, Marc,

and all the others. I also thank Denis Joassin, who worked with us on the predictive maintenance project and with whom I learned a lot. I thank Quentin and Félix for the work we achieved during my second project. Finally, during my last project with IBA, I thank again Guillaume Janssens, for his incredibly useful feedback.

During my time at UCLouvain, I had the chance to meet and work with a lot of great colleagues. Thanks to Ange, Antoine, Benjamin, Cécile, Charles, Damien, Dani, Emilie, Estelle, Félicien, Gauthier, Guillaume, Jonathan, Karim, Kevin, Loïc, Mariana, Maxime, Nicolas, Victorien, Zheming. A special thanks to Kevin Souris, who very much helped me on my last project, and to Antoine and Damien for proofreading part of my manuscript.

My deepest thanks go to my family. Many thanks to my dad, without whom I probably would not have started a PhD and for proofreading this manuscript, and to my mom for always believing in me. Finally, I would like to thank my love, Elénore, for always supporting me.

Funding and context of the thesis

This work was supported by the Walloon region under two research projects. The purpose of the first research project, called BidMed (Big Data in Medicine), was to apply *Big Data* technologies in the healthcare sector, with the objective to improve proton therapy accessibility, for which IBA (Ion Beam Applications, Louvain-la-Neuve) is the renowned world leader. To this end, the project was a collaboration between IBA and UCLouvain. The BidMed project aimed at improving the performance and reducing the costs associated with proton therapy at all stages of the equipment life cycle. It tackled three sub-projects, two of which we worked on in parts 1 and 2 of the thesis, namely automating the system calibration with data-driven approaches and using statistical analysis of equipment monitoring data and predictive analysis to enable predictive interventions.

The purpose of the second research project, called ARIES, was to improve the treatment of mobile tumors in proton therapy by optimizing the acquisition of images, estimating the tumor movement, and supervising the treatment delivery in real time. In the third part of the thesis, we tackled the real-time treatment delivery part of this project, which was also in close collaboration with IBA.

Pour Gauthier.

Contents

II Calibration of a proton therapy beamline using derivative-free optimization

III Treatment of mobile tumors in proton therapy with a library of treatment plans

List of acronyms

BO Bayesian Optimization

CT Computed Tomography

CTV Clinical Target Volume

DVH Dose-volume Histogram

GP Gaussian Process

GTV Gross Tumor Volume

HI Health Indicator

HU Hounsfield Unit

IMPT Intensity Modulated Proton Therapy

IMRT Intensity Modulated Radiotherapy

ITV Internal Target Volume

LCB Lower Confidence Bound

LINAC Linear Accelerator

MFO Multiple Fields Optimization

MI Mutual Information

MRI Magnetic Resonance Imaging

OAR Organ-at-risk

PBS Pencil Beam Scanning

PdM Predictive Maintenance

PT Proton Therapy

PTV Planning Target Volume

RF Radio Frequency

RT Radiotherapy

RUL Remaining Useful Life

SFO Single Field Optimization

SOBP Spread-out Bragg Peak

VMAT Volumetric Arc Radiotherapy

VOI Volume of Interest

1

Introduction

1.1 Some facts about cancer

Cancer refers to a group of diseases caused by the abnormal proliferation of cells that eventually form tumors, which can invade and spread to any part of the body. Cancer is the second cause of death worldwide after cardiovascular diseases, accounting for nearly 10 million deaths in 2020 [FEL+20]. The most common cancers are breast (11.7%), lung (11.4%), colorectum (10%), and prostate (7.3%) cancers, while the most fatal ones are lung (18%), colorectum (9.4%), liver (8.3%) and breast (7.7%) cancers.

Each cancer type requires a specific treatment that depends on various factors such as its location and stage. Common treatments include surgery, radiation therapy, and/or systemic therapies (chemotherapy, immunotherapy, or targeted therapy) [can22, WHO22]. Surgery and radiation therapy are *local* treatments that are used to target a specific part of the body, while drug treatments are *systemic* and affect the entire body. This thesis focuses on radiation therapy and, in particular, proton therapy, a special type of radiation therapy.

1.2 Radiation therapy

Radiation therapy (RT) is one of the most common cancer treatments and is involved in about half of the cases [BLYY12], possibly in conjunction with other types of treatments. It uses ionizing radiations such as X-rays, gamma rays, electron beams, or proton beams to destroy or damage cancer cells. Ionizing radiations are radiations that carry enough energy to ionize matter and damage the DNA of cells, potentially leading to cellular death. The energy deposited in tissues during those interactions is called the *absorbed dose* and is expressed in the unit of Gray (Gy), which is the energy (Joule) absorbed per unit of mass (kg), i.e. $1Gy = 1J/kg$. Radiations can damage both cancerous and healthy tissues. However, healthy tissues have a better repair mechanism than tumor cells [BLYY12]. Nevertheless, high-dose radiation can also lead to healthy cell death; that is why the goal of radiotherapy is to maximize the dose delivered to cancerous tissues while minimizing the exposure of healthy tissues.

Radiation therapy can be administered to patients in three ways:

External beam RT: uses a machine that steers high-energy rays from outside the body into the tumor.

Brachytherapy: also called internal radiation, uses a radioactive source put inside the body at the tumor location.

Systemic radiation: uses a radioactive drug given by infusion or oral ingestion that travels through the body, locating and killing tumor cells.

This thesis will focus on external beam therapy, the most common type of radiation therapy. Usually, the patient is positioned on a couch, and an external radiation source is directed towards the body region to be treated. In external beam therapy, as in other types of radiotherapy, completely avoiding healthy tissues is unfortunately not possible for various reasons. This is because the beam must penetrate healthy tissues to reach the tumor, and on the other hand, due to the uncertainties linked to delivering the dose as planned and the delineation of the target volume by oncologists. Treatment planning deals with this by using physical and mathematical tools to optimize a trade-off between a high and conformal dose to the tumor and a low dose to organs-at-risks (OARs). Many degrees of freedom are possible to deliver radiation, such as the amount of radiation dose, the number of beam directions, the spatial distribution of the dose, and the radiation

modality. In most cases, X-rays (equivalently called photon beam) are the ionizing radiation type used in radiation therapy, which is produced via a linear accelerator (linac). The electron beam produced by the linac can also directly be used to treat superficial tumors (within 6 cm of the patient's surface) [HA06] instead of its usual purpose of being used to bombard a heavy metal target to generate a photon beam. Radiation therapy involving X-rays (or photons) is generally referred to as radiotherapy, X-ray therapy, or photon therapy.

Another radiation modality is *hadron therapy*, or *particle therapy*, which uses a beam of protons or heavier ions to treat cancerous tissues. Proton therapy is by far the most widely used particle therapy, which accounted for 86% of these treatments according to a 2014 research study [Jer15] while carbon ions accounted for the remaining 14%, and other marginal treatment methods are only experimental. However, proton therapy represents less than 1% of radiotherapy treatments today.

1.3 Proton therapy

Proton therapy consists in using high-energy protons to irradiate cancerous tissues. The key difference with photon therapy lies in the way protons deposit their dose in tissues. In Figure 1.1, we compare the depth-dose deposition of protons and photons. A photon beam delivers the highest dose at a depth of 1-3cm and then slowly decreases with depth. In contrast, protons show a small dose deposition increasing with depth, a sharp increase at a certain depth followed by an extremely sharp fall off to zero. This is because proton dose deposition is inversely proportional to the velocity of protons. Protons lose energy and slow down as they traverse tissues due to atomic and nuclear interactions. As they slow down, more and more interactions with orbiting electrons occur, causing a peak energy release at the end of their range, called the *Bragg peak*. This ideally results in a low dose deposition in front of the tumor, a high dose deposition in the tumor, and a minimal dose after the tumor. Thus, protons are physically more advantageous than photons due to their depth-dose profile [Jäk09]. The depth of the Bragg peak depends on the proton beam's energy. To cover the entire tumor, proton beams of different energies must be superposed by either *passive scattering* or *spot scanning* techniques, resulting in the so-called *spread-out Bragg peak* (SOBP). Both of these techniques result in a substantially higher dose deposition in front of the tumor and a simi-

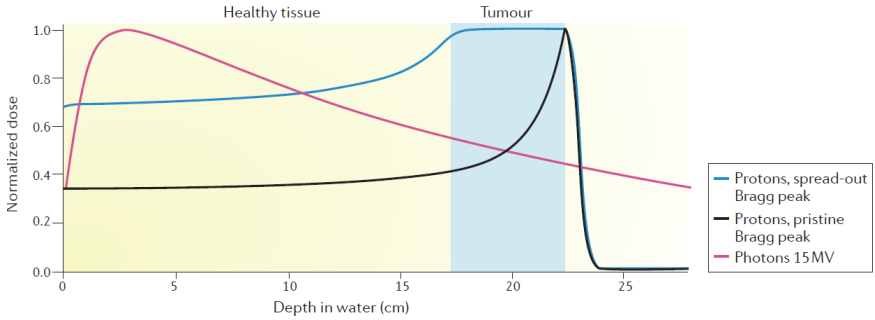


Fig. 1.1 Dose deposition in tissues as a function of depth (measured in water equivalent distance as the mean density of tissues in patients). The superposition of proton beams of different energies leads to the so-called spread-out Brag peak (SOBP) covering the entire tumor. Image taken from [BKO+16] with permission.

lar fall-off after it. Nevertheless, the global depth-dose profile still remains more advantageous than for photons.

The physical limitation of photons can be improved by using multiple beams. In fact, today's state-of-the-art photon therapy uses a technique called *volumetric modulated arc radio therapy* (VMAT) that uses a continuous distribution of beams across a 360-degree arc while modulating the intensity in each field [Ott08]. In clinical practice, this leads to a much higher dose conformity in the target than what is suggested in Figure 1.1, which is comparable to what a proton treatment plan can achieve [BKO+16]. Nevertheless, protons still have three advantages from a physics point of view. First, they provide a more conformal dose to the target with the same number of beams [PJ07]. Second, the dose deposited in surrounding organs is still substantially lower than in photon plans, which in principle leads to less toxicity in normal tissues and decreases the risk of inducing secondary cancer [PAM+12]. Third, in cases where the tumor is very close to a critical organ such as the spinal cord or the heart, proton therapy may be able to give the necessary curative radiation dose to the target while remaining within the surrounding tissue-dose constraints, while photon therapy would be unable to do so [BKO+16].

Despite the physical advantage of protons, their precision comes at the cost of high vulnerability to uncertainties. In particular, the uncertainty of the proton range, which is determined by the position of the Bragg peak in

the body, can yield to a deterioration of the treatment quality. This Bragg peak position depends on the energy of the proton and the tissue density and composition across its path [Pag18]. Should the tissue density change during treatment, e.g. due to weight gain/loss, tumor shrinkage, tumor displacement, or a specific organ characteristic (e.g. empty or full bladder), the Bragg peak is shifted to a position different than originally planned. This can lead to missing the target and an unwanted high dose in normal tissues, leading to an overall deterioration of the optimized dose distribution. Similarly, breathing-induced tumor motion in thoracic cancers can also lead to a shift in the expected proton range. Uncertainties on the tumor position and tissue densities also arise in photon therapy, but increasing the target volume with safety margins around the tumors allows to mitigate the dose deterioration, as we can observe in an example of tumor displacement in Figure 1.2. Indeed, in photon therapy, we can simply add margins around the tumor to encompass possible shifts in the tumor positions to deal with the uncertainties. However, in proton therapy, the simple margin approach is sometimes not sufficient, and more complex techniques are required, namely, robust optimization.

Today, robust optimization in proton therapy effectively takes into account range errors, set-up errors, and even anatomical changes due to organ motion. Robust optimization has been applied clinically and shows good results under multiple error scenarios for many types of cancer locations, e.g. head and neck cancers [LFL+13, vDStH+16], lung cancers [LZP+15, LSC+16], base-of-skull cancers [LMP+14], liver cancers [PKB+19], etc.

Despite the physical superiority of protons over photons, the evidence of their clinical superiority is still mixed. It is commonly accepted that proton therapy is safe, effective, and recommended for pediatric cancers, ocular melanomas, chordomas, and chondrosarcomas [MG17]. However, there is a lack of evidence or only evidence based on studies involving a small number of patients for many other types of cancers [DRLJ+12, MZ14, MAB+17, Oh19]. Due to the limited clinical evidence, there have been concerns about the high cost of proton therapy compared to conventional radiotherapy [YZQ19]. The consensus is that there is a need to conduct more randomized trials and investigate larger patient cohorts on longer follow-up periods from multiple institutions to demonstrate the clear advantage of protons [MG17]. However, although randomized controlled trials are the gold standard for medical evidence, they may be inappropri-

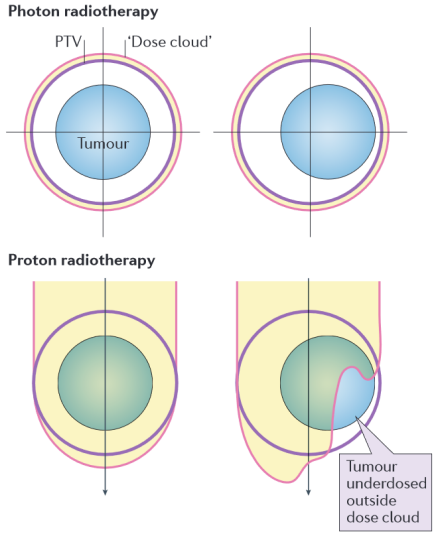


Fig. 1.2 Consequences of set-up or motion uncertainties in photon and proton therapy. Adding planning target volume (PTV) margins around the tumor allows guaranteeing coverage in photon therapy even in case of density variations, while it is not always the case for proton therapy. Image taken from [BKO+16] with permission.

ate for proton therapy and even unethical for cases with clear dosimetric benefits (e.g. dosimetric advantage of proton therapy in pediatric cancers) [MZ14]. That is why Dutch scientists and officials have proposed an approach based on normal tissue complication probability (NTCP) models to select patients who are most likely to experience fewer (serious) adverse effects achievable by state-of-the-art proton therapy treatments and validate and conduct randomized trials based on those NTCP models in appropriately composed cohorts [WVDSL+16]. Validating those models should be the primary focus of research to obtain medical evidence, according to the authors, which is still an ongoing research.

The most important drawback of proton therapy remains its cost. A single-room facility costs in the range of $30 million, while a multi-room PT facility consisting of 3 to 5 rooms costs well over $100 million (up to $300 million for the proton therapy center in Harlem, New York [Han18]). That is an order of magnitude higher than the cost of a high-end photon therapy unit [MG17]. This is due to the vast infrastructure needed: a par-

ticle accelerator such as a cyclotron or synchrotron to accelerate protons to 60% of the speed of light, rotative gantries with wheels of typically 10 meters weighing 100-200 tons to direct the beam of protons from a wide range of angles and a lot of concrete (meters thick) to shield patients from neutrons produced by the system [BL17]. Despite its huge cost, proton therapy remains more cost-effective for certain tumors than conventional radiotherapy. For instance, proton therapy has been proven more cost-effective for pediatric brain tumors, well-selected breast cancers, locoregionally advanced NSCLC, and high-risk head/neck cancers [VMM16]. The cost-effectiveness is computed via models including the cost of intervention but also all aftereffects such as the probabilities of adverse effects with corresponding medical costs, treatment complications, length of hospital stay, disease course, etc. Nevertheless, proton therapy remains expensive, resulting in a high barrier to widespread adoption. Increasing cost-effectiveness is therefore crucial for the field and the accessibility to the best health care possible.

In the next sections, we detail the main parts of a proton therapy system, from producing the beam of high-energy protons up to treatment delivery.

1.3.1 Proton therapy system

An example of a proton therapy system is represented in Figure 1.3 and is composed of four main parts. The first part is the cyclotron, used to accelerate protons to high energies, then a beamline composed of electromagnets is used to transport the beam of protons to the treatment room. In the treatment room, a mechanical gantry allows rotation around the patient, and finally, the beam is delivered to the patient through a nozzle. A PT system can be single room or multi rooms, in which case the beamline extends to other treatment rooms and gantries.

1.3.2 Beam production

Protons lose energy when interacting with matter and stop at a certain range that depends on their initial energy. Proton therapy systems must accelerate protons to energies up to 230-250 MeV[1] to attain a proton mean range of 33-38 cm in water [Pag18]. Since the human body is mainly com-

[1]The electron volt (eV) is the standard unit of energy used in particle physics. It represents the amount of energy gained by accelerating a single electron (or any particle that has a charge equivalent to the electron) to a potential difference of 1 volt.

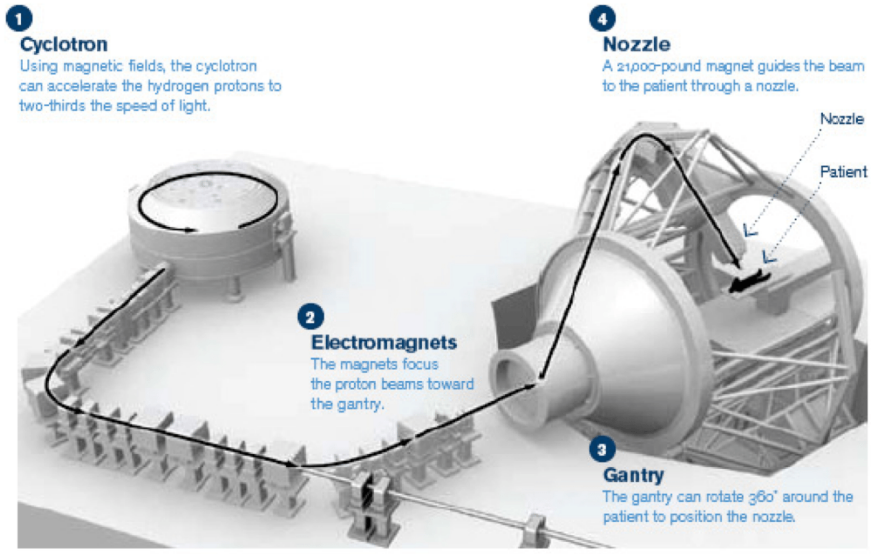
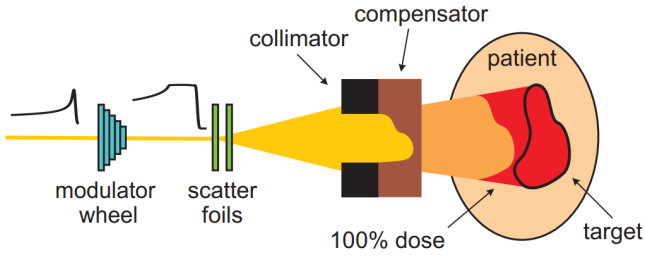


Fig. 1.3 Proton therapy system architecture. *Credits: The New York Times and University of Florida Proton Therapy Institute*

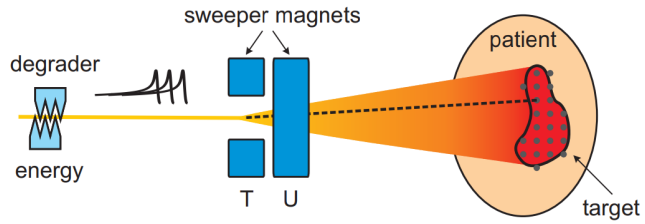
posed of water, this distance is a good approximation of the proton range in the body. Moreover, 30-40cm is sufficient to treat the vast majority of tumors.

Those energies are much higher than what is needed for conventional photon therapy. Hence, the linear accelerator used to accelerate electrons in photon therapy is not sufficient for the application in proton therapy[2]. That is why circular accelerators are used, since they are capable of accelerating protons to those necessary energies. Accelerators currently used in proton therapy are isochronous cyclotrons, synchrocyclotrons, and synchrotrons, each having their own advantages and disadvantages. For a detailed overview of proton therapy accelerators, we invite the reader to refer to [OLJ16] and [Pag18, Chapter 3].

[2]Even though the linear accelerator was not considered a viable alternative for proton therapy until recently, a proton therapy LINAC is currently under study [DSUS16], although it is much bigger than medical LINACs used in photon therapy.



(a) Passive scattering.



(b) Spot scanning

Fig. 1.4 Beam delivery techniques in proton therapy. Image taken from [Zen10]

1.3.3 Beam transport

For radioprotection-related reasons, the particle accelerator is usually not located in the treatment room. Therefore, the beam of protons is transported in a vacuum tube from the particle accelerator to the treatment room via a *beamline* composed of electromagnets that guide the beam to the treatment room but also shape the beam to respect certain constraints.

1.3.4 Beam delivery

There are two techniques to deliver a proton therapy treatment: passive scattering and active spot scanning. A diagram of those two delivery techniques is provided in Figure 1.4.

In *passive scattering*, the beam is spread by scatterers while a modulator wheel and range compensator are used to shape the dose distribution in-depth, producing the so-called spread-out Bragg peak. The lateral dose distribution is shaped by a collimator which stops protons from going outside the tumor. In *active spot scanning*, also called *pencil beam scanning* (PBS), a thin beam of protons is steered laterally via scanning magnets that deflect

the beam in discrete spot positions within the tumor. Spot scanning is delivered in successive *layers* of energy (of possibly different lateral shapes), which correspond to delivering the dose to "slices" of the tumor, thus shaping the dose distribution in depth. While passive scattering was the first treatment modality in proton therapy, the current state-of-the-art approach is PBS which allows better dose conformity and better flexibility (no specific compensators or aperture needed).

1.4 Application of data-driven approaches to proton therapy

As mentioned earlier, increasing the cost-effectiveness of proton therapy is crucial for its widespread adoption. Mathematical models can help reduce some of those costs. For instance, predictive maintenance models, based on sensor data collected in specific parts of the systems, can help make decisions about the right timing for replacing a piece of equipment to avoid unexpected failures, or to increase its total useful life by deciding to replace the equipment later than initially foreseen; hence this may contribute to substantially reducing costs. We investigate the application of predictive maintenance to one of the components of a PT cyclotron. This is the first contribution of this thesis.

Mathematical models can also be used to gain time and therefore decrease labor costs. We can build and use such models to automatically calibrate parts of the system in order to reduce the time usually needed to install a new PT system, which generally can take several months. We investigate the application of specialized optimization methods to automatically calibrate a PT beamline. This is the second contribution of this thesis.

Another contribution to the field of proton therapy that can be made with mathematical and data-driven models is to improve the treatment quality and broaden the number and types of cancers that could benefit from the dosimetric advantages of proton therapy. This thesis investigates possible improvements for treating moving tumors, i.e. thoracic tumors that move due to the patient breathing. This is the third contribution of this thesis. Today, treating mobile tumors in proton therapy is quite challenging due to proton range variation with motion discussed in Section 1.3. Thus, many facilities choose not to pursue treatment of those tumors with proton therapy if the breathing motion is too high. Some techniques exist to mitigate motion, but there is still much room for improvement.

1.5 Thesis outline

This thesis is divided into three parts. Each part contains a background chapter that lays out the theory and state-of-the-art methods in the subject, followed by one or two chapters corresponding to contributions we bring to the field and the application of the developed methods to proton therapy. The contributions chapters are based on peer-reviewed articles that are either published or submitted. A list of publications is given at the end of the manuscript.

The first part is about predictive maintenance, where the goal is to predict the upcoming failure of a component or system via data-driven methods. Chapter 2 introduces the field of predictive maintenance and the currently existing methods. Chapter 3 develops a method for selecting a set of *features* that have high predictive power. The results were published in a scientific article [HG21b]. Chapter 4 is a comparative study between different learning formulations for tackling a predictive maintenance problem via a two-level approach, one dedicated to learning and the other to decision making. It is adapted from a submitted article [HJCG22] that builds upon a conference paper [HG19]. Chapter 3 and 4 are both applied to the prediction of incoming failure of a component of the synchrocyclotron used in IBA[3]'s proton therapy system.

The second part of the thesis is about calibrating a proton therapy beamline via *derivative-free optimization*. Chapter 5 reviews common derivative-free methods, also called *black-box optimization*, which are optimization methods that do not require the computation of the gradient of the objective function. In particular, the Nelder-Mead algorithm and Bayesian optimization are reviewed in detail. Chapter 6 applies the methods reviewed in Chapter 5 to the calibration of IBA's proton therapy beamline. We also develop a novel transfer learning approach to reuse data from previous beamline configurations for subsequent optimizations, which led to a publication [HG21a].

The third and last part of the thesis is about treating moving tumors in proton therapy. Chapter 7 gives some background on treatment planning and delivery in proton therapy as well as specific motion monitoring and

[3]Ion beam applications (IBA) is a company based in Louvain-la-Neuve, Belgium, specialized in the design and manufacturing of proton therapy systems.

mitigation strategies currently in use. In Chapter 8, we propose a real-time image-guided approach based on a library of treatment plans to treat mobile tumors. This chapter is adapted from an article submitted for publication [HSD+22].

Chapter 9 concludes the thesis with a review of the contributions to the field of proton therapy and future perspectives.

PART I
Predictive maintenance: application to a cyclotron component

2

Background in predictive maintenance

Predictive maintenance (PdM), also called *condition-based maintenance* or *prognostic health management*, consists in recommending maintenance decisions based on the information collected through condition monitoring, usually in the form of time series. It is generally formulated in one of the two following ways: i) detecting that the machine under monitoring has entered a faulty state, therefore predicting that a failure is coming, or ii) predicting the remaining useful life (RUL) of the machine. In the scientific literature, those two approaches are referred to as *diagnostics* and *prognostics* respectively. *Prognostics* is defined by Jardine et al. [JLB06] in their review as a way "to predict faults or failures before they occur" and *Diagnostics* as "focusing on detection, isolation, and identification of faults when they occur" or as "a procedure of mapping the information obtained in the measurement/features space to machine faults in the fault space", i.e. pattern recognition. Usually, the first approach can be a trigger for the second one, but this might not always be the case.

In a recent survey [LLG+18], the authors outline a systematic framework for predictive maintenance based on four steps depicted in Figure 2.1: Data acquisition, Health indicator construction, Health states division, and Remaining useful life prediction. Those steps are detailed below.

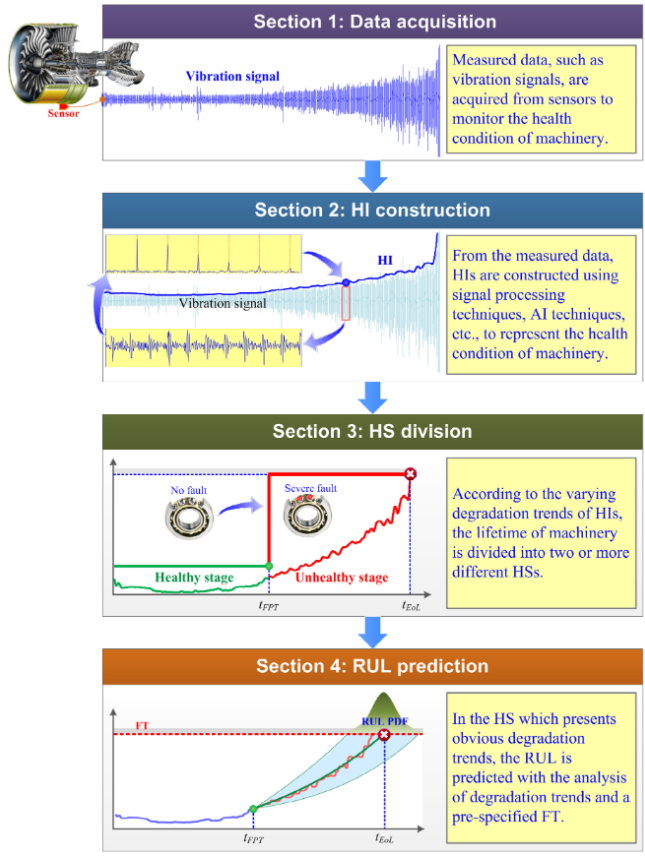


Fig. 2.1 The four steps in predictive maintenance. Image taken from [LLG+18] with permission.

2.1 Data acquisition

Data acquisition refers to the process of acquiring monitoring information from various sensors placed around or inside the equipment of interest. Common sensors used are accelerometers, acoustic emissions sensors, infrared thermometers, current sensors, torque sensors, etc. Today, more and more data are being collected with the emergence of Industry 4.0 and the internet of things (IoT). However, only few, if any, run-to-failure data are generally available, making it difficult to develop and test new models for researchers and engineers [LLG+18]. The reason for the scarcity of run-to-

failure data is due to several factors. First of all, this can be explained by the long degradation process of the monitored equipment, which can take from several weeks or months up to several years, making data collection very time consuming. Secondly, machines are not necessarily allowed to go to complete failure, which prevents capturing the entire lifespan of the equipment and makes a learning algorithm harder to train. Third, very few commercial institutions that collect run-to-failure data are willing to make them accessible because of potential competition or industry secrecy. As a result, very few real datasets are available in the scientific literature. Hence, the literature mainly makes use of datasets acquired from accelerated mechanical degradation instead of real industrial equipment (e.g. bearing datasets in [NGM+12] and [LQY+07]), or even completely virtual datasets for which a degradation is simulated via physical models (e.g. Turbofan engine degradation simulation dataset in [SGSE08]). On top of this run-to-failure data scarcity, the inherent data quality issues (e.g. problems due to interference from the outside world or missing data) observed in many applications make it even more challenging to obtain a good predictive model.

2.2 Health indicator construction

The degree of damage in a machine is something hard to observe or assess directly. For instance, in mechanical systems, directly observing wear areas or crack lengths requires opening the machine, which cannot be done on a regular basis. Even if it was feasible, damages are usually on the order of the microscale and are thus hard to observe. Moreover, it is sometimes impossible to observe such damages for some types of equipment such as rolling bearing without breaking them. Instead, our purpose here is to assess the damage degree of a machine via indirect information from sensor measurements. Not all sensor measurements are relevant, and for those that are, one would also need to distinguish valuable information from regular operations and measurement noise. This is the science of feature extraction and feature selection that is usually referred to as *health indicator construction* in the field of predictive maintenance. A health indicator can have a physical meaning, such as the root mean square (RMS) value in a vibration signal, or no physical meaning at all, such as principal component analysis (PCA) features, or again be the result of the fusion of multiple features. In this thesis, we will use the term *feature* to denote intermediate features that are fed into a model and the term *health indicator* for the out-

put of the model on which a decision is taken. In case the decision is taken on a single feature, we will use the term health indicator for that feature.

Features are extracted from raw sensor measurements using either signal processing methods, statistics, or machine learning. Feature extraction can be further categorized as time-domain, frequency-domain, and time-frequency domain. Time-domain features are directly computed from sensor measurements on user-defined time windows. Some common features are RMS, moving average, higher statistical moments, Crest factor, etc. Frequency domain features are computed based on the Fourier transform of a time-series signal. Common frequency-domain features are amplitudes at specific frequencies, spectral power, spectral kurtosis, etc. Finally, time-frequency domain features analyze a signal in both time and frequency simultaneously, via transformations such as the short-time Fourier transform, wavelet transform, or Hilbert-Huang transform. Machine learning features include the construction of features based on the fusion of features via artificial neural networks, PCA, self-organizing maps, and many other methods. Many reviews exist on the construction of health indicators for predictive maintenance. For vibration data, the reader can refer to [WTM17] for a review of the common features. For standard features in rotary machines, the reader can refer to [ZNSM14]. For a broader range of applications in predictive maintenance, the reader can look into [AMC+20].

Identifying a relevant set of features from the constructed features is called feature selection. There are specialized metrics to quantify the relevance of a prognostic metric that will be discussed in Chapter 3.

2.3 Health state division

Health state division consists in dividing the degradation process of a machine into multiple health states. This is similar to the *fault detection* or *fault diagnostic* terms commonly used within the PdM community. A common health state division is to split a machine into two health states: a healthy and a faulty state. This can be done via a classification algorithm or anomaly detection methods. However, there could be more than two health states if there are several fault patterns or operational conditions. In that case, multi-class classification or clustering algorithms are used. Machine learning techniques applied to fault diagnosis were recently reviewed in [LYJ+20]. Those techniques consist in designing machine learn-

ing models to establish the relationship between selected features and the health state of machines. In [LYJ+20], those methods are described chronologically, starting with traditional machine learning (ML) models such as artificial neural networks (ANN), support vector machines (SVM), and k-nearest neighbors (KNN). At present, deep learning methods such as convolutions neural networks (CNN), auto-encoders (AE), or deep belief networks are more and more used, as they are capable of learning both the task and the feature space simultaneously. However, a common assumption of those methods is the availability of sufficient labeled data, which is usually hard to obtain in real-world engineering scenarios [LYJ+20]. Machine learning is not the only technique used in fault diagnosis. More traditional statistical methods such as hidden Markov models, hypothesis testing, and changepoint detection have also been successfully applied to fault diagnosis [LZLL17, TMMZT11]. Another category of methods is model-based techniques constructed using physical principles or systems identification techniques. The reader can refer to [GCD15a] for a review of those techniques. A final category is hybrid methods which use a combination of several techniques and are reviewed in [GCD15b].

2.4 Remaining useful life prediction

The remaining useful life (RUL) is defined as "the useful life left on an asset at a particular time of operation" [SWHZ11]. There are many definitions of what is regarded as useful life, and the general answer is that it depends on the context. We assume in this thesis that the useful life ends when the machine goes into a failure, i.e. when it cannot perform its task anymore. Thus, RUL prediction aims at predicting at each time step the number of hours/days/cycles remaining before the machine goes into failure or the machine cannot properly perform its task. Mathematically, the RUL can be expressed as

$$\mathrm{RUL}_t = \inf(r_t : X(t + r_t) \geq \gamma) \tag{2.1}$$

where inf represents the infimum, $X(t + r_t)$ represents the value of the health state at time $t + r_t$ and γ is a user-defined threshold. As in Section 2.3, different categories of approaches can be used to solve this problem. Figure 2.2 gives an overview of the categories and methods used in the scientific literature to predict the RUL. Statistical methods were the predominant approaches in 2018, although AI approaches are more and more studied. Moreover, deep learning methods, not properly represented in

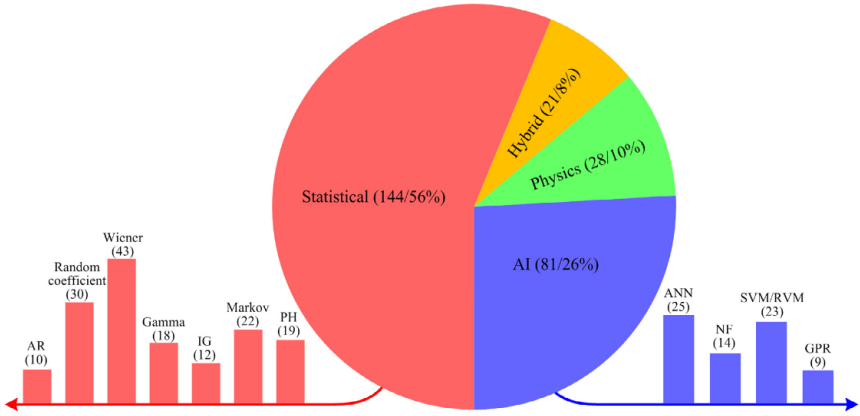


Fig. 2.2 Pie chart of publications related to the four categories of RUL prediction approaches in 2018. The numbers in parenthesis represent the number of articles using the method. Image taken from [LLG+18] with permission.

Figure 2.2, are gaining more and more interest as well. A more recent review gives an exhaustive overview of the deep learning methods that have been applied to RUL estimation and, more generally, to the field of predictive maintenance [RS20].

2.5 Scope of our work

This part of the thesis aims to apply predictive maintenance algorithms to predict an incoming failure of a particular important component in a proton therapy synchrocyclotron, namely a rotating condenser that is subject to wear and possibly failures due to its high speed rotation. The rotating condenser (RotCo) modulates the RF frequency of the synchrocyclotron [KAF+13]. It is composed of a stator and a rotor with eight blades and rotates at a constant speed of 7500 RPM on ball bearings. The bearing system is replaced on a regular basis but an unforeseen bearing failure can always happen. A picture of the system is shown in Figure 2.3. Predicting a failure of the component is of the utmost importance because a failure would require stopping and opening the machine for a replacement, greatly impacting patient treatments that would need to be rescheduled or redirected to photon therapy. Detecting a fault in advance allows to prepare and cause the least amount of downtime possible, reducing costs and maintaining treatment quality. Chapters 3 and 4 participate in this goal by developing

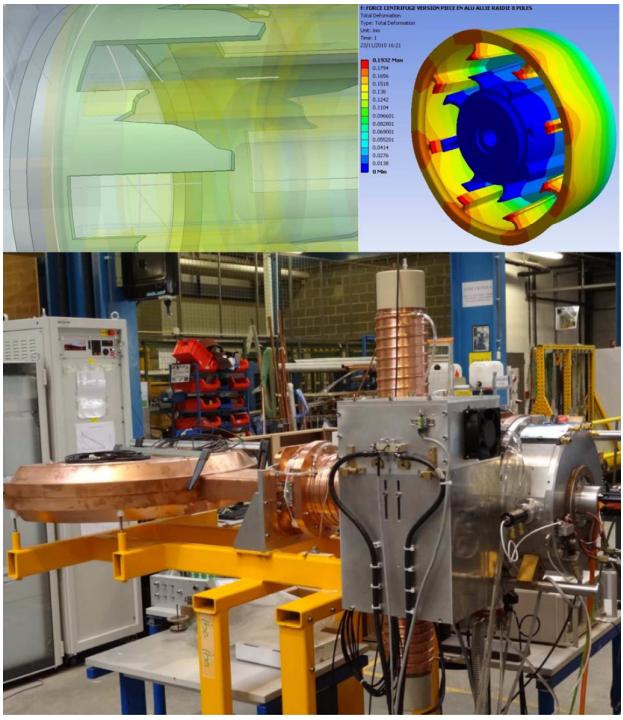


Fig. 2.3 (bottom) RF system of the cyclotron with the rotating condenser on the right. (Top) Detailed view on the rotating condenser. [KAF+13] CC-BY-3.0

predictive models applied to the rotating condenser.

On top of the application, a second goal is to make scientific contributions to the field of predictive maintenance. In Chapter 3, we develop a feature selection method designed to select a relevant and non-redundant subset of features that can be applied to many prognostic tasks without the need for supervised labels. In Chapter 4, we propose a two-level approach composed of a learning algorithm and decision-making part that is generic enough to be applied to a wide range of predictive maintenance problems. We also compare different learning formulations, i.e. binary classification, multi-class classification, regression, and anomaly detection.

3

Feature selection for predictive maintenance

This chapter is adapted from an article published in *International Journal of Prognostics and Health Management* in 2021 originally called *Unsupervised minimum redundancy maximum relevance feature selection for predictive maintenance: application to a rotating machine* [HG21b].

3.1 Introduction

Identifying and selecting optimal prognostic features in the context of predictive maintenance is essential to obtain a good model and make accurate predictions. Several metrics have been proposed in the past decade to quantify the relevance of those prognostic parameters. Other works have used the well-known minimum redundancy maximum relevance (mRMR) algorithm to select features that are both relevant and non-redundant. However, the relevance criterion is based on labeled machine malfunctions which are not always available in real-life scenarios. In this work, we develop a prognostic mRMR feature selection technique, an adaptation of the conventional mRMR algorithm to a situation where class labels are a priori unknown, which we call unsupervised feature selection. In addition, we propose new metrics for computing the relevance and compare different

methods to estimate redundancy between features. We show that using unsupervised feature selection as well as adapting relevance metrics with the dynamic time warping algorithm helps increase the effectiveness of the selection of features for a rotating machine case study.

Our work focuses on the selection of the subset of features that allows the most accurate separation of health states or the best RUL estimation. It falls between steps (2) and (3) of the predictive maintenance framework mentioned in Chapter 2 (see Figure 2.1). It is assumed that a set of HI was previously constructed. There are plenty of scientific articles that tackle this issue. In the case of vibration data, the reader can refer to [WTM17] for a review of the common health indicators (i.e. features).

Feature selection has been applied in the predictive maintenance context in two main ways. In the first approach, the selection of features is based solely on one or several prognostic metrics quantifying the relevance of a feature with respect to the prognostic task. Coble initiated this in her 2010 doctoral dissertation [Cob10] where she derives three complementary metrics of what a good prognostic parameter is. Subsequently, other prognostic metrics were proposed by other authors. An overview of the proposed metrics in the literature is presented in section 3.2.1. A possible drawback with this approach is that the redundancy between features is not taken into account.

The second method for selecting features is based on their relevance to a class label rather than their relevance with respect to a prognostic metric. This falls in the framework of supervised feature selection where we assume that labeled machine malfunctions are available in the case of classification, or that the remaining useful life is used as the label for a regression. Supervised feature selection methods [CS14] can be categorized according to their strategy to select features: filter or wrapper approach. In the filter approach, features are ranked according to a specific criterion, usually a statistical or information theory measure between the feature and the supervised label. As such, the filter approach does not take redundancy between features into account. However, one can use the minimum redundancy maximum relevance (mRMR) algorithm [PLD05] that takes into account both the relevance and the redundancy between features via the mutual information criterion. Several authors applied this mRMR approach with known class labels for selecting features in the predictive maintenance

context, namely e.g. [LYL+17, ZSL+18, YJ19, THG+20, HSQ+20, SMH16]. Liu et al. [LZX13] also include a redundancy analysis in the selection of features, but they do so via a method they call effectiveness–correlation fusion. They compute both effectiveness scores of features using several machine learning criteria (kernel class separability, margin width, scatter matrix, Pearson correlation with labels, etc.) and redundancy between features via Pearson's correlation.

In the wrapper approach, feature selection is performed based on the predictor, i.e. the predictor algorithm is wrapped into a search algorithm that seeks a subset of features that yields the highest classifier performance. This approach is however computationally intensive and depends on the classification/regression algorithm used (meaning that a different set of features would be selected for a different classifier algorithm). A drawback of using the classifier performance as the criterion for selecting features is that the classifier is prone to overfitting [CS14]. Indeed, using classification accuracy in the subset selection can result in a set of features with high accuracy on the training set but a poor generalization power. Moreover, a different set of features will likely be selected on another training set.

While supervised feature selection is the way to go if labeled machine malfunctions are available, it is usually not the case for most real-life applications. Another possibility would be to use the time before failure as class labels for a regression. However, this is not guaranteed to produce good results, since degradation usually appears at a certain point and is rarely a continuous degradation process starting at the beginning of machine life. Moreover, degradation is not necessarily linear, while the regression label based on time before failure will decrease linearly.

The main purpose of this work is to propose a feature selection approach for predictive maintenance that considers both the relevance and redundancy between features without the need for class labels. The idea is to adapt the mRMR algorithm, and more specifically, the "maximum relevance" part where features are not compared to a class label but to a prognostic metric. This prognostic mRMR feature selection technique could be classified as an unsupervised[1] feature selection with a filter approach.

[1]The word *unsupervised* may be misleading since we have some knowledge that a failure occurred. However, we do not have clearly defined class labels, nor do we use any label for selecting a subset of features, hence the term unsupervised.

Several unsupervised feature selection methods have been proposed in the machine learning context. A recent survey [SFCOMT20] details the most common algorithms and provides a taxonomy of those methods. These algorithms can also be divided into filter and wrapper approaches where the former ranks features according to information theory or spectral similarity concepts, and the latter does so mainly through clustering techniques. The criteria for feature relevancy are however difficult to assess. Those methods consist of choosing features to best preserve the manifold structure of original data, seeking cluster indicators or else defining a criterion based on the correlation between features. The last approach is used in [FCBC+19] for a metallurgic application where features are selected according to their lowest pairwise correlation. While those approaches are interesting in the absence of any knowledge about what a good feature should be, we believe there is room for improvement as we can make assumptions about what a good prognostic parameter should be in the context of predictive maintenance. According to [CH09], a good feature should have a monotonic dependence with time, have the same underlying shape across different machines, and show high separability between starting and failure values. Knowing this, we can define prognostic metrics for feature relevancy and use them in a modified version of the mRMR algorithm to select non-redundant features. This association of prognostic metrics and the mRMR algorithm is the central idea proposed in this work.

The remainder of this chapter is structured as follows: section 3.2 discusses previous research on the topic: the existing relevance metrics in prognostics and the minimum redundancy maximum relevance (mRMR) algorithm. In section 3.3, we present our approach, which involves improving existing metrics and adapting the mRMR feature selection approach to a situation without class labels. The algorithm is then tested on a rotating machine application in section 3.4, and section 3.5 concludes.

3.2 Background

First, a set of sensor measurements need to be acquired on several machines from the installation to the failure (or at least until a degradation occurs), as opposed to machines being preventively replaced. Indeed, to identify a relevant subset of features, we can expect to observe signs of deterioration in some of the features for the machines that went through a

failure and not necessarily for those replaced at an early stage. We refer to the data collected from a particular machine as a run-to-failure time series. Let us define $x_i^{(r)} \in \mathbb{R}^{n_r \times 1}$, the i^{th} feature of the run-to-failure time series r, where n_r is the number of samples in the time series. The dataset consists of N features and R run-to-failure time series, i.e. $x^{(r)} \in \mathbb{R}^{n_r \times N}$ for $r = 1, ..., R$. The notation $x_i^{(r)}(t)$ is used to access the t^{th} sample in the array and $x_i^{(r)}(-t)$ to access the t^{th} sample in the array starting from the end. Referring to x_i actually refers to the collection of time series (of possibly different lengths) $x_i = \left(x_i^{(r)}\right)_{r=1,...,R}$. After acquiring data from sensor measurements and constructing a set of features, feature selection can start.

3.2.1 Existing relevance metrics

In her 2010 doctoral dissertation, Coble investigated several prognostic metrics for feature selection [Cob10]. She derived three complementary metrics that define a good prognostic parameter: monotonicity, trendability, and prognosability. The first one quantifies the prognostic feature's underlying positive or negative trend, while trendability indicates the degree to which the features of a set of machines have the same underlying shape. The last complementary metric, prognosability, refers to a measure that encourages well-clustered failure values and high separability with starting values.

Monotonicity is defined as the difference between the number of positive and negative slopes computed for each pair of successive time steps (i.e. by computing $\text{sign}(x(t+1) - x(t))$) divided by the number of time steps. Prognosability is computed as the ratio between the standard deviation of the critical failure values of a set of machines and its mean range between starting and failure values. The result is exponentially weighted to obtain a metric with values between zero and one. The metric encourages well-clustered values, i.e. a parameter with a small standard deviation before failure and a large parameter range across the life of the machine. Finally, the trendability of a feature is defined as the minimum correlation between pairs of machines according to that feature. A caveat in this metric is that it requires computing the correlation with time series of different lengths. Different methods to tackle this issue are discussed in subsection 3.3.1. In [Cob10], the prognostic features are resampled with respect to the fraction of total lifetime into 100 observations, with each observation corresponding to 1% of lifetime.

Other metrics have also been explored since Coble. Camci et al. provide another formulation for monotonicity by dividing a HI into different stages [CMZN13]. Other monotonicity metrics quantify the dependence between the HI and time [JGZN14, LLLL14, JGZN13]. Note that in the aforementioned references, the name trendability is used instead of monotonicity for the metrics that quantify the dependence between the HI and time. This naming convention can be somewhat confusing for the reader. In this work, we shall also refer to those metrics as monotonicity since a correlation between a HI and time induces that the metric is monotonic (since time is monotonic in a time series). Spearman's rank correlation was used in [LLG+16] and [CZD+15] to account for non-linear relationships between the HI and time instead of linear relationships in the conventional Pearson's correlation.

Zhang et al. propose a robustness metric to quantify the smoothness of the degradation trend [ZZX16]. Metrics that quantify the dependence between a HI and different health states via Pearson's correlation (for classification purposes) have been explored in [ZZLH13] and [LZQ16]. Liu et al. also define a metric to quantify the correlation between multiple HI in order to limit the selection of correlated features [LZQ16].

3.2.2 mRMR algorithm

The minimum redundancy maximum relevance algorithm was developed for pattern recognition by Peng et al. [PLD05]. The idea of the algorithm is to select a subset of features $\{x_i\}$ that is both relevant and non-redundant based on the concept of mutual information. The mutual information between two features x and y is expressed based on the joint probability distribution $p(x, y)$ and marginal probabilities $p(x)$ and $p(y)$:

$$MI(x, y) = \sum_{i,j} p(x_i, y_j) \log \frac{p(x_i, y_j)}{p(x_i)p(y_j)} \tag{3.1}$$

It is equal to zero if and only if the two random variables are independent, and higher values mean higher dependency. Mutual information is closely related to the concept of entropy. Indeed, the mutual information between two variables can be expressed as $MI(x, y) = H(x) + H(y) - H(x, y)$ where $H(x)$, $H(y)$ and $H(x, y)$ are respectively the entropy of variables x and y, and the joint entropy between x and y.

From the mutual information point of view, the purpose of feature selection is to select features that jointly have the largest dependency on a target class c. Because it is usually hard to obtain an accurate estimation of multivariate density $p(x_1, ..., x_m)$ and $p(x_1, ..., x_m, c)$, as well as computationally challenging, the mRMR method is used. The concept is based on two concurrent optimization problems, the max-relevance defined as

$$\max_S D(S, c) \triangleq \frac{1}{|S|} \sum_{i=1}^{|S|} \mathrm{MI}(x_i; c) \tag{3.2}$$

and the minimum redundancy defined as

$$\min_S R(S) \triangleq \frac{1}{|S|(|S|-1)} \sum_{i=1}^{|S|} \sum_{j=1, j\neq i}^{|S|} \mathrm{MI}(x_i, x_j) \tag{3.3}$$

The formulation of the optimization problem in equations (3.2-3.3) requires jointly optimizing two different objectives which is not possible as such. Therefore, the problem is reformulated as a single objective optimization by combining the two into a single expression. Two cases are defined:

$$\max(D - R) \tag{3.4}$$

$$\max(D/R) \tag{3.5}$$

that we refer to as OFD (objective function difference) and OFQ (objective function quotient) respectively.

Exact solution to the mRMR problem requires to enumerate $\binom{|S|}{M}$ possible combinations of features, where M is the total number of features and $|S|$ the number of features we wish to select. Note that the number of possible combinations would increase to 2^M should we allow the selection of any number of features. In practice, a near-optimal solution is usually sufficient. Incremental search methods can be used to find a set of features with an $O(|S| \cdot M)$ complexity. Suppose we already have S_{m-1}, the selected set with $m-1$ features, the aim is then to find the m^{th} feature from the set $X \setminus S_{m-1}$. This is done by selecting the feature that maximizes (3.4):

$$\max_{x_j \in X \setminus S_{m-1}} \left(\mathrm{MI}(x_j; c) - \frac{1}{m-1} \sum_{x_i \in S_{m-1}} \mathrm{MI}(x_j, x_i) \right) \tag{3.6}$$

or the feature that maximize (3.5):

$$\max_{x_j \in X \setminus S_{m-1}} \left(\frac{\mathrm{MI}(x_j; c)}{\frac{1}{m-1} \sum_{x_i \in S_{m-1}} \mathrm{MI}(x_j, x_i)} \right) \tag{3.7}$$

In addition to the computational reduction of the mRMR compared to the original joint maximum dependency selection, the authors proved that the mRMR formulation is equivalent to this maximum dependency criterion if one feature is selected (added) at a time [PLD05].

3.3 Unsupervised mRMR feature selection

This section describes our algorithm for unsupervised minimum redundancy maximum relevance feature selection applied to predictive maintenance, which we call prognostic mRMR. Subsection 3.3.1 characterizes the relevance of a feature via three prognostic metrics that are improved versions of the metrics from [CH09]. More specifically, we suggest improvements to increase the robustness of the monotonicity metric and propose alternative strategies to handle the different lengths of the run-to-failure time series in the trendability metric. To take into account the redundancy between features, we adapt the mRMR algorithm in the absence of class labels in section 3.3.2 and propose different strategies to compute the redundancy between features.

3.3.1 Feature relevance: prognostic metrics

Monotonicity

We define monotonicity using Spearman's rank correlation:

$$M(x_i) = \frac{1}{R} \sum_{r=1}^{R} \mathrm{corr}(\mathrm{rank}(x_i^{(r)}), \mathrm{rank}([1, ..., n_r])), \tag{3.8}$$

where $\mathrm{corr}(x, y)$ is Pearson's correlation coefficient between variable x and y:

$$\mathrm{corr}(x, y) = \frac{n \sum x_i y_i - \sum x_i \sum y_i}{\sqrt{n \sum x_i^2 - (\sum x_i)^2} \sqrt{n \sum y_i^2 - (\sum y_i)^2}}$$

and $\mathrm{rank}(x)$ is the relative position label of the observations within the variable. Defining monotonicity in this way instead of counting positive

and negative slopes ($\text{sign}(x(t+1) - x(t))$) as done in [CH09] has the advantage of being a lot less subject to noise in the data as shown in Figure 3.1a. In addition, Spearman's rank correlation is used instead of Pearson's correlation for three reasons. First, Spearman's correlation is better suited for non-linear relationships between the HI and time (Figure 3.1b). Second, it is less sensitive to strong outliers as can be observed in Figure 3.1d. Finally, for mostly uncorrelated data, the two measures are similar (Figure 3.1c).

Prognosability

The prognosability metric used here is the same as in [CH09], except that failure values are not defined as the last value of each machine but rather as the mean failure value of a given time-window T to avoid possibly noisy evaluations, i.e.

$$\text{fv}(x_i) = \left(\sum_{t=1}^{T} \frac{1}{T} x_i^{(r)}(-t) \right)_{r=1,\ldots,R}.$$

The size of the window is application specific and has to be defined by the user. The same applies for start values, i.e.

$$\text{sv}(x_i) = \left(\sum_{t=1}^{T} \frac{1}{T} x_i^{(r)}(t) \right)_{r=1,\ldots,R}$$

Mathematically, prognosability can be expressed as

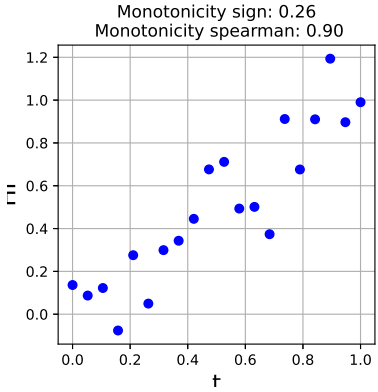
$$P(x_i) = \exp\left(\frac{-\text{std}\left(\text{fv}\left(x_i\right)\right)}{\text{mean}\left(\left|\text{fv}\left(x_i\right) - \text{sv}\left(x_i\right)\right|\right)} \right) \tag{3.9}$$

$$= \exp\left(\frac{-\sqrt{\frac{1}{N}\sum_{r=1}^{R}\left(\sum_{t=1}^{T}\frac{1}{T}x_i(-t) - \mu_f\right)^2}}{\frac{1}{R}\sum_{r=1}^{R}\left|\sum_{t=1}^{T}\frac{1}{T}x_i(-t) - \sum_{t=1}^{T}\frac{1}{T}x_i(t)\right|} \right), \tag{3.10}$$

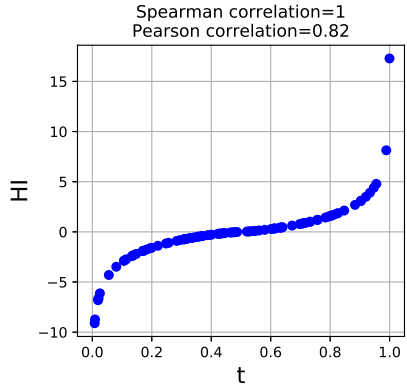
where $\mu_f = \frac{1}{R}\frac{1}{T}\sum_{r=1}^{R}\sum_{t=1}^{T} x_i^{(r)}(-t)$.

Trendability

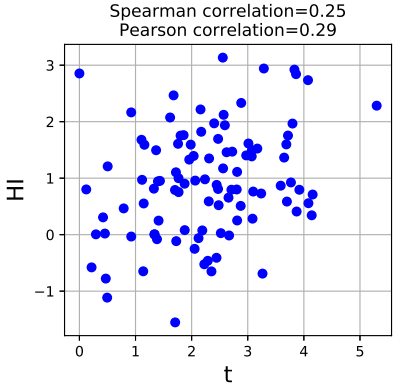
The trendability metric is computed in the same way as in [CH09] i.e. measuring that a feature has the same underlying shape by computing the correlation between pairs of machines. However, we choose to take the mean value instead of the minimum value of the correlations. The reason behind



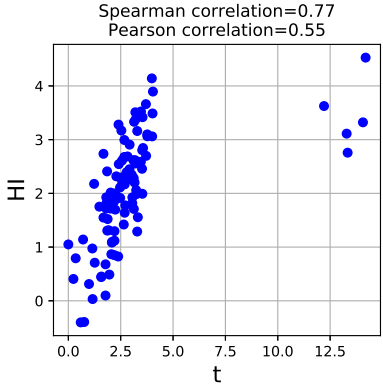
(a) Coble's monotonicity (sign) vs. Spearman's rank correlation



(b) Pearson vs. Spearman correlation



(c) Pearson vs. Spearman correlation



(d) Pearson vs. Spearman correlation

Fig. 3.1 (a) Monotonicity computed with number of positive/negative slopes [CH09] is more sensitive to noise than spearman's correlation. (b) Spearman correlation gives perfect correlation even if the relationship is non-linear. (c) For mostly uncorrelated data Spearman correlation and Pearson correlation give similar results. (d) Spearman correlation is less sensitive to strong outliers in the tails.[2]

this choice is that taking a minimum value could emphasize potentially odd machines or behaviors. However, taking the minimum could also result in a more conservative choice, which might be wanted in some applications. In the end, the choice should therefore be left to the designer

[2]The code and synthetic data used to generate the plots was inspired from author *Skbkekas* on `https://commons.wikimedia.org/wiki/File:Spearman_fig1.svg`. Those examples are purely illustrative, hence, the scales of the axes have no particular meaning.

and in this work, we choose to use a mean value. Mathematically, the trendability can thus be expressed as

$$T(x_i) = \frac{2}{R(R-1)} \sum_{r=1}^{R-1} \sum_{s=r+1}^{R} \left| \text{corr}(x_i^{(r)}, x_i^{(s)}) \right| \text{ for } r, s = 1, ..., R \quad (3.11)$$

To compute the correlation coefficient $\text{corr}(x_i^{(r)}, x_i^{(s)})$ where $x_i^{(r)}$ and $x_i^{(s)}$ are time series with different lengths, several strategies will be compared:

- Resample the time series to the same length with one of the three following solutions and compute the redundancy via the absolute correlation:
  - *Resample long* strategy: Upsample the shortest time series to match the longest-one. This is done in three steps. First, the time index of both series is mapped to 0-1 in the life percentage space. Then, the shortest time series is upsampled via linear interpolation to the same number of samples as the longest one. Finally, the correlation can then be computed as usual.
  - *Resample 100* strategy: Resample the two time series to 100 samples. This is the strategy that was used by Coble in [Cob10]. This is also done in three steps. First, the time index of both series is mapped to 0-1 in the life percentage space. Then, both time series are resampled to exactly 100 samples via a moving average window (each sample then represents 1% of lifetime). Finally, the correlation can then be computed as usual.
  - *History removed* strategy: Truncate the longest time series by removing the samples furthest away from the failure. Note that for this strategy to make sense, both series must have the same sampling rate.
- *DTW* strategy: Keep the time series with different lengths and use the Dynamic Time Warping algorithm (DTW) to compute the distance between the two time series. Dynamic time warping is a technique for comparing time series that computes a distance insensitive to local compression and stretches [G+09]. The algorithm seeks for a warping which optimally deforms one of the two input series onto the other with certain restrictions:

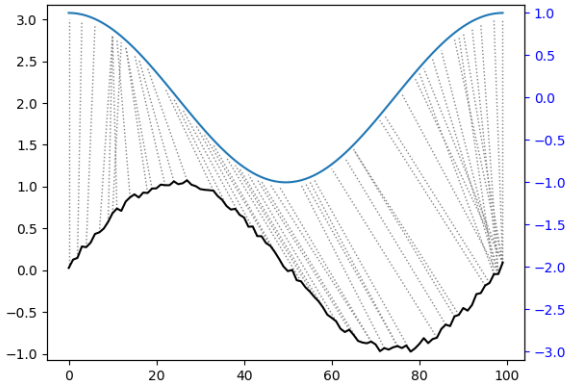


Fig. 3.2 Dynamic Time Warping between two time series[3]

- Every sample from one sequence must match with one or more samples from the other sequence
- The first sample from one sequence must match with the first sample from the other sequence
- The last sample from one sequence must match with the last sample from the other sequence
- The mapping of the samples from one sequence to the other must be monotonically increasing

The distance between the two series is computed, after stretching, by summing the distances of each matched pair of elements (see example in Figure 3.2). Mathematically, it can be formulated as

$$d_\phi(X,Y) = \sum_{k=1}^{T} d(\phi_x(k),\phi_y(k))m_\phi(k)/M_\phi \tag{3.12}$$

where ϕ_x and ϕ_y are the warping functions that remap the time indices of X and Y respectively, $m_\phi(k)$ is a per-step weighting coefficient, and M_ϕ is the normalization constant, which ensures that the accumulated distortions are comparable for different series. For a detailed overview of the algorithm, we refer to [G+09].

[3]The figure was taken from
`https://dynamictimewarping.github.io/python/`

To obtain a score that reflects redundancy with a value between 0 and 1, we take the exponential of the negative distance $\exp(-d_\phi(x_i^{(r)}, x_i^{(s)}))$. The negative term in the exponential is due to the inversely proportional relationship between distance and redundancy (a low distance means a high redundancy). The trendability computation thus becomes:

$$T(x_i) = \frac{2}{R(R-1)} \sum_{r=1}^{R-1} \sum_{s=r+1}^{R} \left(\exp(-d_\phi(x_i^{(r)}, x_i^{(s)})) \right) \text{ for } r, s = 1, ..., R. \tag{3.13}$$

From an intuitive point of view and aside from being able to handle varying length time series, the DTW distance is an interesting measure that allows mapping degradations that occur at different times in different machines. If the degradations show the same underlying trend, the measure will still result in a low distance, and thus a high trendability, even if those degradations did not occur simultaneously.

A single metric for feature relevance: fitness score

To obtain a unique score that quantifies the relevance of a prognostic feature, a fitness score is defined, which is a weighted average of the three metrics mentioned above. It is defined as

$$F(x_i) = w_1 \cdot M(x_i) + w_2 \cdot T(x_i) + w_3 \cdot P(x_i), \tag{3.14}$$

where w_1, w_2, w_3 are the weights associated to each metric. In this work we choose an equal contribution for each metric, i.e. $w_1 = w_2 = w_3 = 1/3$. Note that for each metric to contribute roughly equally to the fitness score, we must further normalize them to spread equally among the range (0-1) with a min-max scaling :

$$m_{\text{scaled}}(i) = \frac{m(i) - \min(m(i))}{\max(m(i)) - \min(m(i))},$$

where $m(i)$ is the metric value associated to feature i.

3.3.2 Taking redundancy into account: prognostic mRMR

Since we assume we are faced with a predictive maintenance application with unknown class labels, a modification to the conventional mRMR algorithm presented in section 3.2.2 is needed.

We adapt the mRMR formulation in eq. (3.2-3.3) where the relevance criterion (mutual information between the feature and class label) is replaced by the fitness score and the redundancy criterion (mutual information between pairs of features) can be interchanged with different measures such as correlation or dynamic time warping. Let $\mathcal{HI}$ be the set of all possible features and let $S \subseteq \mathcal{HI}$ denote the subset of features we are trying to identify, then equations (3.2-3.3) now become:

$$\max_S D(S) \triangleq \frac{1}{|S|} \sum_{i=1}^{|S|} \text{rel}(x_i) \tag{3.15}$$

$$\min_S R(S) \triangleq \frac{1}{|S|(|S|-1)} \sum_{i=1}^{|S|} \sum_{j=1, j\neq i}^{|S|} \text{red}(x_i, x_j) \tag{3.16}$$

where the relevance is defined by the fitness score, i.e. $\text{rel}(x_i) = F(x_i)$ and $\text{red}(x_i, x_j)$ is a measure of redundancy between features i and j. The reformulation of the objective problem as a sum or quotient remains the same as in eq. (3.4) and (3.5) as well as the incremental search methods defined by eq. (3.6,3.7).

One could also define weights associated with each objective D and R and seek optimal ones. However, in this work, we choose to keep unit weights for both D and R. For the contributions in D and R to be similar, we also scale the fitness score and redundancy score via a min-max scaling:

$$D_{scaled}(S) = \frac{D(S) - \min_{i\in S} D(S)}{\max_{i\in S} D(S) - \min_{i\in S} D(S)} \tag{3.17}$$

$$R_{scaled}(S) = \frac{R(S) - \min_{i\in R} R(S)}{\max_{i\in S} R(S) - \min_{i\in S} R(S)} \tag{3.18}$$

for a fair selection process. However, note that the shift (subtraction on the numerator) in the OFD case and the scaling (division in the denominator) in the OFQ case do not impact the outcome of the optimization.

For the redundancy criterion, mutual information is in general used to compare features with each other. However, as mentioned in [DP05], the absolute value of Pearson's correlation can also be used for continuous variables. In this chapter, we compare both measures as well as the dynamic time warping.

To compute the mutual information and estimate the probability distributions, we rely on a non-parametric method based on entropy estimation from k-nearest neighbors' distance. We use the implementation from *scikit-learn* which is based on the algorithms presented in [KSG04] and [Ros14]. To obtain a value between zero and one and thus be able to compare it directly to the usual correlation coefficient, a transformation is performed:

$$\text{corr}_g(x, y) = \sqrt{1 - e^{-2MI(x,y)}} \tag{3.19}$$

We can show that if x, y are normally distributed with correlation ρ, then $MI(x, y) = \frac{1}{2}\log(1 - \rho^2)$ so that $\text{corr}_g(x, y) = \rho$ [GY59].

The third redundancy measure is based on dynamic time warping, which is also used for the computation of the trendability metric in section 3.4.2. In [RGFO17], the authors use the inverse of the dynamic time warping distance as a measure of redundancy for temporal gene data. However, this does not ensure the measure to be between 0 and 1. Instead, we reuse the same approach as for the DTW based trendability, i.e. by computing the redundancy measure as $\exp(-d_\phi(x_i, x_j))$ with d_ϕ defined in eq. (3.12). The redundancy criteria are then averaged across all run-to-failure time series. Mathematically, this is

$$\text{red}_{corr}(x_i, x_j) = \frac{1}{R}\sum_{r=1}^{R}\left|\text{corr}\left(x_i^{(r)}, x_j^{(r)}\right)\right| \tag{3.20}$$

$$\text{red}_{MI}(x_i, x_j) = \frac{1}{R}\sum_{r=1}^{R}\sqrt{1 - e^{-2MI\left(x_i^{(r)}, x_j^{(r)}\right)}} \tag{3.21}$$

$$\text{red}_{dtw}(x_i, x_j) = \frac{1}{R}\sum_{r=1}^{R}\exp\left(-d_\phi\left(x_i^{(r)}, x_j^{(r)}\right)\right) \tag{3.22}$$

As mentioned in section 3.2.2, exact solutions to the feature selection quickly become intractable. Instead, a heuristic is performed where one feature that maximizes one of the two formulations above (OFD or OFQ) is added at a time. The first feature chosen is the feature with the highest fitness score, and the second feature is the one that maximizes one of the two formulations above. We continue adding features until a predefined number of features is obtained. Algorithm 1 fully describes the proposed heuristic and Table 3.1 summarizes the different design choices of the algorithm. Al-

Type	Choices
Relevance	$F(x_i) = w_1 \cdot M(x_i) + w_2 \cdot T(x_i) + w_3 \cdot P(x_i)$ with $T(x_i)$ computed with • Correlation (eq. 3.11) and resample long strategy • Correlation (eq. 3.11) and resample 100 strategy • Correlation (eq. 3.11) and history removed strategy • Dynamic time warping (eq. 3.13)
Redundancy	• Correlation (eq. 3.20) • Mutual information (eq. 3.21) • Dynamic time warping (eq. 3.22) • Relevance only (redundancy not taken into account)
Objective	• OFD (eq. 3.4) • OFQ (eq. 3.5)

Table 3.1 Design choices for selecting features with the prognostic mRMR algorithm.

though we can prove that the set of features selected by the conventional mRMR approach with the incremental search lead to the optimal set of features (in the sense of the maximum joint dependence to the target class) [PLD05], it is not necessarily the case for our method. Unfortunately, there is no way to know how far from the optimal set we are. However, we can get an idea of the predictive power of a set of features by training a classifier with it. By training the classifier on feature sets of different sizes selected by the algorithm, we can estimate the performance and stability of the feature sets.

3.4 Application to a rotating machine

In this section, the prognostic mRMR feature selection is applied to predict incoming failures in a high-speed rotating condenser. Section 3.4.1 de-

input : Dataset: $x = \{x^{(1)}, \ldots, x^{(R)}\}$ where $x^{(r)} \in \mathbb{R}^{n_r \times N}$, features (list of name of the features), $w_1 = w_2 = w_3 = \frac{1}{3}$, trendability_method, redundancy_method (red_{corr}, red_{MI} or red_{dtw} via eq. (3.20-3.22)), n_f (number of features to keep), objective (OFD via eq. (3.4) or OFQ via eq. (3.5))

rel = [] `// relevance: empty array`
red_mat = **0** `// redundancy matrix of size` $N \times N$ `initialized with zeros`
ranked_features = [] `// empty array`
`// Heuristic search`
for *i in 1,...,size(features)* **do**
  `/* Relevance computation */`
  $m = M(x_i)$ (via eq. 3.8) `// monotonicity`
  $p = P(x_i)$ (via eq. 3.9) `// prognosability`
  $t = T(x_i)$ (via eq. 3.11 or 3.13 depending on trendability_method) `// trendability`
  $\text{rel}[i] = w_1 \cdot \frac{m - \min(m)}{\max(m) - \min(m)} + w_2 \cdot \frac{p - \min(p)}{\max(p) - \min(p)} + w_3 \cdot \frac{t - \min(t)}{\max(t) - \min(t)}$ (eq. 3.14) `/* Redundancy computation */`
  for *j in i+1,...,size(features)* **do**
    $\text{red\_mat}[i, j] = \text{redundancy}_{method}(x_i, x_j)$
  end
end
ranked_features.append($\arg\max_i$ rel) `// append the feature with maximum relevance to ranked_features`
features.pop($\arg\max_i$ rel) `// Remove that feature from the feature set`
while *size(features)* $< n_f$ **do**
  score = [] (empty array)
  for *f in features* **do**
    $\Omega = \text{ranked\_features} \cup f$
    $D = \text{rel}[i]$
    $R = \sum_{i=1}^{|\Omega|-1} \sum_{j=i+1}^{|\Omega|} \frac{2}{|\Omega|(|\Omega|-1)} \text{red\_mat}[i, j]$
    score[f] = $D - R$ **if** objective is OFD **else** D/R if objective is OFQ
  end
  best_feature = $\arg\max_f$ score
  ranked_features.append(best_feature)
  features.pop(best_feature)
end
return *ranked_features*

Algorithm 1: prognostic mRMR algorithm

scribes the case study and the evaluation of the performance of the feature selection. Section 3.4.2 compares the different relevance and redundancy measures of the algorithm and section 3.4.3 compares our method with existing techniques proposed in the literature.

3.4.1 Problem description

Test case and features

The predictive maintenance case study is a high-speed rotating condenser (RotCo) modulating the RF frequency inside a cyclotron [KAF+13] and is shown in Figure 2.3. Several sensors are placed inside the machine to gather data. An accelerometer sensor is placed on the condenser to measure vibrations and performs 10-second acquisitions at a rate of 10kHz every hour. Other sensors are placed on the machine to gather data every second. These include temperature sensors, vacuum pressure, and a torque sensor. In total, we have gathered $R = 11$ run-to-failure time series.

After this data acquisition step, several features are constructed. For the vibration data, time-domain and frequency-domain features on each of the 10-second acquisition files are constructed. For the time domain, those include Root mean square (RMS), Median absolute deviation (MAD) which is a robust measure of variability based on the deviations from the median, peak-to-peak values, skewness (third statistical moment), and kurtosis (fourth statistical moment). Those also include metrics based on the peaks of the signals: crest factor, clearance factor, shape factor, margin factor, and max amplitude (for a detailed explanation of those features, the reader can refer to [Mat21]). For the frequency-domain features, we construct the amplitudes at the fundamental frequency and its first three harmonics, the spectral power of all 20 Hz non-overlapping bands from 0-5kHz and finally the amplitudes at characteristic bearing frequencies [SHKB95], i.e.

- Ball Pass Frequency Outer Race: $\frac{n_f}{2}\left(1 - \frac{D_b}{D_p}\cos\phi\right)$
- Ball Pass Frequency Inner Race: $\frac{n_f}{2}\left(1 + \frac{D_b}{D_p}\cos\phi\right)$
- Ball spin frequency: $\frac{D_p f}{2D_b}\left(1 - (\frac{D_b}{D_p}\cos\phi)^2\right)$
- Fundamental train frequency: $\frac{f}{2}\left(1 - \frac{D_b}{D_p}\cos\phi\right)$

where D_p is the pitch diameter, D_b is the ball diameter, ϕ is the contact angle, and n is the number of balls. The first three harmonics of those characteristic frequencies are also included. For the non-vibration data, we perform aggregations of the signals over a one-hour time window to match with the vibration acquisition sampling. Those aggregations include the mean, max, min, standard deviation, skewness, and kurtosis values. This finally results in $N = 317$ features (297 from vibration data and 24 from non-vibration data) computed every hour.

Evaluation of the feature selection's performance

The next step is feature selection. While we can evaluate the best set of features with the algorithm developed (Algorithm 1) for a given method, we cannot conclude which one works better in practice nor how many features should be selected from the obtained ranked features.

Hence, the prediction problem is formulated as a binary classification problem where the machine is either in a healthy state or a faulty state, and we compare the approaches and the number of features to be selected based on the classification score. However, in practice, we do not know when the machine enters a faulty state. In this case, based on engineering expertise, we assume that the machine is likely to be in a faulty state about 5 days before the failure. Moreover, we assume that the machine is in a healthy state from the beginning of its life until 15 days prior to failure. 10 days of data for which we are the most unsure are thus excluded. This results in two artificial classes on which a classification can be performed. Note that in section 3.4.3, other labeling strategies, i.e. different than the 5-day time window, are tested.

The classification task is performed with a Support Vector Machine (SVM) algorithm using a RBF kernel (see e.g. [HDO+98]) which is a suitable classifier for this task. Furthermore, since there are only a few instances of failures in our dataset, a leave-one-out cross-validation is performed where each fold is defined as a run-to-failure time series. The classification score is averaged on the 11 folds of the dataset. No hyperparameter tuning is performed on the SVM as the goal of this article is not to obtain the best prediction capabilities but to compare different feature selection scenarios. The classification score chosen is the F_1 score which is the harmonic mean between the precision and recall, as it is a robust measure against imbalanced datasets (few failure data compared to healthy data).

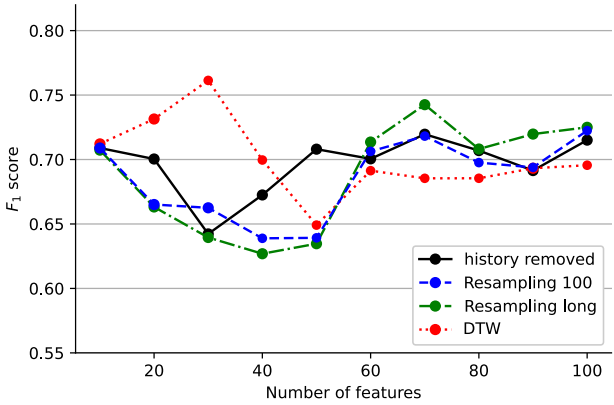


Fig. 3.3 Comparison of the four variants to compute the trendability metric and its consequence on the relevance.

3.4.2 Comparing methods to compute the prognostic mRMR

In this section, the different methods for computing the prognostic mRMR algorithm are compared, as summarized by Table 3.1. Subsection 3.4.2 compares the relevance criteria associated with the different choices in the trendability metric. Subsection 3.4.2 compares the redundancy strategies and the final subsection compares the two objective function formulations.

Relevance measures

The different strategies to compute the trendability metric and hence to compute the relevance of the features are compared. We assume that we want to keep at most 100 features for computational, interpretability, and stability reasons. For each number of selected features k between 10 and 100, we report on Figure 3.3 the cross-validated F_1 score associated with the selection of k best features in terms of their relevance score (see eq. 3.15). The process is repeated for the four strategies to compute the trendability as presented in section 3.3.

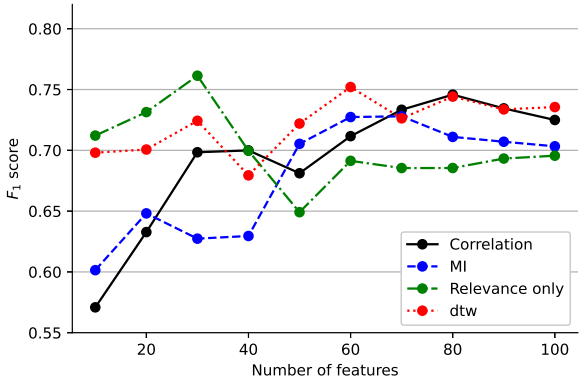
We observe no significant difference between the four approaches proposed except from 20 to 40 features selected, where the DTW approach is outperforming the others. Although a test for consistency on other data should be performed to confirm the trend, DTW seems to be a good candi-

date for comparing the features of different machines and hence for evaluating the relevance of a feature. In addition to the observed positive trend, DTW uses the time structure of the features and can map any instant in the time series of a particular machine to any other instant in another machine. Indeed, intuitively, the start of a degradation phase in a specific machine will most likely never occur at the exact same time in another machine. This is where the correlation measure (on which the other three methods are based) fails by only being able to compare pairs of points at equivalent indexes between two time series.

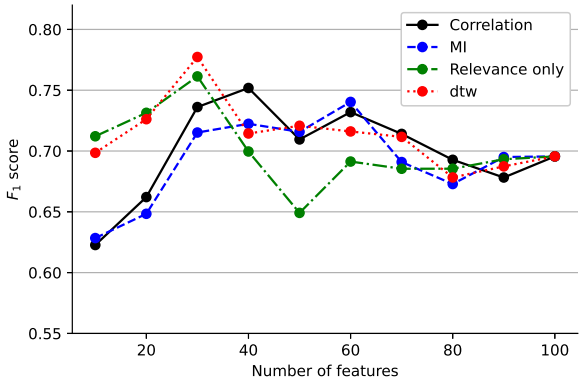
Redundancy measures

This section aims at improving the selection of features by considering the redundancy between features and comparing the proposed redundancy measures. In a first analysis we choose to use the DTW approach for the trendability and relevance computation as it resulted in the best score. We compare the different relevance-redundancy scenarios as well as the no redundancy scenario (based only on the relevance criterion) and choose the OFD as the objective function. The results are reported in Figure 3.4a. We observe that taking into account redundancy does not improve and actually decreases the effectiveness of our model for 10 to 40 features. However, for more features considered, the relevance-only approach performs slightly worse than the others. This can be explained by the fact that the other approaches tend to select features that are sometimes not relevant only because they are highly independent of each other. An improved approach taking the best of both worlds is to preselect features that are at least above a certain relevance threshold or to define a threshold on the maximum number of features to preselect.

In the next analysis the 100 most relevant features are preselected according to their fitness score. The same comparison as in Figure 3.4a is performed with those 100 preselected features. The results are shown in Figure 3.4b. We can observe that the DTW approach now outperforms the relevance-only approach by a small margin and reaches an overall maximum score of 0.78 for 30 selected features. However, for the correlation and mutual information approaches, they still are not able to achieve better performance than the relevance-only approach but they show better performance when 40 to 70 features are selected and similar performance when more than 70 features are selected. The conclusion from Figure 3.4 is that taking into account redundancy between features can definitely help, but a careful preselection should be done first to exclude highly irrelevant



(a) Using all features.



(b) Using only the 100 most relevant features

Fig. 3.4 Comparison of relevance-only approach (None in green) and the 3 mRMR approaches with trendability metric computed via DTW.

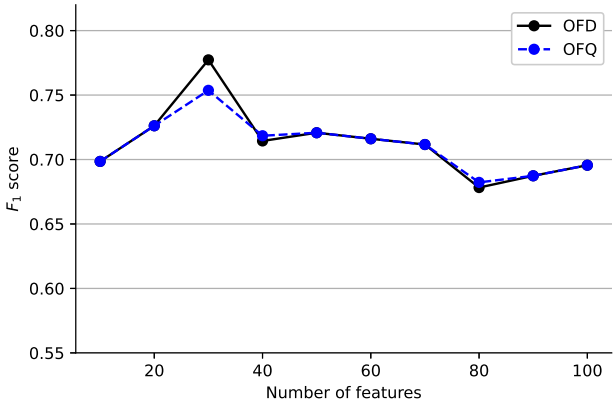


Fig. 3.5 Comparison of the two objective function formulations: OFD (see eq. 3.4) and OFQ (see eq. 3.5)

features. Moreover, as a recommendation, we propose to compute the redundancy between features via the DTW measure as it yields good and stable results.

Objective function formulations

Based on the best approach obtained so far (trendability and redundancy computed with the DTW approach), we compare the two objective function formulations: OFD (eq. 3.4) and OFQ (eq. 3.5). The results are reported in Figure 3.5. We observe no impact on the outcome for the choice of the objective function formulation except for a slightly better performance with OFD when 30 features are selected, which is likely negligible. Since the conventional way to formulate the mRMR is via OFD, we propose to keep this formulation.

3.4.3 Comparison of our method with existing methods

This section compares our prognostic mRMR algorithm with existing feature selection methods proposed in the literature. The prognostic mRMR is computed with the measures that give the best results in the case study, i.e. with the trendability metric and redundancy measure computed via the dynamic time warping measure, OFD chosen as the objective function, and a preselection of the best 100 features according to their fitness score.

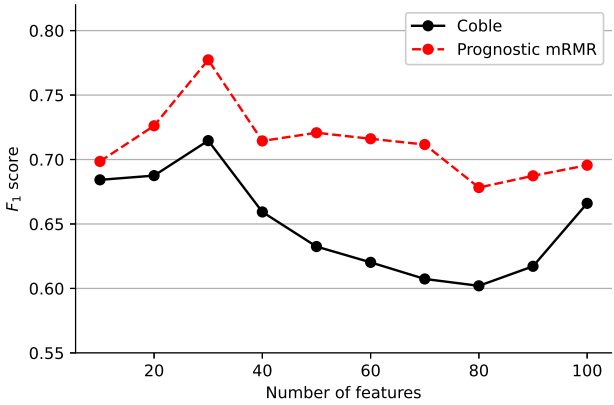


Fig. 3.6 Comparison of the prognostic mRMR with the feature selection from [CH09].

In section 3.4.3, we compare our approach with the feature selection based on the prognostic metrics of [Cob10]. In section 3.4.3, we compare our approach with the conventional mRMR feature selection.

Comparison with feature selection based on the prognostic metrics from Coble

A comparison between our prognostic mRMR algorithm and the feature selection based on the three original metrics defined by Coble [Cob10] is shown in Figure 3.6. We can clearly observe that our method selects a subset of features that is better able to discriminate between a faulty and healthy state (F_1 score is higher), regardless of the chosen number of features. This may be explained by two reasons. First, redundancy is taken into account in the prognostic mRMR approach while it is not in Coble's approach. Second, the prognostic metrics used in the prognostic mRMR are more robust than the ones in Coble. Indeed, this can be observed when comparing Coble's approach (the solid black line from Figure 3.6) with the relevance-only version of our prognostic mRMR (green dotted line from Figure 3.4).

Comparison with the classical mRMR approach

The classical use of the mRMR algorithm requires computing the relevance of the features based on the class labels, usually labeled machine malfunctions such as inner and outer ring defects in the case of bearings. In our

application, such a rich labeling is not available but we can use the class label based on the window labeling described in section 3.4.1.

The labels defining a faulty state were defined somewhat arbitrarily based on engineering expertise. Hence, to fully compare the classical mRMR with our prognostic mRMR algorithm, we compare them for different labeling strategies: 10 different labeling strategies where the machine is considered in a faulty state starting n days before the failure where $n = 1, 2, ..., 10$ are compared. The healthy state spans from machine installation time to 15 days prior to failure as detailed in section 3.4.1. For each labeling strategy, the cross-validated F_1 score is compared for a number of features ranging from 10 to 100. The results are outlined in Figure 3.7. To assess the two approaches, we either consider the global results by comparing the two curves or refer to the maximal score attained for any number of features for each of the labeling strategies. For the latter, we simply need to compare the maximum point on each curve while for the former option, we can either compare the sum of differences (SD) between scores corresponding to the same number of features, or the number of times one curve is above the other (ND), based on the 10 scores computed for the increasing feature set size. Mathematically, this is:

$$SD = \sum_{i \in \{10,20,...,100\}} F_1^{\text{prognostic}}(i) - F_1^{\text{classical}}(i)$$

$$ND = \#\Big\{ i \in \{10, 20, ..., 100\} \colon F_1^{\text{prognostic}}(i) > F_1^{\text{classical}}(i) \Big\}$$

where $F_1^{\text{prognostic}}(i)$ and $F_1^{\text{classical}}(i)$ are the cross-validated F_1 score associated to the feature set of size i computed via the prognostic mRMR algorithm and the classical mRMR algorithm respectively. The prognostic mRMR should be superior to the classical mRMR if $SD > 0$ or $ND > 5$ for a particular labeling strategy. Table 3.2 summarizes the results and individual scores are outlined in Figure 3.7. From a global standpoint, the prognostic approach outperforms the classical approach in 9 out of 10 cases with respect to the SD metric and 7 out of 9 cases + 1 ex aequo with respect to the ND metric. If we only look at the maximum values, the best result is split among the two approaches (5 cases for both). Moreover, prognostic mRMR is consistently better with fewer features and thus is able to select the best compact set of features for the prognostic task.

Label [days]	SD	ND	max score
1	**1.02**	**8**	**prognostic**
2	**0.69**	5	**prognostic**
3	**1.16**	**7**	**prognostic**
4	**0.71**	**7**	classical
5	**0.04**	4	classical
6	-0.08	3	classical
7	**0.47**	**7**	classical
8	**0.46**	**6**	classical
9	**0.88**	**10**	**prognostic**
10	**0.74**	**8**	**prognostic**

Table 3.2 Results of classical vs. prognostic mRMR. Results are highlighted in bold when the prognostic approach outperforms the classical mRMR approach. Prognostic mRMR is better if $SD > 0$ and $ND > 5$.

3.5 Conclusion

We developed an unsupervised minimum redundancy maximum relevance feature selection method for predictive maintenance applications by adapting the conventional mRMR algorithm where the relevance of a feature is computed with respect to prognostic metrics instead of class labels. We also compared different measures to compute the redundancy between features and adapted existing metrics quantifying the relevance of features. We performed a case study for a rotating machine that highlighted the superiority of our feature selection method compared to previous prognostic metrics and the conventional mRMR algorithm, especially for selecting a compact set of features. We also showed that dynamic time warping is a well-suited distance measure for predictive maintenance applications that can help to select a good set of features.

Compared to other unsupervised feature selection methods developed for predictive maintenance in the literature, our method conjointly seeks a set of relevant and non-redundant features while only one criterion is generally achieved for existing methods. Concerning the broader field of pattern recognition and machine learning, the main advantage of our method compared to other unsupervised feature selection methods is the inclusion of prior knowledge about what we know a good prognostic feature should look like (monotonic, etc.) rather than only relying on the underlying data structure.

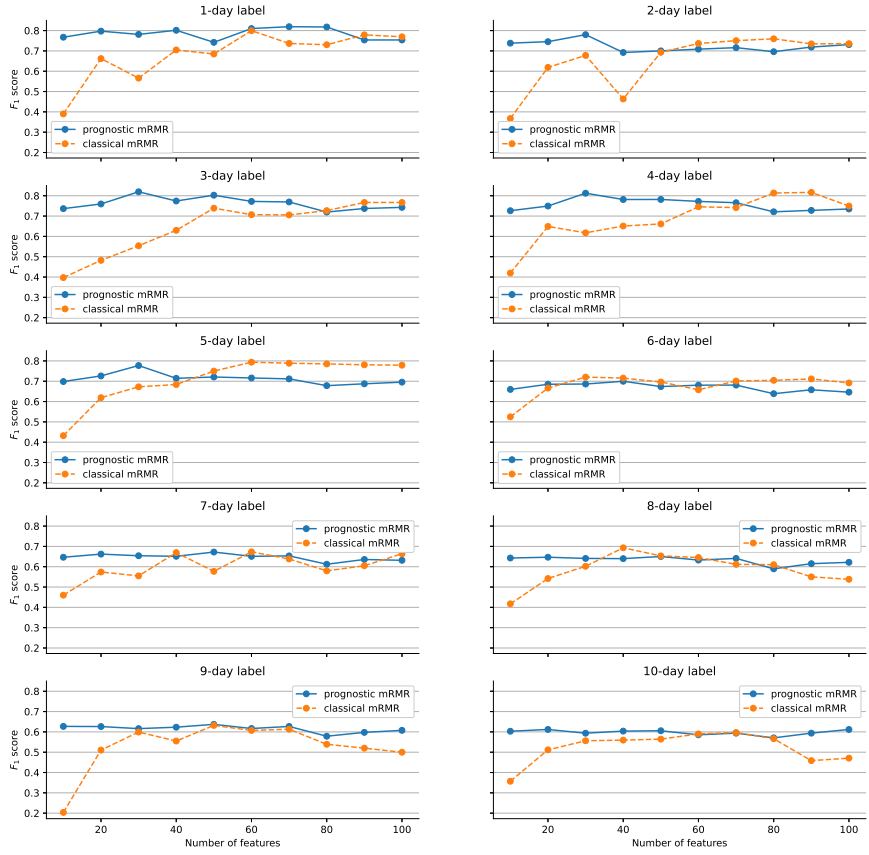


Fig. 3.7 Classical vs. prognostic mRMR feature selection for various labeling strategies.

From the application perspective, we obtained the best results with the trendability and redundancy measures computed with the dynamic time warping and 30 features selected.

The approaches presented in this chapter may still be improved by seeking the best parameters in the fitness metric characterizing the relevance of a feature as well as the weights assigned to the relevance and redundancy in the objective function, which we leave to future work.

4

Comparison of machine learning formulations for predictive maintenance

This chapter is adapted from an article submitted to the Mechanical Systems and Signal Processing journal in 2022 [HJCG22].

4.1 Introduction

Predicting incoming failures and scheduling maintenance based on sensor information in industrial machines is increasingly important to avoid downtime and machine failure. Different machine learning formulations can be used to solve the predictive maintenance problem. However, many of the approaches studied in the literature are not directly applicable to real-life scenarios. Indeed, many of those approaches usually either rely on labeled machine malfunctions in the case of classification and fault detection or rely on finding a monotonic health indicator on which a prediction can be made in the case of regression and remaining useful life estimation, which is not always feasible. Moreover, the decision-making part of the problem is not always studied in conjunction with the prediction phase. This work aims to design and compare different formulations for

predictive maintenance in a two-level framework and design metrics that quantify both the failure detection performance as well as the timing of the maintenance decision. The first level is responsible for building a health indicator by aggregating features using a learning algorithm. The second level consists of a decision-making system that can trigger an alarm based on this health indicator. Three degrees of refinements are compared in the first level of the framework, from simple threshold-based univariate predictive technique to supervised learning methods based on the remaining time before failure. We choose to use the Support Vector Machine (SVM) and its variations as the common algorithm used in all the formulations. We apply and compare the different strategies on a real-world rotating machine case study and observe that while a simple model can already perform well, more sophisticated refinements enhance the predictions for well-chosen parameters.

Predictive maintenance can usually be formulated in one of the two following ways: i) detecting that the machine under monitoring has entered a faulty state, and therefore predicting that a failure is coming, or ii) predicting the remaining useful life (RUL) of the machine. However, in this work, those two concepts are not differentiated as our purpose is to schedule a maintenance procedure based on the current monitoring information, regardless of the type of method used. We propose a framework divided into two levels, the first level consisting of a machine learning model mapping a set of features into a health indicator and the second level being responsible for the actual decision making, where an alarm is raised if the health indicator of the first level crosses a threshold, whose value can be optimized.

4.1.1 ML-based approaches for predictive maintenance

A common assumption of machine learning methods is the availability of sufficient labeled data, which is usually hard to obtain in real-world engineering scenarios [LYJ+20]. A more reasonable assumption would be a case where data has been collected for a few machines that have gone through a failure (or at least a deteriorated state) and underwent a corrective maintenance as well as for some other machines that have been replaced without failure (preventive maintenance). In that case, although different health states are available in the data, an exact labeling is not available since one does not necessarily know when a machine has en-

tered a faulty state. In this work, we investigate different formulations on how to express such problems with a machine learning formulation with variations of the support vector machine (SVM) algorithm. The reason for choosing this family of algorithms over deep learning approaches is threefold. First, we want to isolate the formulation or labeling scenario as much as possible from the algorithm for our comparison. Second, our case study involves only a few instances of failures and SVM tends to have good generalization capabilities even with few instances [CGLRML20]. Finally, the goal of this work is not to seek the best ML algorithm but to compare learning formulations.

4.1.2 Classification

In the case of *classification*, we seek to find whether or not a machine will go to a failure state within a given time window. The duration of the time window is then a parameter that has to be chosen by the user. Surprisingly, very few papers treat the case of finding incoming faults in machines with a labeling purely based on failure time (e.g. the last 5 days are labeled as faulty). One of the papers that is using this kind of approach is [SSP+14] where the authors use a binary classification algorithm with different horizons to define the faulty class and select the one which optimizes a custom cost that is a trade-off between fault detection and unexploited life (i.e. replacing the component too soon and not exploiting the full life of the component).

4.1.3 Anomaly detection

Besides supervised learning, *anomaly detection* can be used for fault diagnosis. In the simple univariate case, a threshold can be learned on a specific feature that is considered anomalous when the threshold is reached (e.g. in [Wan02]). In the multivariate case, a Gaussian distribution can be fitted on healthy data and the Mahalanobis distance used as an indicator of health. This is the approach taken by [JSQ+16, WPZ+16]. Another option is to use the one-class SVM algorithm with a model learned on healthy data such as the authors did in [FFMRFRAB13, MS09]. Deep learning methods based on autoencoders were also applied for anomaly detection in the context of fault diagnosis in [RSVG16].

4.1.4 Regression and remaining useful life estimation

Remaining useful life (RUL) estimation is a regression problem. However, it is traditionally not implemented as a regular supervised learning problem where a mapping is learned between the sensor inputs and the RUL. The standard approach is to extrapolate the trend of one or several health indicators previously extracted from the data via signal processing or machine learning methods until it reaches a predefined threshold. Examples of machine learning approaches using this kind of framework can be found in [SMZ14]. However, a few papers try to directly map the inputs to the actual RUL. This is the case in [KCMM+16] which the authors apply a support vector regression algorithm to NASA's Turbofan engine degradation dataset (CMAPSS) [SGSE08] and more recently with deep learning such as in [LDS18, WYD+18] on the same dataset. However, this dataset is a simulated dataset with manually introduced failures [LLG+18], which can be quite different from real-life scenarios. In practice, it is usually hard to estimate the RUL in engineering scenarios, especially in the absence of clear degradation trends. In this work, a mapping is learned between the feature space and the RUL; however, the final goal is not the RUL itself but the decision taken upon it, i.e. when an alarm for replacement should be raised based on the predicted RUL. Other regression approaches using a different labeling scheme are also tested and discussed in Section 4.3.3.

4.1.5 Decision making

The use of the ML algorithm in the first level of our framework has a different purpose from what is commonly done within the predictive maintenance community. Its role is to fuse a set of features into a health indicator. Indeed, we are interested in extracting a single continuous health indicator from the first level learning stage rather than identifying a certain state in the case of classification, an outlier in the case of anomaly detection, or estimating the RUL in the case of regression. This first level is completely unaware of time, its main purpose is pattern recognition. The resulting health indicator is then fed to the second level of the framework, which is time sensitive and responsible for taking a decision for maintenance based on a smart aggregation of the past values of the health indicator. It sends an alarm signal when the health indicator reaches an optimized threshold.

The rest of the chapter is structured as follows: in Section 4.2, the framework for predictive maintenance is outlined. In Section 4.3, various for-

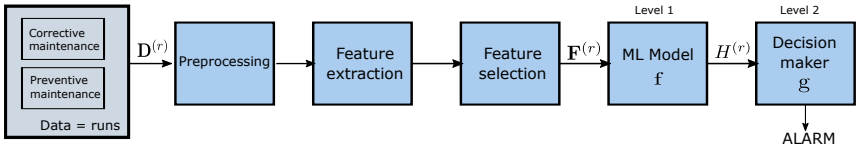


Fig. 4.1 Predictive maintenance framework

mulations for the learning problem are described. The decision-making process is presented in Section 4.4. In Section 4.5, the method of validation of the models is presented. The different methods are tested on a rotating machine case study in Section 4.6, and Section 4.7 concludes.

4.2 Predictive maintenance framework

A flowchart for our predictive maintenance framework is depicted in Figure 4.1. First, data need to be acquired via sensor measurements in the form of time series. The purpose of the data acquisition step is to obtain data from several machines that got a corrective maintenance, i.e. that went through failure (or at least until a deteriorated state) but also possibly from machines that were preventively replaced and did not go into a failure state. We call the data collected from a particular machine from the installation to the replacement (with or without failure) a *run*. Let us define $\mathbf{F}^{(r)} \in \mathbb{R}^{n_r \times N}$, the feature matrix of the run r, where n_r is the number of samples in the time series and N the number of features. The dataset consists of R runs, i.e. $r = 1, ..., R$. The notation $\mathbf{F}^{(r)}(t)$ is used to access the t^{th} sample in the matrix. We also define $x = \left[F^{(r)}\right]_{r=1,...,R}$ the merged feature matrix of size $\sum_r n_r \times N$, and $x_i \in \mathbb{R}^N$ refers to sample number i (regardless of time) of the dataset. Some preprocessing is then achieved on the data collected, which can include filling missing values, removing sensor faults, normalization, etc.

The next step consists in extracting meaningful information from the raw sensor measurements $\mathbf{D}^{(r)}$ i.e. feature extraction. In the context of time-series data, it consists in designing features aggregating sensor inputs that summarize a certain time period rather than a point in time. Those aggregations can be done in the time-domain, frequency-domain, or time-frequency domain (this was also covered in section 3.4.1 of the previous chapter). A description of the feature design is shown in Figure 4.2.

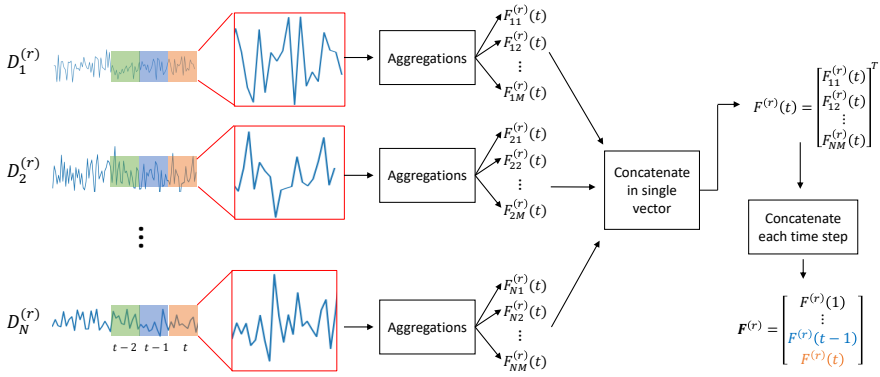
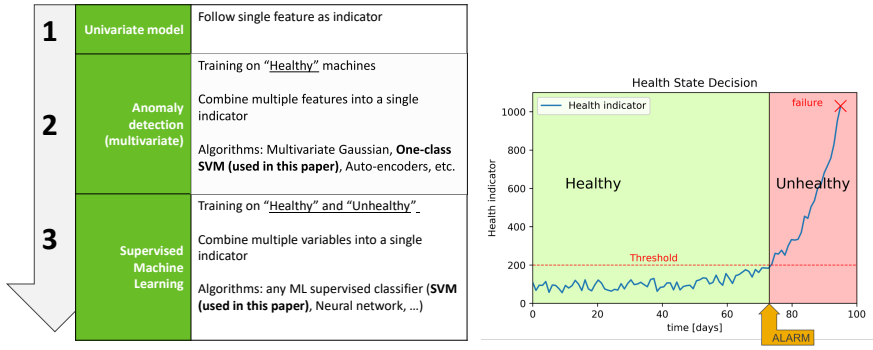


Fig. 4.2 Feature design

A large number of features can be initially produced by the feature extraction step. Selecting a subset of features targets two goals: removing uninformative features and reducing the size of the problems to mitigate some possible overfitting and improve computational efficiency. A filter approach is model agnostic and selects the features *a priori* according to a certain criterion. Since we are comparing different models, we will choose a filter approach so that we have a set of features common in all experiments. However, the criteria to select features for the filter approach should not be supervised, since different labeling strategies will be tested, and choosing one would unfairly favor the algorithm that uses this particular labeling. Instead, we use the unsupervised minimum redundancy maximum relevance feature selection presented in Chapter 3.

From the selected set of features, we wish to establish a causal relationship with the health state of the machine. If the current health state is considered faulty, an alarm should be triggered for repair or replacement. This is done in two steps. In a first step, the input features are fed to a model learned on previous machines, which outputs a single number at each time step (or a single time-series across time), i.e. $h^{(r)}(t) = f(\mathbf{F}^{(r)}(t))$. Then, in a second step, a decision is taken upon this output. An alarm is triggered if $g(h^{(r)}(t))$ exceeds (or goes below) a threshold whose value was optimized beforehand on different machines, where g is a certain time-window transformation applied to the health indicator computed from the model. The function g can be the identity function or a certain aggregation of the out-



(a) First level: Three refinements of the ML model **(b)** Second level: Decision maker

Fig. 4.3 Two-level predictive maintenance: (a) a model translates a set of features into a single indicator (b) a decision is made to trigger an alarm when the health indicator exceeds an optimized threshold (in this case g is the identity function).

put across time such as a moving average or exponential moving average. This two-level predictive maintenance approach is depicted in Figure 4.3.

The next section compares the different strategies that map the set of features into a single health indicator, i.e. the ML model part in the diagram of Figure 4.1.

4.3 First level: Problem formulations

In this section, we compare three different strategies to map a set of features into a health indicator. A first possibility is to simply follow a single feature, a second requires training the algorithm on healthy data and consider as anomalous what deviates from the norm, and finally, the most refined technique is to train on both healthy and unhealthy data with supervised learning algorithms, which can be formulated either as binary classification, multi-class classification or regression. As mentioned earlier, the machine learning models used in this work are all variations of the support vector machine algorithm (SVM). For anomaly detection, one-class SVM is used, for classification and multi-class classification, standard SVM is used and for regression, the support vector regression (SVR) algorithm is used.

4.3.1 Univariate model

In the univariate case, a single feature is followed across time. The selected feature can be chosen according to engineering expertise or according to the maximal relevance score obtained by a feature selection algorithm. In our case, the relevance criterion is the average of three prognostic metrics: monotonicity, trendability, and prognosability [HG21b]. Additionally, some aggregations can be performed on a certain time window up to the current time instant, such as taking the moving average across several hours, or even days if needed. More complex aggregations can also be performed such as exponential moving average, autoregressive models, etc. The health indicator computed from the ML model is then

$$h^{(r)} = f_U\left(\mathbf{F}^{(r)}\right),$$

where f_U is the model (any type of aggregation or simply the identity mapping) and $\mathbf{F}^{(r)}$ contains only one feature in this case.

4.3.2 Multivariate anomaly detection

When performing anomaly detection, the algorithm is only trained on healthy data. When testing on new data, a sample is either marked as an inlier or outlier (i.e. anomalous). In practice, the decision is not binary but is taken based on a threshold for a decision function. For instance, in the case of anomaly detection based on a multivariate Gaussian distribution fitted on healthy data, a decision to flag a sample as anomalous is taken if it is far from the fitted n-dimensional ellipsoid center. Usually, the distance measure chosen is the Mahalanobis distance between the sample and the ellipsoid center. The evolution of that distance across time will therefore be the health indicator on which a decision to trigger an alarm will be made. In our approach, the algorithm chosen for anomaly detection is the one-class SVM, rather than an approach based on the multivariate Gaussian algorithm. Even though the Mahalanobis distance does not apply to the one-class SVM, it involves a similar idea of the distance from a set of *normal* samples.

One-class SVM was proposed as an extension of the support vector machine in the case of unlabeled data [SPST+01]. It tries to estimate the distribution of the input data (considered healthy) by a simpler subset of the input space and estimates a function f that is positive in that subset and

negative on the complement. The corresponding model is formulated as follows:

$$\min_{w,b,\zeta} \frac{1}{2}\|w\|^2 + \frac{1}{\nu\ell}\sum_{i=1}^{n}\zeta_i - b \tag{4.1}$$

$$\text{s.t. } (w^T\phi(x_i)) \geq b - \zeta_i, \tag{4.2}$$

$$\zeta_i \geq 0, i = 1, .., n, \tag{4.3}$$

where x_i are the training vectors, w and b the weights and bias for which we solve, ζ_i are the slack variables allowing some samples to be on the *wrong* side of the hypersurface, $\phi(\cdot)$ is a non-linear mapping to allow for a non-linear boundary, ℓ is the number of samples in the training set and $\nu \in [0, 1)$ is a hyperparameter representing an upper bound on the fraction of training errors and a lower bound of the fraction of support vectors. For our experiments, as well as the other SVM-based models, we used the *scikit learn* implementation which is a wrapper around the LIBSVM library [CL11]. The decision-making process is based on the distance to the hypersurface function, and the decision function is defined as

$$f_{\text{1SVM}}(x) = \sum_{i=1}^{n}\alpha_i K(x_i, x) + \rho, \tag{4.4}$$

where $K(x_i, x) = \phi(x_i)^T\phi(x)$ is the kernel function and α_i and ρ the dual variable and independent term of the optimization problem (4.1). The health indicator computed from the model is then

$$h^{(r)} = f_{\text{1SVM}}\left(\mathbf{F}^{(r)}\right).$$

The same notation for variables, weights, slacks, and kernels will be used throughout the chapter for all SVM-based models.

4.3.3 Supervised learning

Supervised learning refers to algorithms that learn a function mapping from a set of input variables, the features, to a corresponding output, the labels. This requires having the corresponding desired label for each input sample. Those labels can either be numeric values in case of regression or categorical variables in case of classification. In the context of predictive maintenance, those categorical variables can be the fact that a machine is in a healthy or unhealthy state or a certain type of fault on the machine.

If the labels are known, for instance with a healthy machine and a faulty machine on a test bench, then the problem becomes simple, and supervised learning is the way to go. However, labels are usually not available in real-case scenarios. Indeed, we do not necessarily know when exactly a machine enters a faulty state, even if the machine goes into failure at the end of the run. Instead, we use a labeling based on the remaining time before failure. In the case of classification and multi-class classifications, the labels are chosen somewhat arbitrarily by splitting the run into two or more time periods representing healthy or unhealthy states. The chosen duration used to split the run into classes is a parameter that needs to be chosen by the user based on engineering expertise or tuned as a hyperparameter of the problem. In the case of regression, the time before failure can be directly used as the labeling but other possibilities exist and are described in Section 4.3.3.

Binary classification

In the classification approach, a run is divided into two classes: faulty (F) and non-faulty (NF) as follows:

$$y^{(r)}(t) = \begin{cases} NF & \text{if } t < T^{(r)} - w \\ F & \text{if } t \geq T^{(r)} - w \end{cases}$$

where $T^{(r)}$ is the duration of the run r and w is a parameter to choose for the faulty state duration. We thus have to choose a duration *a priori* taking into account that a w too small will lead to a late detection while a w too big could lead to too early detection (and therefore unexploited lifetime). In the case of a machine preventively replaced (no actual failure at the end of the run), the entire run is marked as NF.

The classification algorithm used here is the well-known support vector machine algorithm [BGV92] which finds the hypersurface that separates the classes with maximal margin by solving the following optimization problem:

$$\min_{w,b,\zeta} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\zeta_i \tag{4.5}$$

$$\text{s.t. } y_i(w^T\phi(x_i) + b) \geq 1 - \zeta_i, \tag{4.6}$$

$$\zeta_i \geq 0, i = 1, .., n \tag{4.7}$$

where $y_i \in \{-1, 1\}^n$ are the target values (where F is mapped to 1 and NF to -1) and C is an hyperparameter representing the trade-off between the margin width and the sum of the slack variables ($\sum_i \zeta_i$).

The decision function is based on the distance to the hypersurface, that is defined as

$$f_{\text{SVM}}(x) = \sum_{i=1}^{n} y_i \alpha_i K(x_i, x) + \rho \tag{4.8}$$

and the computed health indicator is thus

$$h^{(r)} = f_{\text{SVM}}\left(\mathbf{F}^{(r)}\right)$$

Multi-class classification

In the multi-class classification approach, a run is separated into more than two classes. Each class represents a different non-overlapping time window between the beginning of the run and the failure. Mathematically, the labels are defined as

$$y^{(r)}(t) = \begin{cases} NF & \text{if } t < T^{(r)} - w_1 \\ F_1 & \text{if } T^{(r)} - w_1 \leq t < T^{(r)} - w_2 \\ F_2 & \text{if } T^{(r)} - w_2 \leq t < T^{(r)} - w_3 \\ \vdots & \\ F_{N-1} & \text{if } t \geq T^{(r)} - w_N \end{cases}$$

where NF is considered the healthy class and $F_i, i = 1, ..., N-1$ are considered the $N-1$ faulty classes with $w_1 > w_2 > ... > w_N$ and increased level of fault severity. In the case of a machine preventively replaced (no failure at the end), the entire run is marked as NF.

The multi-class classification approach is similar to the binary approach. The SVM algorithm is still used but instead of solving one optimization problem as in (4.5-4.7), we solve N optimization problems where N is the number of classes. We use a one-versus-rest strategy[1] where we train a single classifier per class, with the samples of that class being positive ($+1$) and the rest negative (-1), thus keeping the same formulation as in (4.5).

[1] An ordinal strategy might have been better adapted in this case where the faulty classes would be successively increased, i.e. $F_i \leftarrow F_1 + F_2 + ... + F_i$.

Since there are N optimization problems, there are N decision functions such as in (4.8). The decision functions are defined as

$$h_j^{(r)} = f_{\text{SVM}}^j \left(\mathbf{F}^{(r)} \right) \quad \text{for } j = 1, ..., N$$

where the adaptation to multivariate output is described in section 4.4.

Regression

In the regression approach, three labeling scenarios are tested. The first approach is to directly use the remaining useful life (RUL) as labels. In the second approach, instead of using absolute times, relative times are used with the percentage of life used as labels. Finally, a third approach tries to mimic the intuition that a machine is stable at the beginning of its life and deteriorates more and more starting some time before the failure with a piecewise linear function. We refer to this approach as *ReLu*, an analogy to the rectified linear unit in machine learning due to the shape of the labeling function. The three labeling approaches are detailed in Table 4.1. Those labeling strategies are only valid for corrective maintenance. Indeed, runs of machines preventively replaced have to be disregarded in case of RUL or percentage of life strategies. For the ReLu strategy, runs of machines preventively replaced can be kept and labeled as zero across the entire life span of those machines.

Support vector regression (SVR) is the common regression algorithm used for all labeling scenarios. It is a variation of the SVM performing a regression based on the concept of support vectors [DBK+97]. The idea is to find a function $f(x)$ that has at most an ε deviation from the targets y_i by solving the following optimization problem

$$\min_{w,b,\zeta,\zeta^*} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\zeta_i + \zeta_i^*) \tag{4.9}$$

$$\text{s.t. } y_i - w^T\phi(x_i) - b \leq \varepsilon + \zeta_i, \tag{4.10}$$

$$w^T\phi(x_i) + b - y_i \leq \varepsilon + \zeta_i^*, \tag{4.11}$$

$$\zeta_i, \zeta_i^* \geq 0, i = 1, .., n \tag{4.12}$$

where y_i are the targets values, ζ_i, and ζ_i^* are the slack variables allowing some samples to be outside of the tube of radius ε centered around the

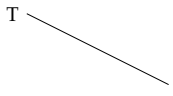
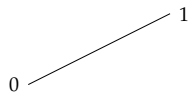
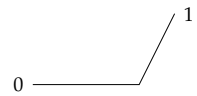
	RUL	Percentage of life	ReLu
Labelling	$h^{(r)}(t) = \frac{T^{(r)} - t}{D}$ with D a normalizing constant	$h^{(r)}(t) = \frac{t - t_0^{(r)}}{T^{(r)} - t_0^{(r)}}$ where t_0 is the start time and T the end time.	$h^{(r)}(t) = \begin{cases} 0 & \text{if } t < T - t_d \\ \frac{T-t}{t_d} & \text{if } t \geq T - t_d \end{cases}$ where T is the end time and t_d is a supposed start of deterioration fixed for training.
Function shape			
Decision	$g\left(h^{(r)}\right) < L$	$g\left(h^{(r)}\right) > L$	$g\left(h^{(r)}\right) > L$

Table 4.1 Labeling strategies for regression & decision making

function, and C is a hyperparameter representing the trade-off between the flatness of f and the sum of deviations larger than ε (that is $\sum_i \zeta_i + \zeta_i^*$). For more information on the SVR and how this optimization problem can be solved efficiently, the reader can refer to [SS04].

The decision function of the SVR is defined as

$$f_{\text{SVR}}(x) = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) K(x_i, x) + \rho \tag{4.13}$$

where α_i, α_i^* and ρ are the dual variables and independent term of the optimization problem (4.9). The computed health indicator (HI) is thus

$$h^{(r)} = f_{\text{SVR}}\left(\mathbf{F}^{(r)}\right)$$

4.4 Second level: Decision making

The purpose of the second level of the predictive maintenance framework is the make a decision on the health indicator computed from the ML model. Let $h^{(r)} = f(\mathbf{F}^{(r)})$ be the HI computed from the model, the decision-maker will trigger an alarm for replacement if $z = g(h^{(r)}) > L$ where g is a

function of time-series $h^{(r)}$ and L is a predefined threshold. In the simplest form, g is the identity function, or the negative function $g(h^{(r)}) = -h^{(r)}$ in case L is a lower bound. When the current output is above (or below) the threshold L, an alarm is raised. The function g can also be an aggregation of past values such as a moving average or an exponential moving average. In case g is a moving average, it is defined as

$$g(y) = \frac{h(t) + h(t-1) + ... + h(t-H)}{H} \tag{4.14}$$

where H is the horizon selected for the aggregation. In case g is an exponential moving average, the aggregation is defined recursively as

$$z_0 = h_0 \tag{4.15}$$
$$z_t = \eta h_t + (1-\eta) z_{t-1} \tag{4.16}$$

The difference with the simple moving average is that in this case, the window size is infinite but the weights are exponentially decreasing. However, the η parameter can be tuned to be interpreted approximately as an H-hour moving average when computed as

$$\eta = \frac{2}{H+1} \tag{4.17}$$

where H is the horizon[2]. For instance, if $H = 12$ and the time between two consecutive sample is 1 hour, $\eta = \frac{2}{12+1} \approx 0.1538$ and is interpreted as a 12-hour moving average.

In the case of binary classification, an alarm is triggered if $z(t) > L$ where L is a previously optimized threshold. For regression, the idea is similar and the decision functions are detailed in Table 4.1. For the multi-class classification, since there are multiple decision functions, the process has to be slightly adapted. An alarm is triggered if the value of the decision function of one of the faulty classes $(F_1, ..., F_{N-1})$ is higher than the healthy class NF. Mathematically, we trigger an alarm at sample i if any $j \neq 1$ satisfy

$$g\left(f_{\text{SVM}}^{F_j}\left(\mathbf{F}^{(r)}\right)\right) > g\left(f_{\text{SVM}}^{NF}\left(\mathbf{F}^{(r)}\right)\right), \tag{4.18}$$

[2]This is because the weights of the exponential moving average and the simple moving average have the same center of mass when $\eta = \frac{2}{H+1}$.

where F_j is the decision function associated with class F_j, $j = 1, ..., N-1$, NF is the decision function associated with the healthy class and g is the function applied to the health indicator.

4.5 Assessing the predictive performance of models

4.5.1 Scoring

The end goal of a predictive maintenance application is to help with the decision to replace or repair the machine under monitoring at the right time. The right time may vary between applications but is usually a trade-off between detecting failure and limiting false alarms. As a first goal, we want to check whether or not a fault (in the case of a run with failure) can be detected and minimize false alarms. As a second goal, we want to score the timing of the alarm. This concept can be translated into two metrics. We define a false positive as an alarm that was triggered too early, in our case more than 15 days in advance. A true positive is defined as an alarm raised between 0 and 15 days in advance. While true positives are only relevant for corrective maintenance runs, false positives are relevant for both corrective and preventive maintenance runs. Indeed, an alarm raised at any time for a machine that was preventively replaced is considered a false positive. We thus define two metrics, the false positive rate:

$$\text{FPR} = \frac{\text{FP}}{C + P}, \tag{4.19}$$

and the true positive rate

$$\text{TPR} = \frac{\text{TP}}{C}, \tag{4.20}$$

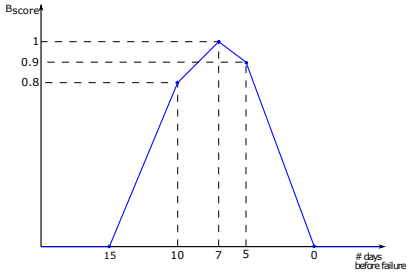
where C and P are the numbers of corrective and preventive runs respectively, and FP and TP are the numbers of false positives and true positives, respectively. Note that we do not assess each individual prediction for each time step, but only evaluate the quality of the first trigger for each run (hence the earliest for that run), so that these metrics are representative of real-world use.

We combine those two metrics into a single metric by taking their harmonic average, similar to what we would do to compute the F_1 score between precision and recall in a conventional classification scenario. Moreover, we add a β parameter, controlling the importance of the TPR over the $1 - \text{FPR}$.

We call this metric F_{score} and define it as

$$F_{\text{score}} = (1 + \beta^2) \cdot \frac{(1 - \text{FPR}) \cdot \text{TPR}}{\beta^2 \cdot (1 - \text{FPR}) + \text{TPR}} \tag{4.21}$$

Since it is also important to score the quality of the timing at which the alarm is raised, another metric called the business metric (B_{score}) is defined, which was designed for the application described in Section 4.6 (although a similar metric could be used for other applications). It provides a score between zero and one (higher is better) computed from a piecewise-linear function of the number of days between the alarm prediction and the actual failure:



- A prediction 7 days ahead is considered optimal and gives a perfect score.
- Predictions between 7 and 0 days assign a score that decreases linearly to zero (with a slightly lower slope between 7 and 5 days).
- Predictions ranging from 7 to 15 days are assigned a score linearly decreasing from 1 to 0 (with a slightly lower slope between 7 and 10 days).
- Predicting a failure more than 15 days in advance leads to a zero score to reflect the unexploited lifetime.

Note that this business score is only applicable for corrective maintenance. While F_{score} is a single score resulting from an ensemble of runs, B_{score} is defined per run and one must take the average across all corrective maintenance to obtain a single score, i.e. $\overline{B}_{\text{score}} = \sum_r \frac{B_{\text{score}}^{(r)}}{C}$. Finally, we combine the F_{score} and the business score into a single score that takes into account both the corrective and preventive maintenance runs. We define $\alpha \in [0, 1]$ as the weight associated to the F_{score} and $1 - \alpha$ the weight associated to $\overline{B}_{\text{score}}$. Since the F_{score} is used for both corrective and preventive maintenance while B_{score} is only applied to corrective maintenance, we must further multiply $1 - \alpha$ (the weight associated to B_{score}) by the ratio of corrective maintenance over the the number of runs, i.e. $\frac{C}{C+P}$. In the end, the final score is defined as

$$\frac{\alpha}{(1-\alpha)\frac{C}{C+P}+\alpha}F_{\text{score}}+\frac{(1-\alpha)\frac{C}{C+P}}{(1-\alpha)\frac{C}{C+P}+\alpha}\overline{B}_{\text{score}} \tag{4.22}$$

4.5.2 Cross validation

In order to obtain an unbiased estimate of the performance of the algorithms as well as tuning the different hyperparameters of the machine learning model and the threshold for the decision making, one must cross-validate the results. The usual way to validate a model in the context of machine learning is to split the data into a training set, a validation set, and a test set. The training set is used to train the algorithm, the validation set is used to tune the hyperparameters of the model, and the test set, a completely independent set, is used to assess the final prediction on unseen data.

In the context of predictive maintenance, some precautions are necessary. Splitting the data into training, validation and test sets cannot be done in a completely random way. Recall that prediction occurs in a continuous fashion along with the time series, i.e. we classify or predict at each time step. Hence, due to the temporal nature of the prediction task, data are correlated in time and one cannot use information learned in the future to predict the past or the present. Therefore, the training set should not contain data that are further in time than the validation and test set. An even better practice is to split data per run, meaning that data from a particular machine cannot be split among different sets. This is the approach that we take. Some machines are assigned to the training set, some to the validation set, and the rest to the test set.

However, in most real-life applications, machine runs-to-failure are scarce. Hence, it is difficult to build a sufficiently large (in terms of the number of corrective maintenance) training set or test set. Therefore, we perform a cross-validation, where a fold is defined as a run. However, a single layer of cross-validation, for instance, a leave-one-out cross-validation, is still biased because the hyperparameter optimization has *seen* all data. Since data are scarce, it is not an option to leave a few runs as the test set as they would most likely not represent very well the behaviors of all machines. Instead, we perform what we call a double cross-validation. The framework for double cross-validation is outlined in Figure 4.4. It consists of

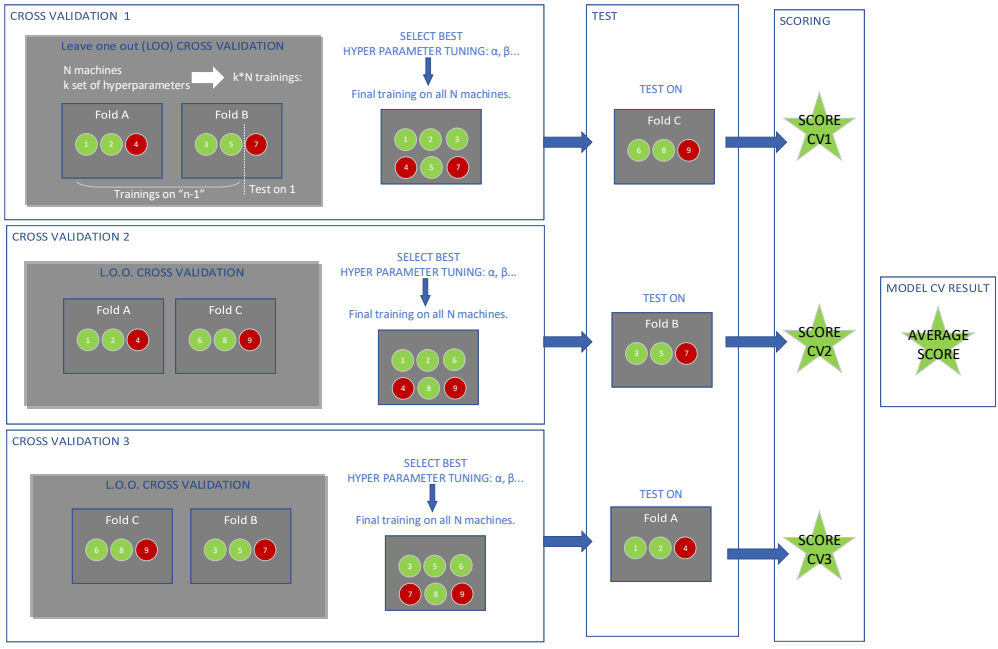


Fig. 4.4 Example of the double cross-validation process for 9 machines subdivided into 3 folds.

two levels. In the inner loop, a leave-one-out cross-validation is performed to tune the hyperparameters of the ML model. Then, a model is trained on all data of the inner loop (training set + validation set) with the best set of hyperparameters found in that inner loop and the threshold is tuned for the decision-making. In the outer loop, the model is tested on an independent set of runs. Then, a new test set is selected at the next outer loop. In the end, we obtain not one but multiple models (with possibly different sets of hyperparameters) that correspond to the number of folds on the outer loop. The average of those scores results in a mostly unbiased[3] estimation of the performance of our algorithms. Note that a fold can contain both corrective or preventive maintenance. We target an equal distribution of those two categories of runs across the different folds.

[3] Although using a nested cross-validation considerably reduces the bias in the performance results, we cannot technically say that it is unbiased since the size of the training set is $(K-1)/K$ as big as the original set. Using bootstrapping methods can circumvent this issue.

Raw signals	Feature extraction	Selected features
Vibration amplitude	*Time-domain:* RMS, MAD, Peak to Peak Amplitude, Skewness, Kurtosis, Crest Factor, Clearance Factor, Shape Factor, Margin Factor, Max Amplitude *Frequency-domain:* Amplitude 1N[5], 2N, 3N, every 20Hz band from 0-1kHz, BPFO 1-3N, BPFI 1-3N, BSF 1-3N, FTF 1-3N	MAD, Margin Factor, BPFO 3N, Peak to Peak Amplitude, Crest Factor, Amplitude 1N, RMS, Spectral Amplitude at 350 +- 10 Hz
Non-vibration features: Bearing Temperature Pyrometer Temperature Torque Vacuum Pressure	*Time-domain:* Mean, Max, Min, Standard Deviation, Skewness, Kurtosis	Torque Mean Torque Max

Table 4.2 Feature design

4.6 Application to a rotating machine

4.6.1 Problem description

The predictive maintenance case study we consider in this work deals with the high-speed rotating condenser (RotCo) illustrated in Figure 2.3 with the data collection and feature engineering described in section 3.4.1 of the previous chapter. Table 4.2 gives an overview of the extracted and selected features. The ten best features[4] are selected according to the feature selection method presented in Chapter 3. This is an unsupervised feature selection; therefore, no formulation is favored. In total, 11 corrective maintenance and 28 preventive maintenance runs have been gathered, i.e. a total of $R = 39$ runs.

All the formulations presented in Section 4.3 are tested on this real-world application. To avoid comparing all combinations of the approach of the first level with the ones of the second level, we first compare the approach

[4] A careful reader will notice that the best performance was obtained with 30 features in Chapter 3 and not 10. This is because the work presented in that chapter used a previous version of the feature selection algorithm with fewer features.

of the first level with the simplest second level decision-making process, where the function g is the identity function. This means that the health indicator is directly compared to a threshold and an alarm for replacement is raised whenever this output exceeds (or goes below) the optimized threshold. This allows us to select a subset of formulations obtaining the best scores, which are then tested against different aggregations g in Section 4.6.3.

The double cross-validation is performed as follows: 11 corrective maintenance runs are distributed among 11 folds and the 28 machines preventively replaced are distributed equally among those 11 folds. After leaving one fold aside as a test set, we perform the inner loop of the cross-validation where the ML model is trained consecutively on all but one run (where a fold is defined as a run). The inner cross-validation allows us to tune the hyperparameters of the model. Then the test set changes to the next outer fold and the whole process starts again.

4.6.2 First level: training & results

Table 4.3 summarizes the different hyperparameters of the problem for each formulation as well as the different labeling scenarios tested. Notice that the univariate model is absent from the table because no training is required at the first level of the framework.
For binary classification, several horizons are tested to split the data among healthy and unhealthy, ranging from 3 to 10 days. The F-score is used to select the best set of hyperparameters. For multi-class classification, two labelling scenarios are tested. The first one includes three classes defined in the following way: from 0 to 5 days prior to failure, 5 to 10 days, and more than 10 days. The second labelling scenario includes 6 classes defined as 2-day periods from the failure and a class defined as more than 10 days prior to failure. Hyperparameters are selected according to the mean of the F-score.

For the One-Class SVM implementing anomaly detection, the model is only trained on data further than 15 days prior to failure for corrective maintenance and all data for preventive maintenance. To select the best hyperparameters set however, the model is tested on all data of a run and the hyperparameters that obtained the best F-score are selected.

Formulation	Hyperparameters	Scoring metric
Binary classification	$C = [10^{-2}, 10^{-1}, 1, 10]$ $\gamma = [10^{-5}, 10^{-4}, 10^{-3}]$ kernel: linear, RBF horizon = [3,5,7,10] days	F-score[6]
Multi-class classification	$C = [10^{-2}, 10^{-1}, 1, 10]$ $\gamma = [10^{-5}, 10^{-4}, 10^{-3}]$ kernel: linear, RBF labelling 1: 3 classes (0-5 days, 5-10days, >10days) labelling 2: 6 classes (0-2 days, 2-4days, ..., 8-10days, >10days)	Unweighted mean of F-score associated to each label
One-class SVM	$\nu = [0.01, 0.05, 0.1, 0.5]$ $\gamma = [10^{-4}, 10^{-3}, 10^{-2}]$ kernel: linear, RBF Horizon = 15 days	F-score (with failure data included during testing)
Regression (RUL, RUL percentage, ReLu)	$C = [10^{-2}, 10^{-1}, 1, 10]$ $\gamma = [10^{-5}, 10^{-4}, 10^{-3}]$ $\epsilon = [0.01, 0.1, 0.5]$ kernel: linear, RBF horizon for ReLu: $t_d = 10$ days	Mean absolute error: $\mathrm{MAE}(y, \hat{y}) = \frac{1}{n}\sum_{i=0}^{n-1} \lvert y_i - \hat{y}_i \rvert$.

Table 4.3 Training & hyperparameter tuning for the first level of the predictive maintenance framework

For the RUL and RUL percentage formulations, the training can only be done on corrective maintenance runs. For the ReLu formulation, preventive runs can be included, as the labels associated with those runs can be defined as zero across their lifetime, since the ReLu is defined as nonzero only at a time t_d prior to failure and monotically increasing until the actual failure. Parameter t_d is fixed at 10 days in all our tests. The metric used to select the hyperparameters is the Mean absolute error for all regression formulations.

The results of the double cross-validation for all formulations are reported in Table 4.4. The count of true positives and false positives are summed

[6] The F-score is defined as the harmonic mean between precision and recall, i.e. $F_{\text{score}} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$

Formulation	False Positives	True Positives	Business score	F score	Final score
Univariate model	**2/39**	8/11	0.436	0.894	0.855
One-class SVM	16/39	5/11	0.257	0.556	0.531
Binary classification	3 days: 6/39 5 days: 5/39 7 days: 4/39 10 days: **2/39**	3 days: 9/11 5 days: 9/11 7 days: 9/11 10 days: 9/11	3 days: 0.495 5 days: 0.526 7 days: 0.465 10 days: 0.500	3 days: 0.840 5 days: 0.860 7 days: 0.880 10 days: **0.919**	3 days: 0.811 5 days: 0.831 7 days: 0.845 10 days: **0.883**
Multi-class classification	3-class: 5/39 6-class: 5/39	3-class: 9/11 6-class: 8/11	3-class: 0.493 6-class: 0.468	3-class: 0.860 6-class: 0.838	3-class: 0.829 6-class: 0.807
RUL	32/39	2/11	0.141	0.180	0.177
RUL percentage	12/39	6/11	0.254	0.657	0.622
ReLu	6/39	**10/11**	**0.617**	0.858	0.837

Table 4.4 Results of first level. A run is considered a false positive if an alarm is raised more than 15 days in advance. An alarm is considered a true positive if it was raised between 0 and 15 days prior to failure in case of a run with failure. The business score is the mean of the business scores for all corrective maintenance. Bold values are the best results for each criterion. Final score is computed via equation (4.22) with $\alpha = 0.75$ and $\beta = 0.5$ and is the score of interest that determines the best formulation.

across the different folds of the test set (every run is at least in one test set) and the F_{score} of equation (4.21) is computed with those counts with parameter $\beta = 0.5$ to give more emphasis on avoiding false positives. The business score is averaged on all corrective maintenance runs since the business score is not defined for runs with preventive replacement. Finally, the final score is computed according to equation (4.22) with parameter $\alpha = 0.75$ to increase the emphasis on the F_{score}.

Surprisingly, we observe that the simplest model, the univariate model based on the best feature with respect to the three prognostic metrics from [HG21b], performs quite well, better than most of the approaches tested even though the other approaches also include this feature within their ten selected features. This feature is the median absolute deviation of the vibration amplitude averaged over the last 12 hours. It is the median of the absolute deviation from the data's median and is computed as $\text{MAD}(x) = \text{median}(|x_i - \bar{x}|)$ where $\bar{x} = \text{median}(x)$.

In the second degree of refinement (see Figure 4.3a), the one-class SVM performs poorly with many false positives resulting in a low final score. For the third refinement, i.e. supervised learning, we can make several observations. For binary classification, the more we increase the size of the faulty class, the better performance we obtain. This could be explained by two different reasons. The first one is that the class imbalance is reduced when the size of the faulty class increases. The second reason is that signs of faults already appear up to 10 days in advance. In the multi-class classification, performances are slightly lower than for the binary classification. Thus, splitting the runs into multiple classes does not seem to help.

For the three regression formulations, the results are quite different from each other. Directly mapping the features input to the RUL does not seem to work at all. This could be explained by the fact that there is too much disparity between the life spans of the different runs, or simply not enough failure samples. This can also partly be explained by the fact that the ϵ parameter in the SVR formulation should be tuned more carefully for this type of formulation, since the labels are not scaled to 0-1 like for the RUL percentage or ReLu. The percentage of life formulation performs better than the conventional RUL but is still far behind the other formulations. This could also be explained by the fact that training is only performed on corrective maintenance for those two formulations. Finally, the ReLu formulation performs quite well although the number of false positive is high.

In conclusion for this analysis of first-level formulations, we find that no method clearly outperforms all the others and that depending on which criteria we focus on (i.e. which column of Table 4.4 we look at), several formulations can be recommended. The trade-off between high failure detection rate and low false alarm is one of those determining aspects. In our case study the binary classification formulation gives the best results in terms of the final score for a well-chosen window size.

In the next section, different decision-making function g are tested on the best algorithms obtained at the first level.

4.6.3 Second level: training & results

In this section, we compare the application of a function g on the computed health indicators of the first level that lead to the best scores, i.e. classifica-

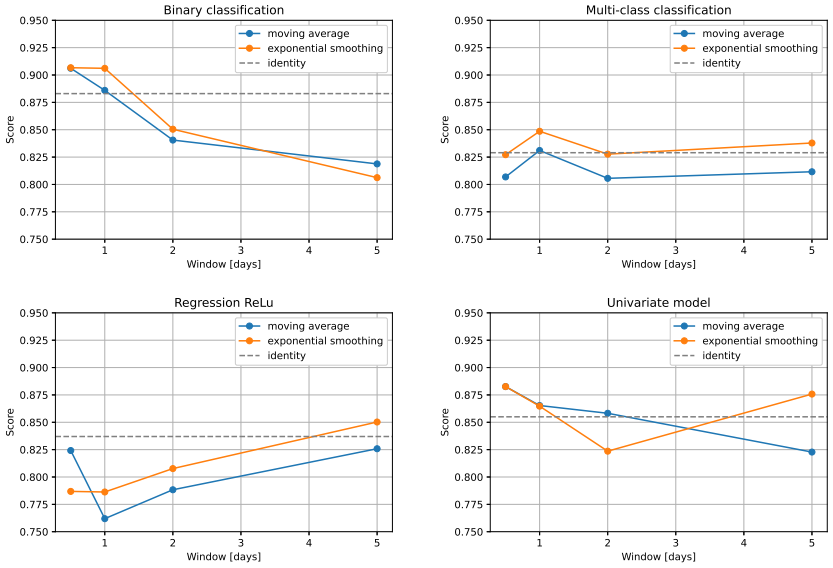


Fig. 4.5 Results of second level

tion with a 10 days window for the faulty class, the multi classification with 3 classes, the ReLu formulation and the univariate model. We compare the case where g is a moving average and an exponential smoothing. For all cases, we test a moving window of size 12-hours, 24-hours, 48-hours and 5-days. In the case of the exponential smoothing, since we have an infinite window, we use equation (4.17) to match the parameter η with the window size. The results are outlined in Figure 4.5 and a more detailed version in Tables 4.5-4.8. We observe that aggregating the computed HI with a moving average or exponential smoothing is not a guarantee for better results. However, by tuning the window size parameter, we are able to obtain better performance than the identity mapping for the exponential smoothing in all models.

We observe that when the window size at the second level increases, the number of false alarms (false positives) decreases, but also the number of cases detected (true positives), which sometimes translates into a lower score. The absolute minimization of false alarms might be wanted in some applications; thus, this second level of aggregation could be effective in that case. We also observe that the exponential smoothing performs better

in general than the simple moving average. Therefore, using a scheme of decreasing weights with respect to time might be a good idea. The relatively modest impact of the function g on the final results might also be due to the fact that the features inputs were also averaged across a 12-hour time-window before the model is applied.

In the end, it is hard to give a definitive conclusion about the best formulation to use in a predictive maintenance scenario, due to the scarcity of runs and failure data, which greatly impacts the final score. However, some insights can still be taken. A simple univariate model can already be effective provided that a health indicator with high predictive power has been found. If we have corrective maintenance data, a supervised learning algorithm should lead to better performance than a one-class classification or anomaly detection algorithm. The binary classification formulation gives the best results when the time-window splitting healthy and faulty data is carefully selected. Finally, while directly mapping the RUL to the inputs is not a good idea, we find that a formulation such as the ReLu formulation which can be trained on both preventive replacements runs and runs-to-failure with a labelling mimicking an increasing fault severity starting at a certain time t_d before the failure, results in good performance.

4.7 Conclusion

The aim of predictive maintenance is to avoid failure by replacing or repairing a machine at the right time. This is done by raising an alarm before the failure, and ideally sufficiently in advance to ease the maintenance scheduling. The best timing for detecting a failure was encoded via a business metric in our work. We developed a two-level framework to tackle this problem. In the first level, we compared several formulations to discriminate healthy and unhealthy states of the machines based on the remaining time before failure, in the absence of true labelled machine malfunctions. In the second level, we compared different ways to exploit the health indicator computed from the learning algorithm and we optimized a threshold for making the decision on when to raise an alarm. Our two-level framework approach gives promising results on the rotating machine case-study we considered. One of the key take-away is that the most complex models do not necessarily give the best results, and an univariate model can already perform very well when a powerful predictive feature can be first extracted. Although more complex multivariate

models rely on more information and should theoretically lead to a more robust solution, it is not always the case, especially in situations where relatively little training data is available. Special care must be taken to select the proper hyperparameters. Depending on the sought trade-off between false alarm and failure detection rate, certain methods and thresholds will perform better than other combinations, which should be carefully selected by the user and probably depend on the considered case study.

From the application perspective, the binary classification approach with a faulty class defined with a window length of 10 days combined with an exponential smoothing with a window length of 12 hours gave the best results: detection of an incoming failure happens in 90% of the cases (more than 75% of those detections being between 2 and 10 days ahead of the failure), and only triggers 5% of false alarms.

Aggregation	FP	TP	B score	F score	Final score
MA 12h	2/39	10/11	0.54	0.940	0.906
EX 12h	2/39	10/11	0.546	0.940	0.906
MA 24	2/39	9/11	0.531	0.919	0.886
EX 24h	2/39	10/11	0.54	0.940	0.906
MA 48h	3/39	11/11	0.464	0.937	0.84
EX 48h	4/39	9/11	0.532	0.880	0.85
MA 5D	2/39	7/11	0.339	0.863	0.819
EX 5D	0/39	6/11	0.265	0.857	0.806

Table 4.5 Binary classification results with moving average (MA) aggregation and exponential smoothing (EX)

Aggregation	FP	TP	B score	F score	Final score
MA 12h	5/39	9/11	0.438	0.860	0.824
EX 12h	6/39	8/11	0.441	0.819	0.787
MA 24	6/39	7/11	0.423	0.793	0.762
EX 24h	6/39	8/11	0.435	0.819	0.786
MA 48h	6/39	8/11	0.458	0.819	0.788
EX 48h	5/39	8/11	0.48	0.838	0.808
MA 5D	2/39	7/11	0.42	0.863	0.825
EX 5D	0/39	7/11	0.348	0.897	0.85

Table 4.6 Regression ReLu results

Aggregation	FP	TP	B score	F score	Final score
MA 12h	5/39	8/11	0.47	0.838	0.807
EX 12h	5/39	9/11	0.472	0.860	0.827
MA 24	5/39	9/11	0.518	0.860	0.831
EX 24h	5/39	10/11	0.525	0.879	0.848
MA 48h	5/39	8/11	0.455	0.838	0.805
EX 48h	5/39	9/11	0.478	0.860	0.827
MA 5D	5/39	8/11	0.525	0.838	0.811
EX 5D	3/39	8/11	0.433	0.875	0.838

Table 4.7 Multi-class classification results

Aggregation	FP	TP	B score	F score	Final score
MA 12h	2/39	9/11	0.492	0.919	0.883
EX 12h	2/39	9/11	0.492	0.919	0.883
MA 24	3/39	9/11	0.496	0.9	0.865
EX 24h	3/39	9/11	0.489	0.9	0.864
MA 48h	2/39	8/11	0.475	0.894	0.858
EX 48h	4/39	8/11	0.465	0.857	0.824
MA 5D	2/39	7/11	0.385	0.863	0.822
EX 5D	0/39	8/11	0.297	0.930	0.875

Table 4.8 Univariate model results

4.8 Alternative approaches

This section and the next one were not part of the article submitted to the Mechanical Systems and Signal Processing journal. This section details other approaches we investigated for solving the predictive maintenance problem.

4.8.1 RUL prediction

When introducing the field of predictive maintenance in Chapter 2, we mentioned the possibility of predicting the remaining useful life of a machine, mathematically defined in equation (2.1). Even though predicting the RUL was not the primary goal in our use case, it could nevertheless have been a nice extra. However, we quickly found out that this is a challenging task, at least in our application use case. In Chapter 4, we tried to predict the RUL by directly learning a mapping between the inputs and the RUL via a support vector regression algorithm. However, it turned

out to be the worst-performing formulation. We concluded this was probably because there might be too many disparities between the life spans of different runs or simply because there were not enough failure samples. Altogether, a SVR was probably not the right choice for this prediction task (if we do not consider the second level in our PdM framework) as the algorithm is completely unaware of time and relationships between successive samples. That is why we also tried a few alternative models specifically designed to handle time series. A constraint that we absolutely wanted to satisfy was to include some information about the uncertainty of our prediction on top of the actual RUL prediction. Therefore we decided to try statistical models that can quantify the uncertainty in the prediction by design. The idea was to extrapolate future values of a health indicator up to reaching a predefined threshold which would give the predicted RUL. We first tried to fit an exponential model on different time windows on the health indicator of the form

$$h(t) = \phi + \theta(t) \exp(\beta(t)t).$$

We also used a Kalman filter (KF) where the HI is approximated as a linear or quadratic function. Finally, we also tried non-linear versions of the Kalman filter, i.e. extended KF and unscented KF, where we parameterized the HI as an exponential function. We evaluated the performance by computing a RUL estimate at each time step and comparing this estimate with the true RUL. For all runs, we did not obtain satisfying results. Indeed, the true RUL was almost never contained within the 95% confidence interval around the estimate. We also evaluated the correctness of the prediction two days before failure. Even in this setting, 60% of the estimates failed to predict the true RUL within their confidence interval for the best model. The reason of the bad performance of those algorithms can be explained by the relatively non monotonic trend of the health indicators we try to estimate. Hence, we did not pursue in that direction.

4.8.2 Taking signal history into account

In Chapter 4, the signal inputs are aggregated over a particular time window in order to summarize the signal characteristics across time. However, the learning algorithm is only aware of the current time window since no mechanisms are used to remember and use previous inputs. In a preliminary work [HG19] (conducted before the work described in Chapter 4) with less run-to-failure data, we investigated the training of a binary classi-

fication algorithm with and without history-enriched features. The difference between the two approaches is that the history-enriched model is fed with multiple time windows (the current one and a certain number of previous windows) instead of just the current one for the non history-enriched version. We found that enriching the inputs with multiple previous time windows helped increase the performance. The history-enriched approach was, however, not applied to the models in Chapter 4, although this could potentially have improved their performance. Nevertheless, even though multiple time windows are fed to the model, a traditional learning algorithm such as a SVM is still unaware of time and how each time window is linked to one another.

4.8.3 Deep learning and recurrent neural networks

Instead of designing features by hand or selecting the right number and length of time windows, we can wonder whether using a deep learning architecture could perform that and the prediction task for us. Moreover, why not use specialized neural networks designed to handle time series which are aware of time, such as recurrent neural networks (RNN) and their variants? This was actually the work of a master thesis conducted in 2020 [MGH20]. In his master thesis, Mercurio tried to train various neural network architectures on the same case study with a binary classification approach. In particular, he tried RNN, Long short-term memory networks (LSTM), and Gated Recurrent Unit (GRU) networks. Even though those networks are designed to handle time series, they could not achieve the same performance obtained by more traditional ML methods. One possible explanation was the relatively low number of run-to-failure data available for training.

4.8.4 Application on another dataset

In another master thesis [dPGH22], the framework developed in Chapter 4 was applied to the prediction of Hard Drive Disk failures. The open-source real-world dataset [Bla22] used contained 35,000 runs and 95 features. de Patoul managed to get a better performance than existing methods described in the literature, which suggests that the framework developed in Chapter 4 is well suited for different types of predictive maintenance applications.

4.9 Future perspectives

A possible improvement to the predictive maintenance framework developed in this Chapter would be that the learning algorithm directly optimizes the final metric, i.e. the scoring function used in the second level. Indeed, currently, the learning algorithm optimizes the parameters of a model to fulfill a classification or regression task, i.e. separation between classes with maximal margin in the case of a SVM. What we suggest here would be to optimize the parameters of a new learning algorithm directly with respect to the metric of the second level, i.e. equation (4.22). However, it is not trivial to formulate this as an optimization problem. Indeed, the metric to optimize is defined based on the outcome of a run (whether an alarm was raised or not for the F_{score} and when the alarm was raised for the business score) rather than based on each sample or time step. To overcome this issue, we could formulate the problem as a stochastic process where the model has a probability of triggering an alarm at each time step. If we simplify the problem where we only take into account corrective maintenance, use the business score defined in Figure 4.5.1 as the metric to optimize and use a simple linear model, the optimization problem could be formulated as follows:

$$\max_{\mathbf{w}} \frac{1}{C} \sum_{r \in C} \sum_{t \in T_r} p^{(r)}(t) \cdot \text{metric}(t) - \lambda R(\mathbf{w}) \tag{4.23}$$

$$\text{s.t. } p^{(r)}(t) = S^{(r)}(t) \cdot \prod_{k<t} (1 - S^{(r)}(k)) \tag{4.24}$$

$$S^{(r)}(t) = \frac{1}{1 + \exp(-\mathbf{w} \cdot \mathbf{F}^{(r)}(t))} \tag{4.25}$$

where

- $f = \mathbf{w} \cdot \mathbf{F}^{(r)}(t)$ is a linear predictor, with $\mathbf{w}$ the weights to optimize and $\mathbf{F}^{(r)}(t)$ the data at time t.
- $S^{(r)}(t)$ is a sigmoid function mapping the predictor output to a range between zero and one. It is a probability to trigger an alarm at each time step.
- $p^{(r)}(t)$ is a probability to have the first trigger at time t.
- The objective function is the expectation of the metric according to the probability of the first trigger.

- $R(\mathbf{w})$ a possible regularization function and λ a constant.

Instead of using a linear predictor, we can of course use more complex functions and kernels. Unfortunately, that optimization problem is not convex, but so are almost all deep learning objective functions nowadays. This optimization problem can be formulated as a recurrent neural network, and we could solve this problem using stochastic gradient descent and use the many heuristics and tricks used in deep learning to find a good local minimum. We can also optimize the learning algorithm according to the F_{score} metric defined in equation (4.21), although it is formulated a little differently. Mathematically, the problem can be formulated as follows:

$$\max_{\mathbf{w}} \frac{\left(1 - \frac{1}{P}\sum_{r\in P}\sum_{t\in T_r} p^{(r)}(t)\cdot 1\right)\left(\frac{1}{C}\sum_{r\in C}\sum_{t\in T_r} p^{(r)}(t)\cdot 1\right)}{\left(1 - \frac{1}{P}\sum_{r\in P}\sum_{t\in T_r} p^{(r)}(t)\cdot 1\right) + \frac{1}{C}\sum_{r\in C}\sum_{t\in T_r} p^{(r)}(t)\cdot 1} - \lambda R(\mathbf{w}) \tag{4.26}$$

$$\text{s.t. } p^{(r)}(t) = S^{(r)}(t)\cdot\prod_{k<t}(1 - S^{(r)}(k)) \tag{4.27}$$

$$S^{(r)}(t) = \frac{1}{1+\exp(-\mathbf{w}\cdot\mathbf{F}^{(r)}(t))} \tag{4.28}$$

Finally objective functions (4.23) and (4.26) can also be combined into a single objective function as in equation (4.22).

This approach could not be tested during the time period devoted to this part of the thesis. We believe it may offer a novel and promising point of view for predictive maintenance, and leave its exploration for future research.

PART II
Calibration of a proton therapy beamline using derivative-free optimization

5

Background in derivative-free optimization

In many optimization problems, we do not have access to an analytical expression for the objective function, nor do we have access to its gradient, but instead we only have access to (possibly noisy) evaluations of this objective at chosen sampling points. In this case, we refer to the objective function f as a black-box function.

If the function f is cheap to evaluate, we could sample many points via brute-force grid search methods, or evaluate the gradient numerically with finite differences. However, there are many examples where this is not feasible or where we cannot afford too many evaluations of the objective function. For example, this is the case for tuning hyperparameters in a machine learning problem [SLA12]. Another example is reservoir engineering, where we want to minimize the number of drills at different geographical locations in order to find petroleum wells [ARC+12] (one function evaluation corresponds to physical drilling). Noisy evaluation of the function also typically prevents the use of finite differences (due to huge loss of accuracy). Our application involves calibrating the beamline of a proton therapy system, for which a derivative-free optimization approach would help limit the number of function evaluations required by a gradient-based method for computing the gradient at each step, therefore

speeding up the calibration process.

In this chapter, we are interested in solving the optimization problem

$$\min_{\mathbf{x}\in A} f(\mathbf{x}) \tag{5.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a black-box function and $A = \{\mathbf{x} \in \mathbb{R}^n : a_i \leq x_i \leq b_i\}$ is a bounded set.

5.1 Derivative-free algorithms in a nutshell

Derivative-free optimization, also called *gradient-free* or *black-box optimization*, has been used for many years. One of the first derivative-free algorithms was developed by Nelder and Mead in the 1960s, sometimes called the Nelder-Mead simplex algorithm, is still in use today. It only requires sampling the objective functions at a finite number of points and immediately decides what actions to do next based on the concept of a simplex and geometric operations upon it. The Nelder-Mead algorithm falls into the direct search methods category, which also includes the coordinate-search method and other directional direct-search methods. Another category of derivative-free approaches consists of model-based methods that use a surrogate of the objective function to guide the search process. A common category of algorithms used in the model-based approach is trust-region methods, which approximate a subset region of the objective function close to the current solution, called trust region, by a simpler model and on which appropriate decrease directions can be extracted. The reader can refer to [LMW19] for a recent review of these methods, and for a more exhaustive yet older reference, the reader can refer to the book of Scheinberg and Vicente [CSV09]. Derivative-free methods can be further categorized as *local* or *global*. The methods we just briefly mentioned are referred to as local optimization methods because, as the name suggests, they aim at finding a local minimum (in the case of minimization) in a local region around an initial value. Global optimization, on the other hand, aims at finding the global minimum on the entire domain on which the function is defined. The latter approach obviously raises many more mathematical and computational challenges. Global optimization in the context of derivative-free optimization includes multi-level coordinate search, response surface methods, branch-and-bound search, simulated annealing, genetic algorithms, particle swarm optimization, and much more. It is im-

portant to realize that only a subset of those methods will typically guarantee a global minimizer. Those methods are reviewed by Rios and Sahinidis in [RS13]. Finally, algorithms can be classified as *deterministic* or *stochastic* based on whether or not they use a random search step.

Bayesian optimization is a model-based global optimization method that emerged relatively recently as a powerful solution for black-box function optimization [SSW+15]. It has been widely used in the machine learning community to tune deep neural networks hyperparameters [BBBK11] but has also been applied to many other problems such as robotics [LWB+07], experimental design [GRG+20], and reinforcement learning [BCDF10]. The method is based on Gaussian processes and relies on a Bayesian posterior update.

This part of the thesis focuses on Bayesian optimization and the Nelder-Mead algorithm to solve a beamline calibration problem. In Section 5.2, we describe the Nelder-Mead algorithm, and in Section 5.3, we explain how Gaussian process regression and Bayesian optimization work. We also apply this framework to synthetic examples for illustration purposes.

5.2 Nelder-Mead algorithm

This section uses [SN09] as the primary reference for describing the Nelder Mead algorithm and reuses a modified version of the figures.

The Nelder-Mead algorithm, originally published in 1965 by Nelder and Mead [NM65], is a direct unconstrained optimization search method that does not require information on the derivatives of the function to be optimized. It is a widely used optimization algorithm due to its ease of use and suitability for optimizing black-box functions, non-smooth functions, or even discontinuous functions.

The algorithm only uses function values to guide the optimization, hence the classification as a direct search method. The algorithm uses the concept of a simplex that can move, expand or contract at each iteration to decrease the function values at its vertices. A simplex $\mathcal{S} \in \mathbb{R}^n$ is defined as a polytope of $n+1$ vertices $\mathbf{x}_0, ..., \mathbf{x}_n \in \mathbb{R}^n$. For example, in 2-dimension, the simplex is defined as a triangle, and in 3-dimension, the simplex is defined as a tetrahedron (see Fig. 5.1).

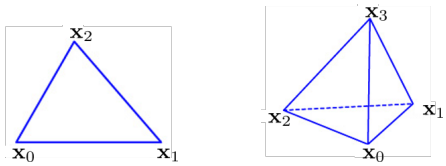


Fig. 5.1 Examples of simplex in 2D and 3D.

The method begins with an initial simplex composed of $n+1$ vertices and their corresponding function values $f_j = f(\mathbf{x}_j)$ for $j = 0, ..., n$. Then a series of geometric transformations are applied to the simplex aiming to decrease the function values at the vertices. When the simplex is sufficiently small, or the function values f_j are sufficiently close, the optimization is terminated.

5.2.1 Description of the algorithm

The general algorithm works as follows:

- Create the initial simplex $\mathcal{S}$.
- Transform the current simplex while termination test not satisfied
- Return the best vertex of the current simplex $\mathcal{S}$.

Initial simplex Choosing the initial simplex is a non-trivial problem in itself. A simplex too small would lead to a more localized search, while a simplex too big might take longer to converge. In the original article [NM65], the authors suggest to choose an initial point $\mathbf{x}_0 \in \mathbb{R}^n$ and generate the other vertices using a fixed step size along each dimension, i.e.

$$\mathbf{x}_j = \mathbf{x}_0 + s_j \mathbf{u}_j$$

with $\mathbf{u}_j \in \mathbb{R}^n$ is a unit vector and s_j the step size along $\mathbf{u}_j$.

Simplex geometric transformation An iteration of the algorithm consists of three steps

1. **Ordering:** Order the vertices of the current simplex $\mathcal{S}$ in terms of their function values in increasing order, i.e. $f_0 < f_1 < ... < f_n$. The

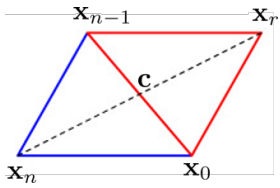
worst, second-worst and best vertex are then used in the subsequent computations.

2. **Centroid computation:** Computes the centroid of all vertices excluding the worst vertex $\mathbf{x}_n$:
$$\mathbf{c} := \frac{1}{n} \sum_{j \neq n} \mathbf{x}_j$$

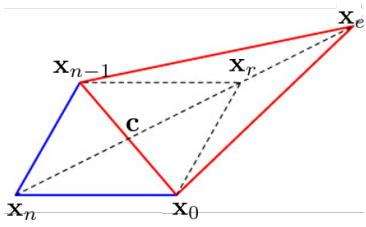
3. **Geometric transformation:** Apply a geometric transformation to the current simplex in order to decrease the current worst value f_n. The four allowed transformation are reflection, expansion, contraction or shrink.

The four geometric transformations (step 3 of the algorithm) are described below and illustrated in a 2-dimension setting (in blue the original simplex and in red the transformed simplex):

Reflection Compute the reflection point $\mathbf{x}_r = \mathbf{c} + \alpha(\mathbf{c} - \mathbf{x}_n)$ with $\alpha > 0$ and its function value $f_r = f(\mathbf{x}_r)$. If $f_0 \leq f_r < f_{n-1}$, construct a new simplex by replacing $\mathbf{x}_n$ with $\mathbf{x}_r$ and go to the next iteration.

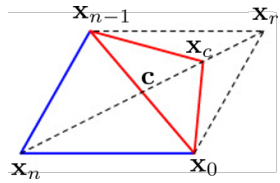


Expansion If $f_r < f_0$, computes the expansion point $\mathbf{x}_e := \mathbf{c} + \gamma(\mathbf{x}_r - \mathbf{c})$ with $\gamma > \alpha$ and its function value $f_e = f(\mathbf{x}_e)$. If $f_e < f_r$, construct a new simplex by replacing $\mathbf{x}_n$ with $\mathbf{x}_e$ and go to the next iteration. Otherwise (if $f_e \geq f_r$), construct a new simplex by replacing $\mathbf{x}_n$ with $\mathbf{x}_r$ and go to the next iteration.

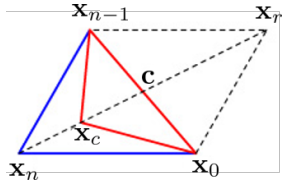


Contraction If $f_r \geq f_{n-1}$, computes the contraction point $\mathbf{x}_c$. There are two types of contraction:

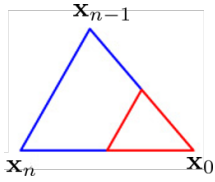
Outside contraction If $f_{n-1} \leq f_r < f_n$, computes $\mathbf{x}_c := \mathbf{c} + \beta(\mathbf{x}_r - \mathbf{c})$ with $0 < \beta < 1$ and its function value $f_c = f(\mathbf{x}_c)$. If $f_c \leq f_r$, construct a new simplex by replacing $\mathbf{x}_n$ with $\mathbf{x}_c$ and go to the next iteration. Otherwise, perform a shrinkage transformation.



Inside contraction If $f_r \geq f_n$, computes $\mathbf{x}_c := \mathbf{c} + \beta(\mathbf{x}_n - \mathbf{c})$ with $0 < \beta < 1$ and its function value $f_c = f(\mathbf{x}_c)$. If $f_c < f_n$, construct a new simplex by replacing $\mathbf{x}_n$ with $\mathbf{x}_c$ and go to the next iteration. Otherwise, perform a shrinkage transformation.



Shrinkage Replace all vertices except $\mathbf{x}_0$ with n new vertices $\mathbf{x}_j := \delta(\mathbf{x}_j - \mathbf{x}_0)$ with $0 < \delta < 1$ and their function values $f_j = f(\mathbf{x}_j)$ for $j = 1, ..., n$.



Four parameters related to the geometric transformation need to be chosen: α for reflection, β for contraction, γ for expansion and δ for shrinkage. The values used in standard implementations (also used in this thesis) are

$$\alpha = 1, \quad \beta = \frac{1}{2}, \quad \gamma = 2, \quad \delta = \frac{1}{2}$$

Note that there are very little theoretical guarantees about the algorithm. Hence, it is essentially a heuristic that is observed to often perform well in practise.

5.3 Bayesian optimization

This section uses the book of Rasmussen and Williams on Gaussian processes [WR06] and the book of Murphy [Mur12] as the primary resource, as well as the work of Krasser for the code used to generate the figures [Kra21].

Bayesian optimization is a global optimization approach to solve black-box optimization problems. It is best suited for optimization problems of moderate dimension and can handle stochastic noise in function evaluations [Fra18]. The method works in two phases. First, a surrogate model of the objective function is built via Gaussian process regression, which also quantifies the uncertainty in that surrogate by design. Then, the method suggests the next point to evaluate via the maximization of an acquisition function.

In the following two sections, we describe how Gaussian process regression works and how Bayesian optimization is conducted via the choice of an acquisition function.

5.3.1 Gaussian process regression

Probabilistic regression is usually formulated as follows: given a training data set $D = \{(\mathbf{x}_i, y_i) \in \mathbb{R}^d \times \mathbb{R}, i = 1, ..., N\}$ of N input $\mathbf{x}_i$ and possibly noisy outputs y_i, compute the prediction y_* at a new point $\mathbf{x}_*$. Generally, we assume that we have an additive i.i.d. Gaussian noise on the function values, i.e.

$$y_i = f(\mathbf{x}_i) + \epsilon_i, \qquad \epsilon_i \sim \mathcal{N}(0, \sigma^2)$$

where σ^2 is the variance of the noise.

In most cases, we assume that the function f admits a parametric representation to model the data: $p(y|\mathbf{x}, \boldsymbol{\theta})$; where the parameters $\boldsymbol{\theta}$ can be determined via maximum likelihood estimation. On the other hand, non-parametric methods do not assume a specific parametric model. The k-nearest neighbor algorithm, kernel regression, or regression trees are examples of non-parametric methods.

Gaussian process regression is a special type of non-parametric method that directly infers a distribution over functions. The difference with usual probabilistic regression is that we infer $p(f|D)$ for some function f instead of inferring $p(\boldsymbol{\theta}|D)$ for some parametric function with parameters $\boldsymbol{\theta}$. A Gaussian process defines a prior over functions:

$$f(\mathbf{x}) \sim GP(m(\mathbf{x}), \kappa(\mathbf{x}, \mathbf{x}')), \tag{5.2}$$

where $m(\mathbf{x})$ is the mean function and $\kappa(\mathbf{x}, \mathbf{x}')$ is the covariance or kernel function. The kernel function sets the prior information on the distribution and the shape of the function. When observing a pair of points in D, the GP can be converted to a posterior over functions. Even though it seems hard to compute a distribution over functions, it turns out that we only need to compute a distribution over function values at a finite but arbitrary set of points $\mathbf{x}_i$ [Mur12]. Formally, a Gaussian process is a random process where data points $\{\mathbf{x}_i\}_{i=1,...,N}$ are assigned a random variable $f(\mathbf{x}_i)$ and where the joint distribution of those variables $p(f(\mathbf{x}_1), ..., f(\mathbf{x}_N))$ is also Gaussian. That is

$$p(\mathbf{f}|\mathbf{x}_1, ..., \mathbf{x}_N) = \mathcal{N}(\boldsymbol{\mu}, \mathbf{K}), \tag{5.3}$$

where $\mathbf{f} = [f(\mathbf{x}_1), ..., f(\mathbf{x}_N)]^T$, $\boldsymbol{\mu} = [m(\mathbf{x}_1), ..., m(\mathbf{x}_N)]^T$ is the mean function and $\mathbf{K}$ is a covariance matrix with entries given by kernel function $K_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$. Note that it is common practice to use a GP with a zero mean function ($m(\mathbf{x}) = 0$ hence $\boldsymbol{\mu} = \mathbf{0}$) and is not necessarily a limitation, since the mean of the posterior process is not confined to be zero [WR06]. Therefore, we use a zero mean function in the rest of this work.

The GP can then be used to predict new values $\mathbf{f}_*$ at new inputs $\mathbf{x}_*$. Since the joint distribution of observed values $\mathbf{f}$ and predictions $\mathbf{f}_*$ is Gaussian by definition of the GP, we can express the joint distribution as

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} & \mathbf{K}_* \\ \mathbf{K}_*^T & \mathbf{K}_{**} \end{bmatrix}\right), \tag{5.4}$$

where $\mathbf{K}_* = \kappa(\mathbf{x}, \mathbf{x}_*)$ is a $N \times N_*$ matrix and $\mathbf{K}_{**} = \kappa(\mathbf{x}_*, \mathbf{x}_*)$ is a $N_* \times N_*$ matrix. Using rules for conditional probability on Gaussian distribution (see [WR06], section A.2 for details), we can compute the posterior distri-

bution:

$$p(\mathbf{f}_*|\mathbf{X}_*, \mathbf{X}, \mathbf{f}) = \mathcal{N}(\mathbf{f}_*|\boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*) \tag{5.5}$$

$$\boldsymbol{\mu}_* = \mathbf{K}_*^T \mathbf{K}^{-1} \mathbf{f} \tag{5.6}$$

$$\boldsymbol{\Sigma}_* = \mathbf{K}_{**} - \mathbf{K}_*^T \mathbf{K}^{-1} \mathbf{K}_* \tag{5.7}$$

Kernel function The key idea of a kernel function is that if data points $\mathbf{x}$ and $\mathbf{x}'$ are considered similar by the kernel, we expect the output at those points to be similar. Moreover, kernel functions have to be positive definite functions.

The most commonly used kernel is the Gaussian kernel, also called squared exponential or RBF kernel:

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 \exp\left(-\frac{1}{2l^2}(\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j)\right) \tag{5.8}$$

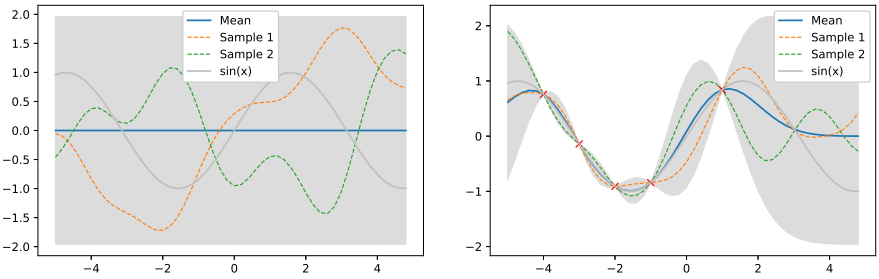
with σ_f^2 a parameter for the signal variance and l a length-scale parameter. The Gaussian kernel results in a smooth function because the kernel is infinitely differentiable.

Another commonly used kernel is the *Matèrn* kernel, which is a generalization of the Gaussian kernel to relax the infinitely differentiable property of the Gaussian kernel. It was defined by Stein as a replacement for the Gaussian kernel, which yields unrealistic results for physical processes due to the infinite differentiability property [Ste99]. It is expressed as

$$\kappa(\mathbf{x}_i, \mathbf{x}_j; \nu) = \sigma_f^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu}\frac{\|\mathbf{x}_i - \mathbf{x}_j\|}{l}\right)^{\nu} K_\nu \left(\sqrt{2\nu}\frac{\|\mathbf{x}_i - \mathbf{x}_j\|}{l}\right), \tag{5.9}$$

where l is a length-scale parameter, K_ν is a modified Bessel function, $\Gamma(\cdot)$ is the gamma function and ν a parameter that controls smoothness of the resulting function. As $\nu \to \infty$ the Matérn kernel converges to the RBF kernel. When $\nu = 1/2$, the Matèrn kernel becomes the absolute exponential kernel

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{1}{l}\|\mathbf{x}_i - \mathbf{x}_j\|\right) \tag{5.10}$$



(a) GP prior **(b)** GP posterior

Fig. 5.2 1D Gaussian process regression of function $f(x) = \sin(x)$. The dotted lines represent functions drawn from the GP. The solid blue line represents the mean function, and the light grey filling represents a 95% confidence interval around the function value. The red cross in figure (b) represents the observed values.

Popular choices for ν are $\nu = 3/2$ which is a once-differentiable kernel function

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \left(1 + \frac{\sqrt{3}}{l}\|\mathbf{x}_i - \mathbf{x}_j\|\right) \exp\left(-\frac{\sqrt{3}}{l}\|\mathbf{x}_i - \mathbf{x}_j\|\right), \tag{5.11}$$

or $\nu = 5/2$ for a twice-differentiable kernel function:

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \left(1 + \frac{\sqrt{5}}{l}\|\mathbf{x}_i - \mathbf{x}_j\| + \frac{5}{3l}\|\mathbf{x}_i - \mathbf{x}_j\|^2\right) \exp\left(-\frac{\sqrt{5}}{l}\|\mathbf{x}_i - \mathbf{x}_j\|\right) \tag{5.12}$$

1D example of a Gaussian process Let us show an example of a 1D Gaussian process with a Gaussian kernel. In this case, we suppose that $\sigma_f^2 = 1$ and $l = 1$ in the equation of the Gaussian kernel (5.8) and that the mean is zero. Three sample functions drawn from the GP prior are shown in Fig. 5.2a as well as the mean and a 95% confidence interval, which is defined as $\boldsymbol{\mu} \pm 1.96\sqrt{\text{diag}(\mathbf{K})}$.

Let us try to regress the function $f(x) = \sin(x)$ from 5 observed values at $\mathbf{X} = [-4, -3, -2, -1, 1]^T$. After computing the posterior with equations (5.6-5.7), we plot three sample functions drawn from the GP posterior as

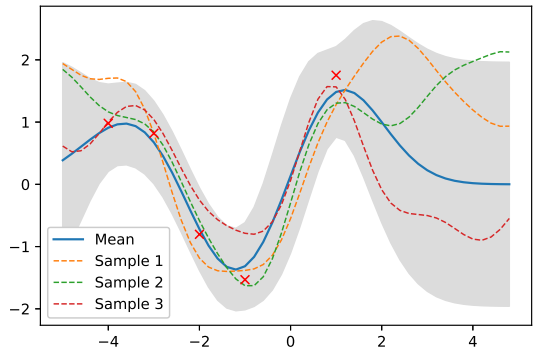


Fig. 5.3 1D GP regression of $f(x) = \sin(x)$ of noisy observations with variance $\sigma_n^2 = 0.4$.

well as the mean and a 95% confidence interval, as shown in Fig. 5.2b. The GP is able to approximate quite well the function $\sin(x)$ in the interval $[-4, 1]$ as shown in Fig. 5.2b. The uncertainty is also relatively small in the interval $[-4, -1]$ as there exist several observed values and a little more uncertain in the interval $[-1, 1]$ as we only have two observed values. However, in the interval $[1, 5]$, the uncertainty grows as the GP does not have enough observed values. The mean value is therefore affected as well.

Prediction from noisy observations In a more realistic situation, it is possible that we do not have access to the true function values but only noisy evaluations, i.e. $\mathbf{u} = \mathbf{f}(\mathbf{x}) + \boldsymbol{\epsilon}$ with $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_n^2 \mathbf{I})$. If we assume that the noise is an i.i.d. Gaussian noise with variance σ_n^2, we can take it into account in our GP. Let us define $\mathbf{K}_n = \mathbf{K} + \sigma_n^2 \mathbf{I}$. We can now rewrite equations (5.5-5.7) as

$$p(\mathbf{f}_*|\mathbf{X}_*, \mathbf{X}, \mathbf{f}) = \mathcal{N}(\mathbf{f}_*|\boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*) \tag{5.13}$$

$$\boldsymbol{\mu}_* = \mathbf{K}_*^T \mathbf{K}_n^{-1} \mathbf{f} \tag{5.14}$$

$$\boldsymbol{\Sigma}_* = \mathbf{K}_{**} - \mathbf{K}_*^T \mathbf{K}_n^{-1} \mathbf{K}_* \tag{5.15}$$

Taking back the example of regressing the function $\sin(x)$ shown in Fig. 5.2b, we compute the posterior distribution assuming that the observed

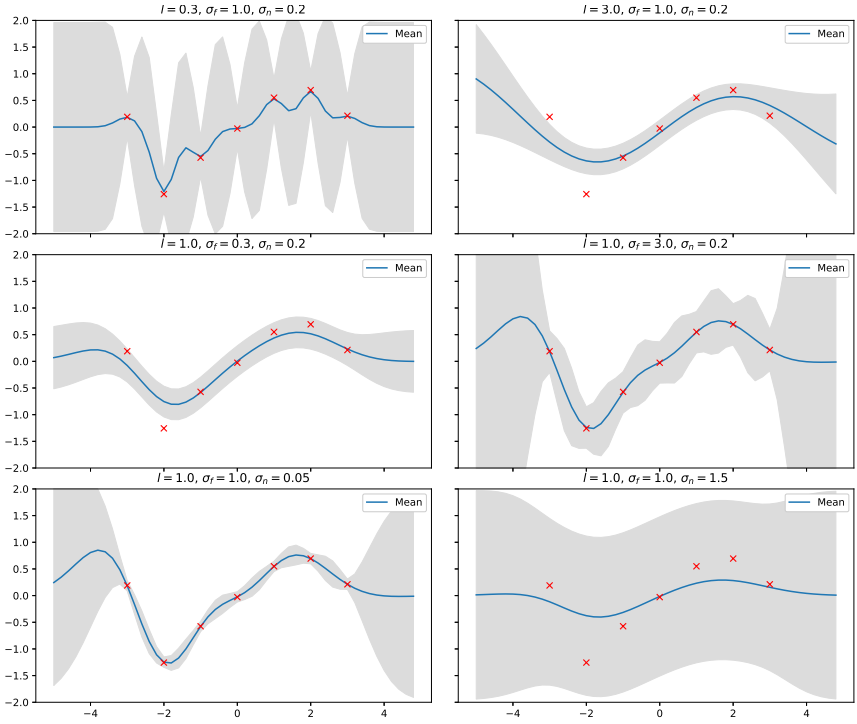


Fig. 5.4 Impact of the variations of hyperparameters of the kernel and noise variance. Each row represents a variation of one component, the other remaining fixed. On row 1, parameter l vary from 0.3 to 3. On row 2, parameter σ_f vary from 0.3 to 3. On row 3, noise standard deviation σ_n vary from 0.05 to 1.5.

values are noisy with equations (5.14-5.15). In Fig. 5.3, we show the results of the regression with a noise variance $\sigma_n^2 = 0.4$. There is now an uncertainty around the observed values as expected. Moreover, the mean function does not necessarily pass through those observations anymore.

Tuning hyperparameters Until now, we assumed that the kernel parameters (σ_f^2, l) were fixed. In fact, they strongly impact the outcome of the posterior distribution. Let us vary those parameters and the noise variance and study their impact in Fig. 5.4. For this example, we gather observations at equally spaced input in the range $[-3, 4]$ with step 1 and add an i.i.d. Gaussian noise of variance $\sigma_n^2 = 0.4$.

Lower value of length-scale parameter l leads to sharp transitions of the posterior and high uncertainty in between observations, while a higher value leads to a smoother approximation.

Low values of σ_f restrict the amplitude variations of the posterior, while a large value of σ_f allows more frequent variations.

Low values of the noise variance lead to low uncertainty at the observations and in between observations, which could lead to overfitting. In contrast, large values result in a bad fit of the observations.

Now that we have seen that those parameters strongly impact the predictions, the question becomes how can we choose them optimally? It turns out that optimal values for these parameters can be derived by maximizing the log marginal likelihood[1] [WR06] which is expressed as

$$\log p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}) = \log \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K}_n, \boldsymbol{\theta}) \tag{5.16}$$

$$= -\frac{1}{2}\mathbf{y}^T\mathbf{K}_n^{-1}(\boldsymbol{\theta})\mathbf{y} - \frac{1}{2}\log\left|\mathbf{K}_n(\boldsymbol{\theta})\right| - \frac{N}{2}\log(2\pi) \tag{5.17}$$

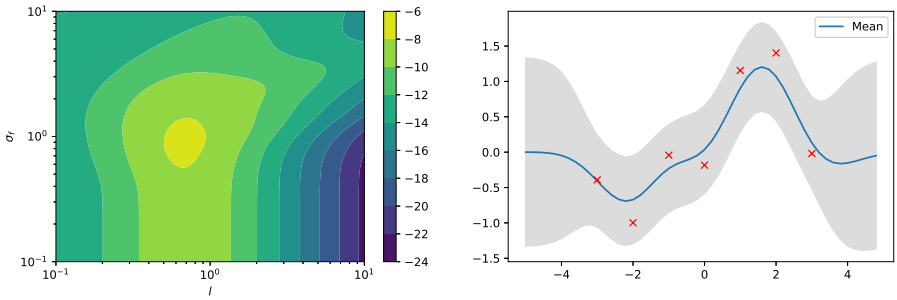
To obtain the optimal hyperparameters we compute the partial derivatives of the marginal log likelihood with respect to the hyperparameters as follows (see [WR06], chapter 5 for details):

$$\frac{\partial}{\partial\theta_j}\log p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^T\mathbf{K}_n^{-1}\frac{\partial\mathbf{K}_n}{\partial\theta_j}\mathbf{K}_n^{-1}\mathbf{y} - \frac{1}{2}\operatorname{Tr}\left(\mathbf{K}_n^{-1}\frac{\partial\mathbf{K}_n}{\partial\theta_j}\right) \tag{5.18}$$

$$= \frac{1}{2}\operatorname{Tr}\left(\left(\boldsymbol{\alpha}\boldsymbol{\alpha}^T - \mathbf{K}_n^{-1}\right)\frac{\partial\mathbf{K}_n}{\partial\theta_j}\right) \tag{5.19}$$

where $\boldsymbol{\alpha} = \mathbf{K}_n^{-1}\mathbf{y}$. The complexity of computing the marginal log-likelihood in (5.17) is dominated by the kernel matrix inversion and requires $O(N^3)$ operations for a $N \times N$ matrix. Once $\mathbf{K}^{-1}$ is known, the computation of the derivative in (5.19) requires only $O(N^2)$ operations per hyperparameter. Thus the cost of computing the derivative is small, and using a gradient-based optimizer is advantageous [WR06]. However, the optimization problem is non-convex, which may result in local maxima, but is usually not much of a problem with only a few hyperparameters to opti-

[1] The reason it is called marginal likelihood is because we have marginalized out the latent Gaussian vector $\mathbf{f}$: $p(\mathbf{y}|\mathbf{x}) = \int p(\mathbf{y}|\mathbf{f}, \mathbf{x})p(\mathbf{f}|\mathbf{x})d\mathbf{f}$



(a) Marginal log-likelihood **(b)** GP posterior with optimal hyperparameters

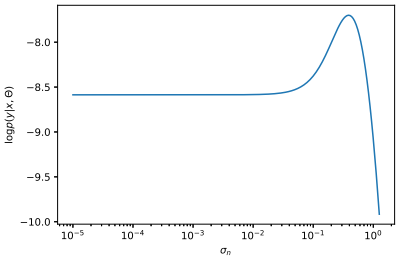
Fig. 5.5 Optimization of hyperparameters l and σ_f of the kernel function. On the left figure, we have the contour plot of the marginal log-likelihood as given in equation (5.17). On the right figure, the GP posterior is plotted with the optimal hyperparameters found from the maximization ($l = 0.89$,$\sigma_f = 0.68$).

mize. A common optimizer used for the optimization is the quasi-Newton L-BFGS-B algorithm [LN89].

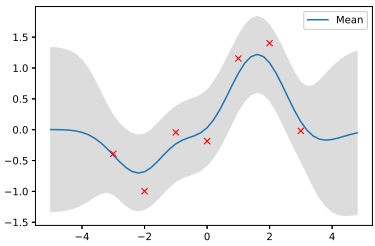
Let us now take back our noisy example $f(x) = \sin(x)$ and seek the optimal hyperparameters. We compute the optimal hyperparameters of the kernel function l and σ_f by minimizing the negative marginal log-likelihood with the L-BFGS-B algorithm and obtain $l = 0.89$ and $\sigma_f = 0.68$. The GP posterior with these hyperparameters is shown in Fig. 5.5b, and the values of the marginal log-likelihood as a function of the pair of hyperparameters are shown in Fig. 5.5a. We observe that this solution is simultaneously smooth and correctly fitting the data. Therefore, the parameters found with the optimization seem optimal, at least qualitatively.

In this case, there is only one local maximum, that the algorithm correctly finds it in a few iterations. However, in practice, equation (5.17) is non-convex, potentially leading to multiple local maxima. An optimization procedure with restarts from several different initial solutions is therefore recommended.

Note that in this case, we assumed that we knew the variance of the noise that generated the data in advance. However, this is often not the case, and it is actually possible to infer this noise variance in the same way we



(a) Marginal log-likelihood as a function of noise std



(b) GP posterior with optimal noise std

Fig. 5.6 Optimization of the noise standard deviation (std) σ_n. On the left figure, we have the plot of the marginal log-likelihood as given in equation (5.17) as a function of σ_n. On the right figure, the GP posterior is plotted with the optimal noise std found from the maximization ($\sigma_n = 0.39$).

inferred the kernel hyperparameters, i.e. by maximizing (5.17) with respect to σ_n.

Let us now suppose that we do not know the noise variance that generated the data. Let us try to infer this value thanks to equation (5.19) with $\theta_j = \sigma_n$. We will fix the kernel parameters to what we obtained above after the optimization, i.e. $l = 0.89$ and $\sigma_f = 0.68$. The value of the marginal log-likelihood for σ_n in a logarithmic range $[10^{-5}, 10^{0.1}]$ is shown in Fig. 5.6a. After maximization, we found a noise with standard deviation $\sigma_n = 0.39$, almost exactly the same as the one used to generate the data. Hence, inferring the noise value with the marginal log-likelihood is possible and works very well in this case. In Fig. 5.6b, the GP posterior is plotted with this noise standard deviation, resulting in essentially the same posterior as in Fig. 5.5b.

Implementation details The implementation of the GP regressor is based on algorithm 2.1 of the Gaussian Processes for Machine Learning (GPML) by Rasmussen and Williams [WR06]. The implementation is shown in Algorithm 2.

The algorithm computes the posterior distribution and the marginal log-likelihood and replaces the inversion of the kernel matrix in equation (5.17) with a Cholesky decomposition since it is faster and numerically more sta-

input : $\mathbf{X}$ (inputs), $\mathbf{y}$ (targets), κ (covariance function), $\boldsymbol{\theta}$ (kernel hyperparameters), $\sigma_{n_1}^2, \sigma_{n_2}^2$ (noise levels), $\mathbf{X}_*$ (test input)

1 $\mathbf{K} := \kappa(\mathbf{X}, \mathbf{X}; \boldsymbol{\theta})$, $\mathbf{K}_* := \kappa(\mathbf{X}, \mathbf{X}_*; \boldsymbol{\theta})$, $\mathbf{K}_{**} := \kappa(\mathbf{X}_*, \mathbf{X}_*; \boldsymbol{\theta})$
2 $L := \text{cholesky}(\mathbf{K}_n)$ with $\mathbf{K}_n$ given by equation (6.1)
3 $\boldsymbol{\alpha} := L^T \backslash (L \backslash \mathbf{y})$
4 $\bar{\mathbf{f}}_* := \mathbf{K}_*^T \boldsymbol{\alpha}$
5 $\mathbf{v} := L \backslash \mathbf{K}_*$
6 $\mathbb{V}[\mathbf{f}_*] := \mathbf{K}_{**} - \mathbf{v}^T \mathbf{v}$
7 $\log p(\mathbf{y}|\mathbf{X}) := -\frac{1}{2}\mathbf{y}^T \boldsymbol{\alpha} - \sum_i \log L_{ii} - \frac{N}{2} \log 2\pi$
8 **return** $\bar{\mathbf{f}}_*$ *(mean)*, $\mathbb{V}[\mathbf{f}_*]$ *(variance)*, $\log p(\mathbf{y}|\mathbf{X})$ *(log marginal likelihood)*

Algorithm 2: GP regression algorithm

ble. It returns the predictive mean and variance as well as the value of the marginal log-likelihood.

Algorithm 2 is then placed in an optimization loop where $\boldsymbol{\theta}$ and σ_n^2 are optimized. The Jacobian is computed via equation (5.19), and the problem is solved with the L-BFGS-B algorithm [LN89]. Since the optimization is non-convex, we use several restarts with different initial values for those hyperparameters to avoid getting stuck in a poor local maximum.

Since the hyperparameters can vary quite significantly, it is also a good idea to optimize those parameters in log space, i.e. we give to the optimizer the log values of the parameters $\log([\boldsymbol{\theta}, \sigma_n^2])$ and replace those by their exponential in Algorithm 2. For the targets $\mathbf{y}$, it is also a good idea to normalize them before the optimization (e.g. subtracting the mean and dividing by the standard deviation) to avoid high magnitude values for the hyperparameters. The details of the hyperparameter optimization are given in Algorithm 3.

5.3.2 Bayesian optimization

Now that we have established how Gaussian process regression works, it is time to investigate the optimization component. In Bayesian optimization, Gaussian process regression is used to build a surrogate model of the objective function. Then, an acquisition function is used to decide where to sample next according to that surrogate. The acquisition function trades off between exploration of regions with high uncertainty and exploitation of regions where it believes a minimum can be found (in the case of mini-

input : $\mathbf{X}$ (inputs), $\mathbf{y}$ (targets), κ (covariance function), $\mathbf{X}_*$ (test input), bounds (hyperparameters bounds), $n_{\text{restarts}} = 10$

bounds $= \log(\text{bounds})$
$\boldsymbol{\theta}_{\text{init}} \sim \mathcal{U}(bounds)$: Draw n_{restarts} points from uniform distribution over (log) search space.
$f_{best} = \infty$
for $\boldsymbol{\theta}_i$ *in* $\boldsymbol{\theta}_{init}$ **do**
  $f(\boldsymbol{\theta}) = -\log p(y|\mathbf{X}; \mathbf{exp}(\boldsymbol{\theta}_i))$ the negative log-likelihood obtained via Algorithm 2.
  $\mathbf{J} = -\nabla \log p(y|X; \mathbf{exp}(\boldsymbol{\theta}_i))$ from equation (5.19)
  $\boldsymbol{\theta}_{opt}, f_{opt} =$ L-BFGS-B$(f, \boldsymbol{\theta}_i, \mathbf{J})$ optimal $\boldsymbol{\theta}$ and function value obtained via the L-BFGS-B optimization algorithm.
  if $f_{opt} < f_{best}$ **then**
    $\boldsymbol{\theta}_{best} = \boldsymbol{\theta}_{opt}$
    $f_{best} = f_{opt}$
  end
end
return $\exp(\boldsymbol{\theta}_{best})$

Algorithm 3: GP regression hyperparameter optimization

mization). Both correspond to high acquisition function values. Formally, the next sample to evaluate $\mathbf{x}_{t+1}$ will result from the maximization of the acquisition function a :

$$\mathbf{x}_{t+1} = \arg\max_{\mathbf{x}} a(\mathbf{x}|D_{1:t}) \tag{5.20}$$

with $D_{1:t} = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_t, y_t)\}$, the dataset containing the t samples previously evaluated and used to build the surrogate model. Common acquisition functions used are the maximum probability of improvement (MPI), expected improvement (EI), and upper confidence bound (UCB). More recently, the mutual information (MI) acquisition function has proven very effective [CPV14]. Each of these acquisition functions is described below.

Maximum probability of improvement The idea is to maximize the probability of improving over the best current value [SLA12]. Suppose the current best value is $f_{\text{best}} = \min \mathbf{f}$, then MPI evaluates f at the points most likely to improve this value which is described by the utility function

$$u(\mathbf{x}) = \begin{cases} 1, & \text{if } f(\mathbf{x}) \leq f_{\text{best}} \\ 0, & \text{if } f(\mathbf{x}) > f_{\text{best}} \end{cases}$$

The acquisition function is the expected utility as a function of x:

$$\begin{aligned} a_{MPI}(\mathbf{x}) = \mathbb{E}(u(\mathbf{x})|\mathbf{x}, D) &= \int_{-\infty}^{f_{\text{best}}} \mathcal{N}(f|\mu(\mathbf{x}), \sigma^2(\mathbf{x}))df & (5.21)\\ &= \Phi(f_{\text{best}}; \mu(\mathbf{x}), \sigma^2(\mathbf{x})) & (5.22)\\ &= \Phi(\gamma(\mathbf{x})) & (5.23) \end{aligned}$$

with $\gamma(\mathbf{x}) = \frac{f_{\text{best}} - \mu(\mathbf{x})}{\sigma(\mathbf{x})}$, $\Phi(\cdot)$ is the cumulative distribution function of the standard normal, and $\mu(\mathbf{x})$ and $\sigma(\mathbf{x})$ are respectively the mean and standard deviation of the GP posterior distribution predicted at $\mathbf{x}$.

Expected improvement The idea is to maximize the expected improvement given over the best current value [Gar15]. Compared to the MPI, the EI accounts for the size of the improvement. EI evaluates f at the point that, in expectation, improves f the most which is described by the utility function

$$u(\mathbf{x}) = \max(0, f_{\text{best}} - f(\mathbf{x}))$$

The acquisition function is the expected utility as a function of x:

$$\begin{aligned} a_{EI}(\mathbf{x}) &= \mathbb{E}(u(\mathbf{x})|\mathbf{x}, D) & (5.24)\\ &= \int_{-\infty}^{f_{\text{best}}} (f_{\text{best}} - f)\mathcal{N}(f|\mu(\mathbf{x}), \sigma^2(\mathbf{x}))df & (5.25)\\ &= (f_{\text{best}} - \mu(\mathbf{x}))\Phi(\gamma(\mathbf{x})) + \sigma(\mathbf{x})\mathcal{N}(\gamma(\mathbf{x})) & (5.26) \end{aligned}$$

The two terms in equation (5.26) can be interpreted as an exploitation (evaluating at points with low mean) and exploration (evaluating at points with high variance) term, respectively. The trade-off is directly captured via the Bayesian decision theoretic treatment [Gar15].

Lower confidence bound (LCB) The lower confidence bound name typically describes a minimization process, i.e. min f rather than a maximization[2]. The idea is to look at the curve G, which is β standard deviations below the posterior mean of each point. This function takes the form

$$G(\mathbf{x}) = \mu(\mathbf{x}) - \beta\sigma(\mathbf{x}), \quad (5.27)$$

[2]The term upper confidence bound (UCB) is usually used instead in the case of maximization. However, since we developed the framework for minimizing function instead of maximizing, we will keep the name LCB.

where $\beta > 0$ is an explicit trade-off parameter between exploration and exploitation. $G(\mathbf{x})$ is the lower confidence envelope of the surrogate model. The acquisition function is defined as the negative of G, i.e. $a_{\text{LCB}}(\mathbf{x}) = -\mu(\mathbf{x}) + \beta\sigma(\mathbf{x})$ which we maximize to suggest the next point.

The acquisition function cannot be interpreted as an expected utility. However, theoretical results are known for which the acquisition function will converge to the global minimum under certain conditions [Gar15].

Mutual information (MI) The MI is an acquisition function based on the concept of mutual information in information theory. It was introduced more recently and has been shown to surpass the state-of-the-art LCB on several datasets [CPV14]. In the context of minimization, it can be formulated as

$$a_{\text{MI}}(\mathbf{x}) = \mu_t(\mathbf{x}) - \phi_t(\mathbf{x}) \tag{5.28}$$

with

$$\phi_t(\mathbf{x}) = \sqrt{\alpha}\left(\sqrt{\sigma_t^2(\mathbf{x}) + \hat{\gamma}_{t-1}} - \sqrt{\hat{\gamma}_{t-1}}\right) \tag{5.29}$$

and $\hat{\gamma}_t = \sum_{i=1}^{t} \sigma_i^2(\mathbf{x}_i)$.

The novel concept of this acquisition function is that ϕ_t is empirically controlled by the amount of exploration that has already been done; that is, the more the algorithm has gathered information on f, the more it will focus on the optimum [CPV14]. The parameter α controls the trade-off between precision and confidence.

The term mutual information comes from the fact that $\sum_{i=1}^{t} \sigma_i^2(\mathbf{x}) =: \hat{\gamma}_t(\mathbf{x}) \leq C\gamma_t(\mathbf{x})$ with $\gamma_t(\mathbf{x}) = \max_{\mathbf{x}_t} I_t(\mathbf{x}_t)$ and where $I_t(\mathbf{x}_t)$ is the mutual information between f and the noisy observations y_t at $\mathbf{x}_t$. For more information, you can refer to [CPV14].

1D example of Bayesian optimization In this section, we optimize a 1D function $f(x) = \cos(6x) + x^3 - x + \epsilon$ on the space $[-1, 1]$ with ϵ an additive Gaussian noise with standard deviation $\sigma = 0.2$. We do not assume we know the standard deviation of the noise (this will be inferred during the optimization of the GP). We use the most commonly used acquisition function, the expected improvement. For the Gaussian process, we will use the Matèrn kernel with parameter $\nu = 2.5$ (as defined in equation (5.12)),

which is also a conventional choice.

The results are displayed in Fig 5.7. We start with two initial noisy evaluations of the function. A Gaussian process builds a surrogate function (left figure) with the knowledge of those two data points. An acquisition function is plotted on the right to evaluate the best next candidate based on that surrogate function, which in this case is the point on the left boundary. After this new point is evaluated, a new surrogate function is built via the Gaussian process regressor. In this case, the new surrogate is somewhat nonsmooth and does not give us much information about the true function. This is because we do not yet have many data points to evaluate a good surrogate, and also because the non-convex optimization of the hyperparameters of the Gaussian process of equation 5.17 might be stuck in a local minimum or flat landscape. The next acquisition (iteration 3) allows to build a better surrogate estimate of the true function. From iterations 3-5, a local minimum is found around $x = -0.7$. Some more explorations are performed in iterations 6-7. Finally, from iterations 7-10, the global maximum is reached at $x = 0.5$.

Implementation details The implementation is shown in Algorithm 4. First n_{init} points are drawn from a uniform distribution bounded by *bounds*. Then, as long as do not reach the maximum number of function evaluations, we perform the followings steps:

1. Transform $\mathbf{X}$ coordinates in a 0-1 scale to avoid different scales between inputs (line 5)
2. Transform function evaluations $\mathbf{y}$ in a 0-1 scale to avoid high/low magnitude parameters in the Gaussian process (line 6)
3. Since the acquisition function is non-convex, performing one optimization is insufficient. However, since computing the posterior and evaluating the acquisition function is cheap, we can evaluate the (acquisition) function at many points to find suitable initial points to start the optimization. We choose to sample 1000 points. (line 7-10)
4. Choose the five best candidates among the 1000 points and start five optimization routines with those initial guesses. (line 11)
5. The optimization is performed via the L-BFGS-B algorithm (lines 12-20) and suggests the best next candidate to evaluate (lines 21-22).
6. The process restarts with new training points (lines 23-24)

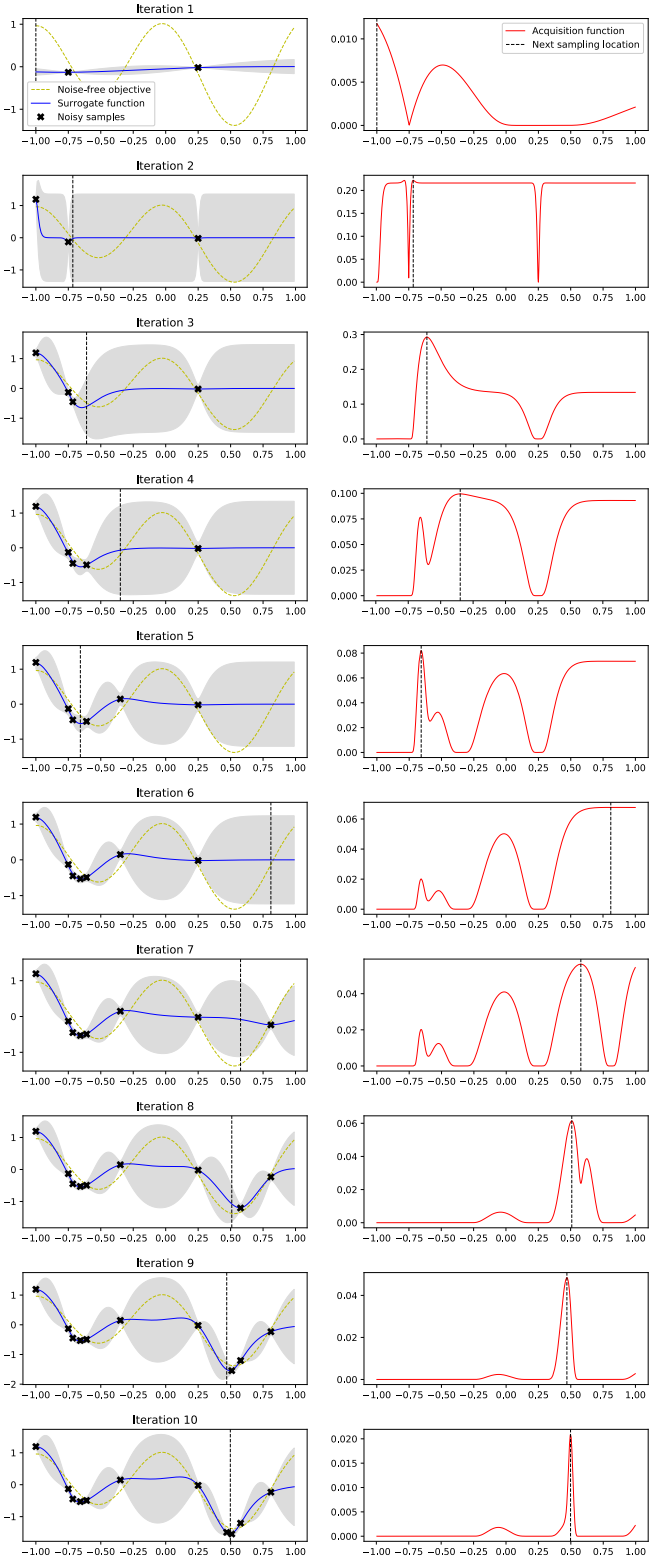


Fig. 5.7 Example of Bayesian optimization on a 1D function

input : f (objective function), Bounds, κ (covariance function), n_{init} (number of initial evaluations), a (acquisition function), N (number of function evaluations)

$\mathbf{X}_{\text{init}} \sim \mathcal{U}(bounds, n_{\text{init}})$: Draw n_{init} points from uniform distribution over search space.
$\mathbf{Y}_{\text{init}} = f(\mathbf{X}_{\text{init}})$
$\mathbf{X}_{\text{train}}, \mathbf{Y}_{\text{train}} = \mathbf{X}_{\text{init}}, \mathbf{Y}_{\text{init}}$
while $size(\mathbf{X}) < N$ **do**
    $\mathbf{X}_{\text{train}} = 01\_transform(\mathbf{X}_{\text{init}})$
    $\mathbf{Y}_{\text{train}} = 01\_transform(\mathbf{Y}_{\text{init}})$
    $\mathbf{x}_{\text{test}} \sim \mathcal{U}(0, 1, 1000)$: draw 1000 points from uniform distribution
    $\boldsymbol{\theta} =$ Algorithm 3.
    $\boldsymbol{\mu}, \boldsymbol{\sigma}, p = GP(\mathbf{x}_{\text{test}}, \mathbf{X}_{\text{train}}, \mathbf{Y}_{\text{train}}; \kappa, \boldsymbol{\theta})$ as given by algorithm 2.
    $\mathbf{v} = a(\boldsymbol{\mu}, \boldsymbol{\sigma})$: evalute the acquisition function at those 1000 points.
    $\mathbf{v}_{best} = \text{sort}(\mathbf{v})[5:]$: take the 5 best values obtained on those 1000 points.
    for $\mathbf{z}$ *in* $\mathbf{v}_{best}$ **do**
        $g(\mathbf{z}) = -a(GP(\mathbf{z}))$ the negative acquisition function evaluated at the GP (posterior) at $\mathbf{z}$ that we minimize.
        $\mathbf{J} = -\nabla_{\mathbf{z}} a$, gradient of the acquisition function
        $\mathbf{z}_{opt}, g_{opt} = \text{L-BFGS-B}(g, \mathbf{z}, \mathbf{J})$: optimal $\mathbf{z}$ and function value obtained via the L-BFGS-B optimization algorithm.
        if $g_{opt} < g_{best}$ **then**
            $\mathbf{z}_{best} = \mathbf{z}_{opt}$
            $g_{best} = g_{opt}$
        end
    end
    next_x $= 01\_inverse\_transform(\mathbf{z}_{best})$
    next_x $= f(\text{next\_x})$
    $\mathbf{X}_{\mathbf{train}} = [01\_inverse\_transform(\mathbf{X}_{\mathbf{train}}), \text{next\_x}]$
    $\mathbf{Y}_{\mathbf{train}} = [01\_inverse\_transform(\mathbf{X}_{\mathbf{train}}), \text{next\_x}]$
end

Algorithm 4: Bayesian optimization algorithm

6

Calibration of a proton therapy beamline

This chapter is an extended version of a conference paper published in peer-reviewed conference proceedings [HG21a].

6.1 Introduction

This chapter aims to apply optimization techniques, in particular derivative-free methods, to the automatic calibration of a proton therapy beamline. As a reminder from Chapter 1, the beamline allows transporting the beam of protons produced by the particle accelerator to the treatment room via an array of successive magnets. Those magnets, which are actually eletromagnets, tunable via their currents, need to be calibrated in order to obtain a treatment beam that respects a set of constraints. Finding the current set-points of those magnets for many different configurations and in the least amount of time is the goal of this chapter. On top of applying derivative-free methods to the calibration of the beamline, namely the Nelder-Mead algorithm and Bayesian optimization, we also propose a transfer learning approach based on Bayesian Optimization that reuses information from a previous configuration in order to speed up subsequent optimizations. Indeed, it turns out that the calibration procedure has to be repeated for

different configurations of the beamline, with the problem data changing moderately yet significantly enough to require a new optimization procedure. Our approach involves learning the noise variance to apply to the function values of the previous configuration and adapting the exploration-exploitation trade-off of the acquisition function of the previous configuration. This transfer learning approach was published in conference proceedings [HG21a] and is adapted into an extended version in this chapter.

In Section 6.2, we detail the transfer learning approach to decrease the number of function evaluations needed in subsequent beamline configurations. In Section 6.3, we describe our case study, which is based on the IBA Proteus ONE beamline [App]. The problem is then formally defined in Section 6.4 where we determine the constraints on the treatment beam characteristics and the design of the objective function. In Section 6.5, we introduce *Manzoni*, a digital twin developed by Tess et al. [RT18] that simulates beam-matter interactions in the beamline. This enables us to test and fine-tune our optimization routines easily. In section 6.6, we present the results obtained with our approach, compare the Nelder-Mead algorithm with the Bayesian Optimization approach and investigate the impact of transfer learning. We finish with a brief conclusion in Section 6.9.

6.2 Transfer learning

Transfer learning is usually described as transferring information gained from one domain to a related domain to improve the learner's capability or because of insufficient data available for the actual learning task in the new domain [WKW16]. However, in this work, we consider that the transfer of information is from the same domain but from a different configuration. Transfer of information in the context of Bayesian optimization has been studied in the literature for tuning the hyperparameters of machine learning algorithms. In [B+13], Bardenet et al. used a Gaussian process (GP) to learn a surrogate-based ranking function to transfer knowledge across tasks. In [YM14], Yogatama and Mann transfer knowledge from past experiments using deviations from the previous dataset mean via a common surrogate function. More recently, in [J+19], Joy et al. introduced a noise variance to model the relatedness between datasets and estimate it via an inverse gamma distribution. In this work, we propose an idea similar to [J+19], but we estimate the noise by maximizing the marginal log-likelihood of the GP model. Moreover, we use mutual information

[CPV14] as the acquisition function of the BO and reuse the exploration-exploitation trade-off of the initial task to optimize the second.

In Chapter 5, we have seen that it is possible to infer the noise standard deviation with the marginal log-likelihood (MLL). What if we have a noise-free (or minimal noise) dataset and a noisy dataset upon which we would like to regress? Or two data sets with possibly different noise variance? We consider that an optimization problem is first solved in a source configuration with dataset D^S of size N_S. Then we wish to solve it again in a target configuration D^T of size N_T applying a transfer learning approach. The approach for transfer learning we propose in this work is two-fold. First, we infer the noise variance of D^S with respect to D^T. We define $\sigma^2_{n_S}$ to be the noise variance of the source configuration with respect to the target configuration and $\sigma^2_{n_T}$ to be the noise on the observations of the target configuration. We can take into account those noise variances directly in the GP by reformulating the covariance matrix $\mathbf{K}$ as

$$\mathbf{K}_n = \mathbf{K} + \sigma^2_{n_S} \left[\begin{array}{c|c} \mathbf{I}_{N_S} & \mathbf{0}_{N_S \times N_T} \\ \hline \mathbf{0}_{N_T \times N_S} & \mathbf{0}_{N_S \times N_T} \end{array}\right] + \sigma^2_{n_T} \left[\begin{array}{c|c} \mathbf{0}_{N_S \times N_S} & \mathbf{0}_{N_S \times N_T} \\ \hline \mathbf{0}_{N_T \times N_S} & \mathbf{I}_{N_T} \end{array}\right] \tag{6.1}$$

The optimal noise variances can be estimated in the same way as the kernel hyperparameters, i.e. by maximizing the MLL with respect to $\sigma^2_{n_S}$ and $\sigma^2_{n_T}$, i.e. by computing $\frac{\partial}{\partial \sigma_{n_1}} \log p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})$ with

$$\frac{\partial \mathbf{K}_n}{\partial \sigma_{n_S}} = 2\sigma_{n_S} \left[\begin{array}{c|c} \mathbf{I}_{N_S} & \mathbf{0}_{N_S \times N_T} \\ \hline \mathbf{0}_{N_T \times N_S} & \mathbf{0}_{N_S \times N_T} \end{array}\right] \tag{6.2}$$

$$\frac{\partial \mathbf{K}_n}{\partial \sigma_{n_T}} = 2\sigma_{n_T} \left[\begin{array}{c|c} \mathbf{0}_{N_S \times N_S} & \mathbf{0}_{N_S \times N_T} \\ \hline \mathbf{0}_{N_T \times N_S} & \mathbf{I}_{N_T} \end{array}\right] \tag{6.3}$$

Note that the approach can easily be extended to account for more datasets by seeking different noise variances. However, the computation of the optimal parameters can quickly become time-consuming. A similar approach was undertaken in [J+19], in which the noise variance was estimated via an inverse gamma distribution instead of maximizing the MLL. We show in Section 6.6 that this approach performs worse for our specific application.

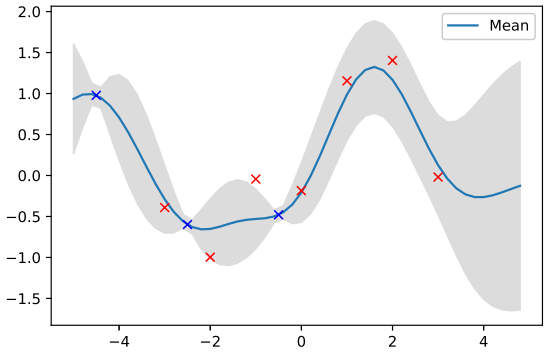


Fig. 6.1 GP posterior after optimization of kernel hyperparameters and noise std. The red cross indicates the noisy data points, and the blue cross indicates the noise-free data points.

In this work, we assume that the noise on the observations is fixed and very low. The purpose is thus to seek the optimal $\sigma_{n_S}^2$ for transfer learning, i.e. $\sigma_{n_S} = \arg\max_{\sigma_{n_S}} \log p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})$ while $\sigma_{n_T}^2$ remains fixed.

Let us go back to the 1D example of Chapter 5 and apply this principle. We generate two sets of points: one with $\sigma_{n_S} = 0.4$ and another with $\sigma_{n_T} = 10^{-5}$. We assume that the data points coming from the target set are noise-free (i.e. $\sigma_{n_T} = 10^{-5}$) and that we know that the data points coming from the source set are noisy (with respect to the target set) but that we do not know the noise standard deviation σ_{n_S}. The data points for the noisy set D_S are the same as from previous examples (see Figure 5.5 and 5.6), while for the second data set, we added 3 points $(-4.5, -2.5, -0.5)$. For this synthetic example, we reuse the same kernel function as in previous examples, i.e. a squared exponential kernel function. We simultaneously optimize the kernel hyperparameters l, σ_f and the noise's standard deviation of the source set σ_{n_S}. The optimal results obtained after maximizing the marginal log-likelihood are $(l, \sigma_f, \sigma_{n_S}) = (0.63, 1.02, 0.36)$. The optimizer was able to get quite a good approximation of the correct noise standard deviation σ_{n_S} used to generate the data points (off by 10%). Results of the GP posterior using those hyperparameters are shown in Fig. 6.1. We observe that the uncertainty around the noise-free data points (in blue) is small compared to the rest of the data points. Note that we could

also have optimized the noise's standard deviation σ_{n_T} with the rest of the hyperparameters; it would also work. However, if we know in advance that those observations should, in principle, be noise-free, there is no need to complexify the search domain for the optimizer.

The second ingredient in our transfer learning approach is based on a modified version of the mutual information acquisition function. As a reminder from Chapter 5, the conventional mutual information acquisition function is defined as

$$a_{\mathrm{MI}}(\mathbf{x}) = \mu(\mathbf{x}) - \sqrt{\alpha}\left(\sqrt{\hat{\gamma}(t)} - \sqrt{\hat{\gamma}(t-1)}\right) \tag{6.4}$$

where $\hat{\gamma}(t) = \sum_{i=1}^{t} \sigma_i^2(\mathbf{x})$ with $\sigma_i^2(\mathbf{x})$ the variance of the i^{th} GP at $\mathbf{x}$. In this work, we make a small change to the quantity $\hat{\gamma}_T$ to empirically control the trade-off between exploration and exploitation of the source configuration. We define

$$\hat{\gamma}_T(t) = \sum_{j=1}^{t} \sigma_{T_j}^2(x_j) + C \sum_{i=1}^{N_S} \sigma_{S_i}^2(x_i) \tag{6.5}$$

where $\sigma_{T_j}^2$ and $\sigma_{S_i}^2$ are the variance of the j^{th} GP of the target and i^{th} GP of the source respectively, and $C \in [0, 1]$ is a weight that we choose in this work to be either $C = 1$ or $C = 0$ (although one could also adjust it with respect to $\sigma_{n_S}^2$).

6.3 The IBA Proteus ONE beamline

A schematic representation of the IBA Proteus ONE beamline is shown in Fig. 6.2. It is composed of a static and a dynamic part. The dynamic part is contained within the gantry and is moving with it. It has a rotating angle of 220°. The static part, called the extraction beamline, is not contained in the moving structure.

The beamline is composed of 4 different types of magnets:

- 9 **quadrupole magnets** [QXC & QXG]: they aim to focus the beam in the vacuum tube.
- 3 **bending magnets** [BXG]: dipole magnets used to bend the trajectory of the protons in one direction.

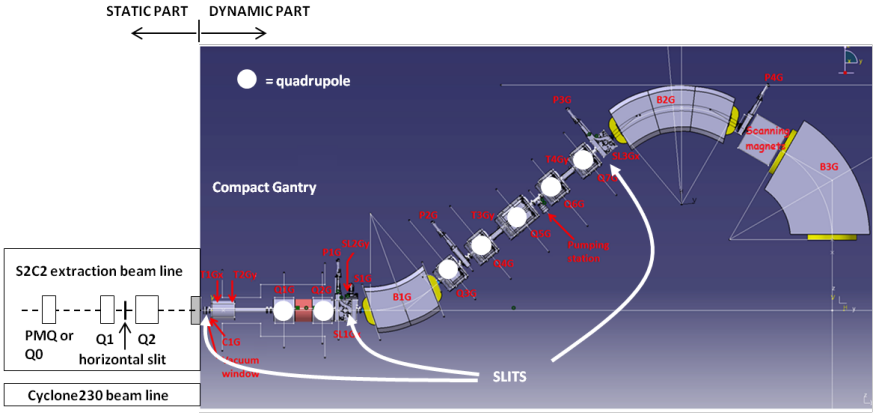


Fig. 6.2 IBA Proteus ONE beamline

- 4 **steering magnets** [TXG]: small dipole magnets used to adjust the beam trajectory in one or two directions.
- 2 **scanning magnets**: dipole magnets, one along the X direction and the other along the Y direction. They deflect the beam to target different spot positions at the isocenter[1].

Others non-magnets components compose the beamline:

- 4 **slits** [SL1E & SLXG]: adjustable apertures used to reshape and constrain the proton distribution.
- 1 **degrader** [D1G]: wheel composed of different material blocks used to decrease proton energy by selecting the right thickness. It is the last element of the static part.
- 4 **beam profile monitors** [PXG]: stripped ionization chambers used to measure the transverse proton distribution. They can be inserted or removed from the beam path.
- 2 fixed **ionization chambers** [IC]: complex ionization chamber with strips and integral planes to measure the number of protons and the transverse proton distribution. They are located after B3G (not visible in Figure 6.2). They are used in real-time to check beam properties at the beamline exit and stay in the beam path during the irradiation.

[1]The isocenter is the reference point in space relative to the treatment machine where the beams intersect when the gantry is rotated.

6.4 Problem description

The calibration goal is to tune the various elements in the beamline to satisfy a set of constraints on the characteristics of the beam. A total of 20 parameters must be adjusted for different energies and gantry angles. The goal of the calibration is to find an application

$$\{\text{Energy; gantry angle}\} \rightarrow \{20 \text{ setpoints}\} \tag{6.6}$$

which fulfills the constraints. This application is called an optical solution.

As the system is not ideal and quite complex, it is impossible to find an analytical expression of this application. It is presented as a large look-up table (LUT). Building the LUT is the purpose of the calibration process. The system is designed to work for a range of proton depth from 4.1cm to 32.5cm, and setpoints are computed for every centimeter.

Part of the beamline elements are independent of the energy and gantry angles (or at least not significantly interfering in the optical solution) and are tuned offline before the calibration. The elements that substantially impact the beam characteristics are the quadrupoles, and we will focus on those for the optimization procedure. A second part of the calibration is related to the two scanning magnets. It is more straightforward and does not require such a large LUT. It is not strictly part of the optics calibration.

The calibration procedure is divided into multiple phases. In the first phase, the slit apertures are set up manually. The bending magnets, steering magnets, and the two quadrupoles of the static part of the beamline are also tuned manually. Then, the optimization process can seek an optical solution for the seven remaining quadrupoles (Q1G, Q2G, Q3G, Q4G, Q5G, Q6G, Q7G, see Figure 6.2). This operation is repeated for multiple energies (or proton ranges) at a fixed gantry angle. Then, the gantry angle is also varied. There is a total of 30 ranges and 8 gantry positions to optimize. Between those, an interpolation between the solutions is used. The calibration procedure usually takes several weeks to complete manually.

6.4.1 Constraints on the beam characteristics

The following constraints need to be satisfied in order to find an optical solution:

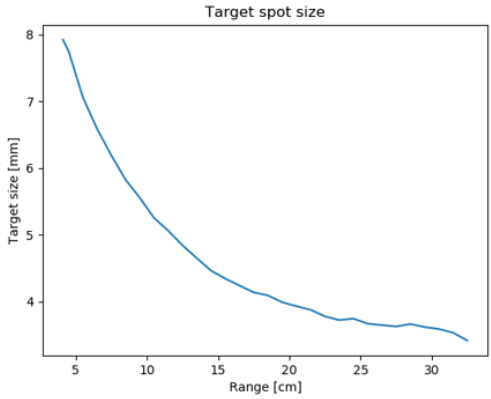


Fig. 6.3 Target spot size as a function of the proton range.

- Spot size difference between measurement and target < 0.1 mm for both x and y. The spot target size for the different levels of energy (ranges) is shown in Fig. 6.3.
- Central spot asymmetry along major/minor axes of the 2D Gaussian < 10% (with the aim to be below 5%)
- Spot size difference while scanning[2] < 10% (with the aim to go below 3.5%) for both x and y
- Beam size on Ionization chambers (IC1, IC2) > 2.4 mm for x and y
- Smoothness of spot size vs. range

6.4.2 Design of the objective function

There are several objectives to meet in order to obtain an acceptable beam shape at the isocenter. The objective function is defined as a weighted sum of the different penalties (representing the beam constraints). The objectives are weighted by their tolerances so that each objective is weighted equally independently of the objective magnitude. Once the constraints are satisfied, the optimization stops. Therefore, this is more a feasibility

[2]The scanning magnets can steer the beam in any direction. At each iteration of the calibration, five beams are shot. The difference in spot size between the beam at the isocenter and the four corners should be below 10% at most (3.5% is better)

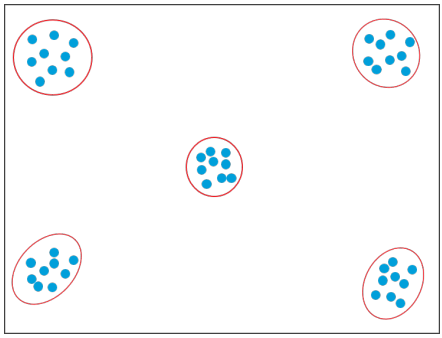


Fig. 6.4 Illustration of the beam calibration. At each iteration, five successive beams hit the receiver: one at the isocenter and four on the corner of the receiver. An ellipse is then fitted on those data.

problem than a conventional optimization problem.

When the beam of protons hits the receiver (i.e. an ionization chamber at the output of the beamline), a 2-dimensional ellipse is fitted around it where the lengths of its axes are defined by twice the standard deviations of the data ($2\sigma_x, 2\sigma_y$), representing a 95% confidence interval across the data. We can then derive the beam characteristic sizes and shapes based on that ellipse.

During the calibration phase, whenever we test specific set points, the system shoot five beams: one at the isocenter, and four on the corner of the receiver via the scanning magnets, as illustrated in Figure 6.4. The beam characteristics are based on the ellipse fitting around those five spots. We define the following beam characteristics:

- size_x (respectively size_y): size of the beam along the x-axis (resp. y-axis) in mm, defined as $\frac{\max(\text{sizes\_x})+\min(\text{sizes\_x})}{2}$ where sizes_x is an array containing the sizes of the beam along the x-axis (resp. y-axis) of the 5 spots.
- sizes_L (resp. sizes_l): array of the sizes of the beam along the major (resp. minor) axis of the 2D ellipse in mm of the 5 spots.
- assym_LS: assymetry of the beam defined along the major and minor axis of the 2D ellipse, defined as $\max_i \left(\left| \frac{\text{sizes\_L}_i - \text{sizes\_l}_i}{\text{sizes\_L}_i + \text{sizes\_l}_i} \right| \right)$ where i designates a specific spot.

- size_IC1_x (resp. size_IC1_y): size of the beam on the x-axis (resp. y-axis) at ionization chamber 1 in mm
- size_IC2_x (resp. size_IC2_y): size of the beam on the x-axis (resp. y-axis) at ionization chamber 2 in mm
- var_size_x (resp. var_size_y): difference in size across the 5 spots, defined as $\frac{\max(\text{sizes\_x})-\min(\text{sizes\_x})}{\max(\text{sizes\_x})+\min(\text{sizes\_x})}$ where sizes_x (resp. sizes_y) represent the sizes along the x-axis (resp. y-axis) obtained at iso-center and the four corners of the receptor via the scanning magnets.

From those beam characteristics, we define the following penalty terms that are used in the objective function:

$$s_x = \begin{cases} 0 & \text{if size\_x} < \frac{\text{tol\_size}}{2} \\ |\text{size\_x} - \text{tol\_size}| & \text{otherwise} \end{cases}$$

$$s_y = \begin{cases} 0 & \text{if size\_y} < \frac{\text{tol\_size}}{2} \\ |\text{size\_y} - \text{tol\_size}| & \text{otherwise} \end{cases}$$

$$a = \begin{cases} 0 & \text{if assym\_LS} < \frac{\text{tol\_assym}}{2} \\ \text{assym\_LS} & \text{otherwise} \end{cases}$$

$$s_{IC1x} = \begin{cases} 0 & \text{if size\_IC1\_x} > \text{target\_IC} + \text{tol\_IC} \\ \text{target\_IC} + \text{tol\_IC} - \text{size\_IC1\_x} & \text{otherwise} \end{cases}$$

$$s_{IC1y} = \begin{cases} 0 & \text{if size\_IC1\_y} > \text{target\_IC} + \text{tol\_IC} \\ \text{target\_IC} + \text{tol\_IC} - \text{size\_IC1\_y} & \text{otherwise} \end{cases}$$

$$s_{IC2x} = \begin{cases} 0 & \text{if size\_IC2\_x} > \text{target\_IC} + \text{tol\_IC} \\ \text{target\_IC} + \text{tol\_IC} - \text{size\_IC2\_x} & \text{otherwise} \end{cases}$$

$$s_{IC2y} = \begin{cases} 0 & \text{if size\_IC2\_y} > \text{target\_IC} + \text{tol\_IC} \\ \text{target\_IC} + \text{tol\_IC} - \text{size\_IC2\_y} & \text{otherwise} \end{cases}$$

$$v_x = \begin{cases} 0 & \text{if var\_size\_x} < \frac{\text{tol\_var\_size}}{2} \\ \text{var\_size\_x} & \text{otherwise} \end{cases}$$

$$v_y = \begin{cases} 0 & \text{if var\_size\_y} < \frac{\text{tol\_var\_size}}{2} \\ \text{var\_size\_y} & \text{otherwise} \end{cases}$$

The tolerance parameters (tol_size, tol_assym,...) reflect the beam's constraints expressed in section 6.4.1. The different tolerances are

- tol_size = 0.1 mm
- tol_assym = 10%
- tol_IC = 0.1mm
- tol_var_size = 10%

Those tolerances correspond to the maximum allowed violation. Tighter constraints for asymmetry and var_size can be used with values of 5% and 3.5%, respectively, to obtain a better solution.

Notice that if a penalty term is lower than half of the tolerance, we consider it to be zero, as further changes would not make the solution significantly better. For the size of the beam on the ionization chambers, the penalty is considered zero if the size is greater than the target (2.4mm) plus the tolerance (0.1mm).

The objective function that we want to minimize is finally defined as

$$f = \frac{s_x^2 + s_y^2}{\text{tol\_size}^2} + \frac{|s_x - s_y|^2}{\text{tol\_size}^2} + \frac{a}{\text{tol\_assym}} + \frac{s_{IC1x} + s_{IC1y} + s_{IC2x} + s_{IC2y}}{\text{tol\_IC}} + \frac{v_x + v_y}{\text{tol\_var\_size}} \quad (6.7)$$

Every penalty term is divided by its tolerance leading to an equal contribution for each one. Notice the design choice to square the beam size parameters and hence put more emphasis on reducing large variations with the target beam size for those penalties. The reason behind this is to obtain a beam size close to the actual target size before focusing on the other penalty terms. Finally, we added the absolute difference between the sizes ($|s_x - s_y|$) in the objective function. This could be disputed since we already have an asymmetry term for this purpose with the variable a. However, this modification was found to work better in practice.

6.5 Digital twin

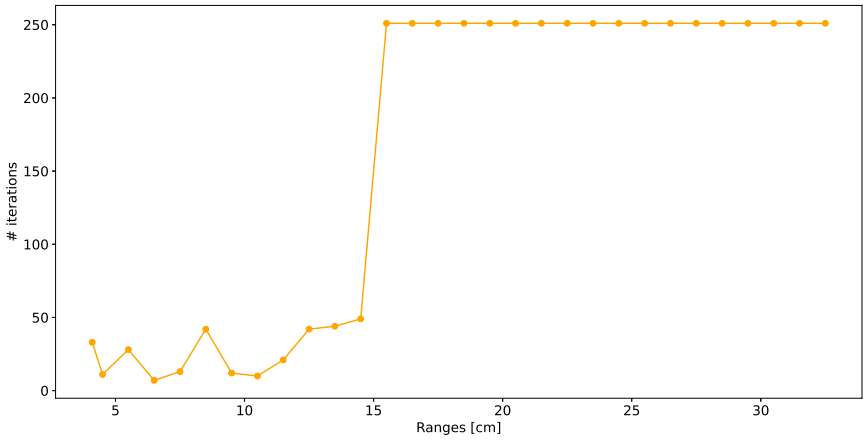
A simulation software called *Manzoni* was designed by Tesse et al. in collaboration with IBA. It is a fast-tracking code for beam transport and sim-

ulation of beam-matter interactions in hadron therapy beamlines [RT18]. It is used in this thesis as a digital twin of the IBA Proteus ONE beamline. The simulation system models the behavior of a bundle of particles, the beam, represented in phase space by a six-dimensional ellipse describing the beam size, divergence, energy, and energy dispersion. The effects of the different elements of the beamline are assumed to be linear, represented by matrix multiplications of this six-dimensional ellipse. On top of those interactions, multiple Coulomb scatterings are also simulated. Although the system is considered linear, the relationship between the magnet fields and currents is not linear. This, on top of the multiple Coulomb scattering between particles, makes this problem impossible to formulate via an analytical function, requiring black-box optimization techniques. This software is used to test and fine-tune our optimization procedures.

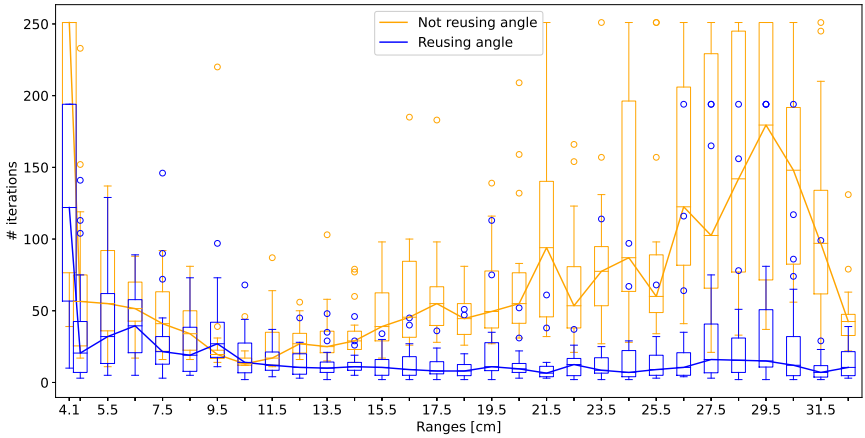
Although the simulator has been validated with experimental data, there are still differences between the simulator and real site measurements. This can potentially be due to a cyclotron output distribution different than what is used in the simulator, a different current-to-field relationship in the magnets, non-Gaussian effects not taken into account in the simulator, or many other approximations used by the model. Nevertheless, the simulator remains an essential tool for testing our optimization procedures.

6.6 Results

In this section, we apply the Nelder-Mead algorithm and Bayesian optimization for optimizing the magnet currents in 7 quadrupoles of the IBA Proteus ONE beamline. The problem must be solved for several configurations, i.e. several beam ranges and gantry angles. We suppose the beam ranges to be independent problems, while the gantry angles can use a transfer learning approach to reuse the information obtained from a previously calibrated gantry angle. In a first analysis, we compare the optimization outcome of Nelder-Mead and BO with the Manzoni digital twin. For the BO case, we also show the outcome of reusing the information from another gantry angle on the number of function evaluations needed to satisfy the constraints. As input, we provide a list of magnet currents and receive a 2D dose distribution as output. Objective function (6.7) translates this dose distribution into a scalar metric quantifying its correctness. The optimization stops when all constraints on the beam are met, regardless of the objective function value (which can be nonzero), or when a maximum



(a) Nelder-Mead algorithm



(b) Bayesian optimization with(out) transfer learning. Experiments are restarted 20 times to account for uncertainties. Solid lines represent the median number of function evaluations.

Fig. 6.5 Calibration results on *Manzoni* digital twin for various proton ranges. Reaching 250 iterations means that no solution was found.

of 250 iterations has been reached. The number of function evaluations needed to satisfy the constraints is given in Figure 6.5a for the Nelder-Mead algorithm and Figure 6.5b for the Bayesian optimization with the mutual information (MI) acquisition function. Each method starts at the same initial solution, which is the average of the current setpoints of previously calibrated sites for that (energy, gantry angle) pair.

We observe that qualitatively Bayesian optimization outperforms the Nelder-Mead (NM) algorithm. NM was indeed unable to find a solution that satisfies the beam constraints for proton ranges above 15 cm. BO, on the other hand, was able to find a solution in all cases. Notice that since our implementation of BO has a random component (e.g. drawing initial solutions for the hyperparameters in Algorithm 3 or drawing initial solutions for the acquisition function in Algorithm 4), experiments have to be restarted 20 times to account for this randomness. We can see that the number of function evaluations can vary quite a lot for certain ranges. This can be explained because sampling elsewhere can lead to a completely different surrogate and another exploration scenario. Another observation is that using a previously calibrated gantry angle via transfer learning drastically decreases the number of function evaluations needed and its uncertainty. A second, more quantitative analysis is depicted in Table 6.1 and the results are explained hereunder.

One gantry angle, no transfer learning First, we solve the optimization problem for the initial configuration, in our case, a specific angle ($\theta_S = 0°$), for ten beam ranges without transfer learning. We compare BO with the mutual information (BO-MI) acquisition function against the conventional lower confidence bound (BO-LCB) acquisition function and the Nelder-Mead algorithm in the first part of Table 6.1. Experiments are restarted 20 times to account for the random nature of the BO. We observe that optimization of higher ranges is more complex, and the Nelder-Mead algorithm is not capable of finding a solution satisfying the constraints in the maximal 250 iterations allowed. Between the two acquisition functions for the Bayesian optimization, MI performs better in almost all cases.

Two gantry angles with transfer learning Next, we solve the calibration problem for another configuration ($\theta_T = 90°$) by reusing the currents-objective values pairs of the source configuration for each beam range. We also report a baseline without transfer learning for this new angle (second part of Table 6.1). We notice that the problem with this second angle seems easier for the Nelder-Mead algorithm but is still failing for the first range (4.1), while BO-MI performs equally well on average and better than LCB. In the third part of the table, we compare the different approaches for transfer learning. We compare our proposed approach (σ_{MLE}, $C = 1$ in Table 6.1) with the approach from [J+19], i.e. by estimating the noise variance with an inverse gamma distribution, and with the Nelder-Mead algorithm where

Method \ range	4.1	6	9	12	15	18	21	24	27	30	Mean
Source angle $\theta_S = 0°$											
BO-MI	85	47	32	18	32	47	58	69	114	108	61
BO-LCB	108	97	76	59	52	62	53	64	251	163	98.65
Nelder-Mead	33	7	12	42	251	251	251	251	251	251	160
Target angle $\theta_T = 90°$ without reusing information from θ_S											
BO-MI	102	64	22	16	28	60	40	65	97	96	59
BO-LCB	122	103	78	38	54	52	74	73	146	171	91.35
Nelder-Mead	251	45	16	11	46	57	52	59	56	56	64.9
Target angle $\theta_T = 90°$ reusing information from θ_S with BO-MI											
$\sigma_{MLE}, C = 1$	**21**	**12**	16	**5**	**6**	5	**4**	**3**	**5**	16	**9.3**
Nelder-Mead	24	28	**11**	20	9	10	10	8	6	**9**	13.5
Inv. gamma	194	122	74	63	77	97	68	143	194	121	115.3
Inv. gamma, MI	111	38	43	25	33	48	57	73	76	103	60.7
$\sigma_{MLE}, C = 0$	27	20	31	8	13	6	8	6	**5**	17	14.1
$\sigma = 10^{-5}, C = 1$	66	24	18	**5**	8	**4**	**4**	5	**5**	11	15
$\sigma = 0.1, C = 1$	79	53	32	27	39	30	52	64	52	50	47.8

Table 6.1 Median number of iterations to reach an acceptable solution for different transfer learning scenarios. Columns are different beam ranges (independent problems). The first part of the table refers to the source configuration ($\theta_S = 0°$), the second part refers to the target configuration ($\theta_T = 90°$) without transfer learning, and the third part refers to the target configuration ($\theta_T = 90°$) and reuses the input-output-pairs computed by the best algorithm (BO-MI) of the first configuration ($\theta_S = 0°$).

the initial simplex is built with the last values of the source configuration (the last $N + 1$ input-output pairs of the BO-MI for $\theta_S = 0$). From those results, we first observe that our approach of estimating $\sigma^2_{n_S} = \sigma^2_{MLE}$ by maximizing the MLL of the GP and reusing the $\hat{\gamma}$ parameter of the MI acquisition function ($C = 1$) helps to reduce the number of iterations needed for the target configuration by more than 80% (ratio of the means). Second, it performs better than Nelder-Mead and the approach from [J+19]. Even if we modify the approach from [J+19] to select the same acquisition function as our approach (MI) and reuse the gamma parameter of the source configuration, we get worse results than σ_{MLE} (although better than the original method). This may be due to a poor prior candidate for the inverse gamma distribution. We also tested a few variations of our approach (σ_{MLE}). When we choose not to reuse the $\hat{\gamma}$ parameter of the MI acquisition function (i.e. $C = 0$), the number of iterations increases for all ranges. Moreover, if we arbitrarily fix the noise variance to $\sigma = 0.1$ or $\sigma = 10^{-5}$ instead of estimating it, the number of iterations also increases.

6.7 Onsite results

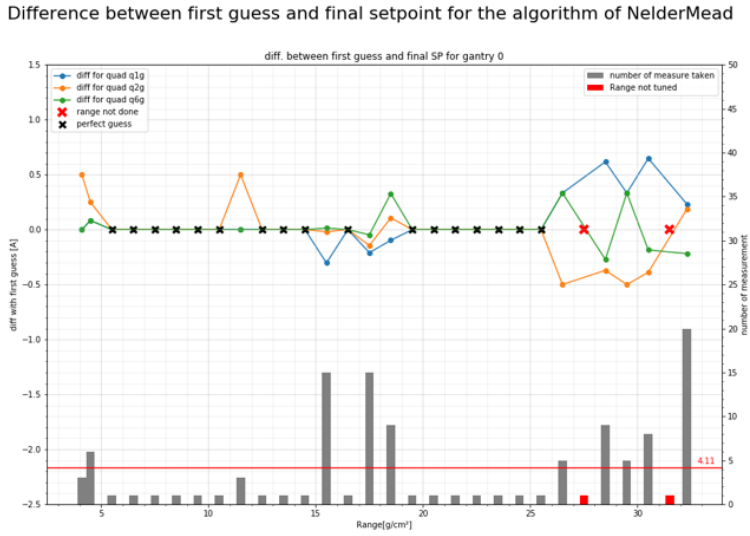
A mission to the Leuven Proton Therapy site took place in January 2020 to evaluate in the real world the penalty-based objective function used to model the problem and to validate the algorithms developed for the calibration. Two approaches were tested, referred to as the pre-optic calibration phase and the optic calibration phase. In the pre-optic phase, we start from a generic solution and need to calibrate seven quadrupole magnets for seven proton ranges. In the optic phase, we start from a pre-tuned solution, which is an interpolation between previously optimized proton ranges done in the pre-optic phase, and calibrate three quadrupole magnets (the other four remaining fixed) for the remaining ranges. Only one gantry angle was tested, meaning no transfer learning was performed. The results for the Nelder Mead algorithm and the Bayesian optimization are discussed below. The maximum number of function evaluations was fixed at 100 for the pre-optic phase; for the optic phase, it was set at 20. A function evaluation takes on average a few seconds to perform.

6.7.1 Nelder-Mead

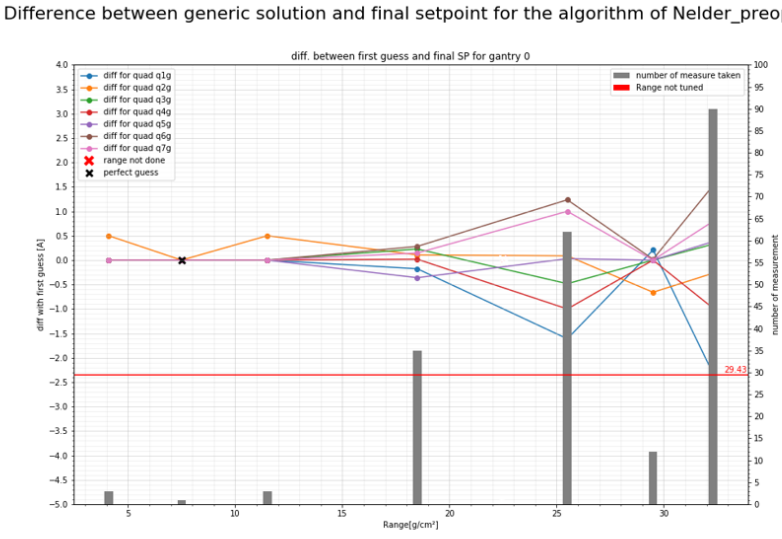
The results of the calibration with the Nelder-Mead algorithm are shown in Figure 6.6. We can observe that the setpoints of many ranges were found even without optimization, simply via interpolation between their closest ranges (black crosses on the figure). For the other ranges, the optimization is carried in about 10-20 iterations for the optic phase and 10-90 iterations for the pre-optic phase. Note that higher ranges tend to be more difficult to optimize. We also note that the algorithm failed to optimize two ranges. The reason might be that the simplex became too small or the maximum number of function evaluations was set too low.

6.7.2 Bayesian Optimization

The calibration results with the Bayesian Optimization algorithm are shown in Figure 6.6. The optimization was carried out with a previous version of our BO algorithm using the LCB acquisition function, which we know from the results in Table 6.1 is not as good as the MI acquisition function. Nevertheless, we can observe that the setpoints for many ranges were found without optimization, as previously observed for the Nelder-Mead algorithm. The optimization is carried out for the other ranges in about 20-30 iterations for the optic phase and 20-70 iterations for the pre-optic phase.

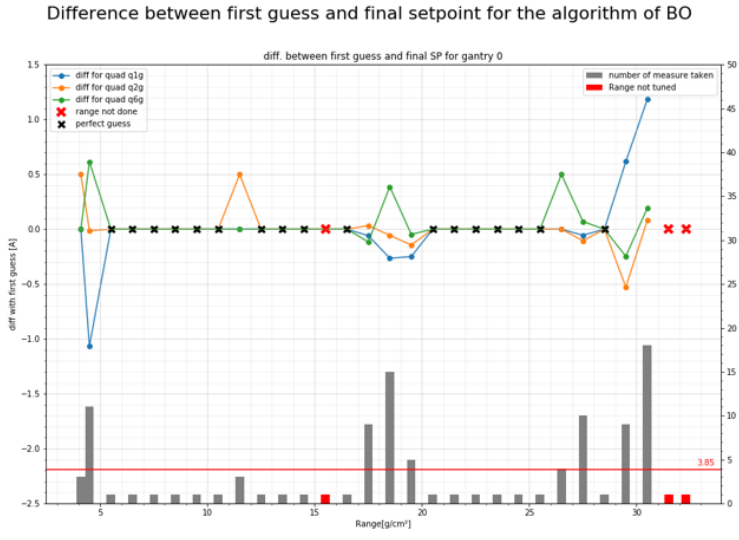


(a) Optic phase: 3 quadrupoles with a pre-optimized solution

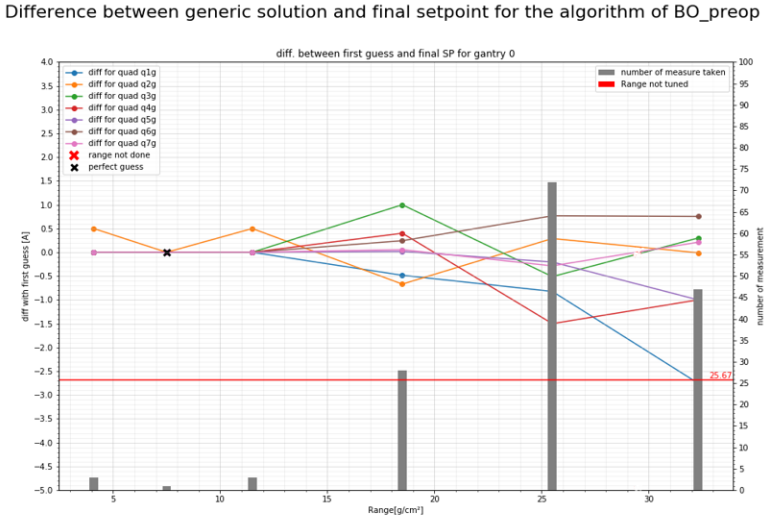


(b) Pre-optic phase: 7 quadrupoles with a generic solution

Fig. 6.6 Calibration results on Leuven PT site with Nelder-Mead algorithm. The bins represent the number of measurements taken, the red horizontal lines represents the mean number of measurements, and the solid lines represent the differences in current amplitudes from the initial solution.



(a) Optic phase:3 quadrupoles with a pre-optimized solution



(b) Pre-optic phase:7 quadrupoles with a generic solution

Fig. 6.7 Calibration results on Leuven PT site with Bayesian Optimization.

Note that higher ranges tend to be more difficult to optimize. We also note that the algorithm failed to optimize the two highest ranges in the optic phase. The reason might be that number of maximum iterations was set too low.

The optimization algorithm was further used in the installation of new proton therapy systems. On average, the time required to calibrate the entire system (30 energy ranges and 8 gantry angles) was reduced by 40% compared to the manual operation. However, this gain is likely underestimated for several reasons. First, the script used on site was a previous version of the BO algorithm using the LCB acquisition function and did not use our transfer learning approach, resulting in more time spent to optimize. Second, the handling of the calibration script by the engineers on site takes some time as the approach is not fully automated yet. A better performance measure would be the ratio between the number of function evaluations required by the algorithm and the manual procedure. Unfortunately, this information was not available to us.

6.8 Discussion

Recommendation Our recommended algorithm for calibrating a new proton therapy beamline is the Bayesian optimization algorithm with the mutual information acquisition function, using the transfer learning approach we developed to speed up the calibration once a gantry angle has been fully calibrated. Indeed, this is the approach that led to the fewest number of function evaluations. The Nelder-Mead algorithm could perhaps play a role when we know that the current setpoints should be close to the initial solution provided to the algorithm; for instance, if a PT site would need recalibration. As it requires very few computational resources, it would make sense to try Nelder-Mead first before a computationally intensive bayesian optimization.

Derivative-based approach Estimating the gradient of the objective function with finite differences is probably not a good idea. However, we could wonder if it is possible to differentiate through the beamline simulator with an automatic differentiation tool. As such, the digital twin contains non-differentiable elements such as the degrader and the slits. However, most of the elements are differentiable since they can be represented via matrix

multiplications of a six-dimensional vector representing the beam characteristics. We can therefore wonder if neglecting or approximating non-differentiable elements and back-propagating the gradient only through those differentiable elements could provide a decent estimate of the gradient and be used in a gradient-based algorithm. This could be investigated in a future work.

6.9 Conclusion

In this chapter, we developed methods for automatically calibrating the magnet currents in a proton therapy beamline in order to produce a treatment beam respecting tight constraints. We showed that this problem was solvable with a moderate number of function evaluations, allowing to considerably decrease the physicists' work time for manually adjusting those currents. The digital twin allowed us to test and fine-tune our methods and develop a transfer learning framework for Bayesian optimization. This framework is based on the mutual information acquisition function that estimates a source noise variance by maximizing the Gaussian process marginal likelihood and by reusing the exploration-exploitation trade-off of the source configuration. We showed that it drastically reduces the number of iterations needed to calibrate a proton therapy beamline and outperforms existing transfer learning approaches. Our methods were also validated at an actual proton therapy site. However, the transfer learning approach was not tested on site due to time constraints.

PART III
Treatment of mobile tumors in proton therapy with a library of treatment plans

7

Background in radiation therapy

This chapter is inspired by the course of John Lee, Guillaume Janssens, and Edmond Sterpin on proton therapy given at UCLouvain [LSJ21] and the book of Harald Paganetti on proton therapy physics [Pag18].

7.1 Clinical workflow

The clinical workflow of a radiotherapy treatment is outlined in Figure 7.1. It consists of five steps that are detailed in the following sections.

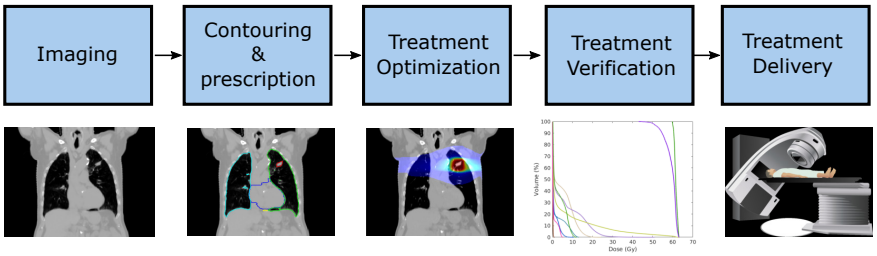


Fig. 7.1 Clinical workflow of a radiotherapy treatment

7.1.1 Imaging

Imaging is the first step of the treatment workflow. It usually consists of taking a computed tomography (CT) scan of the patient. It works by shooting multiple fan beams of X-rays with the machine rotating around the patient. The signals generated are then picked up by detectors on the opposite side of the X-ray source, which are then processed by a computer to generate a cross-sectional 2-dimensional image referred to as a *slice*. By taking cross-sectional images at successive close positions (usually in the order of 1-10 millimeters) and stacking them together, we can obtain a 3-dimensional computed tomography (3DCT) scan that gives detailed information on the patient's anatomy, unlike a conventional X-ray radiograph that only uses a single beam angle.

On top of the anatomy qualitative information, the CT scan provides quantitative information on the tissue density and stopping power ratios (although the conversion to stopping power is not straight)), needed to compute the absorbed dose in the body and optimize a treatment plan. The stopping powers represent the average energy loss per unit of distance along the track of the proton or photon beam. The CT scan is represented as a 3-dimensional regular grid where each voxel[1] is expressed in a dimension called the Hounsfield unit (HU). It is a linear transformation of the original photon attenuation coefficient where 0 is defined as the radiodensity[2] of water and -1000 as the radiodensity of air in standard temperature and pressure. It is given by

$$HU = 1000 \cdot \frac{\mu - \mu_{water}}{\mu_{water} - \mu_{air}}$$

where μ is a measured attenuation coefficient. CT scanners can then be calibrated to find the relationship between HU and stopping powers. For more information on this process, the reader can refer to [ZP16].

When motion of the anatomy needs to be taken into account for a radiotherapy treatment such as breathing-induced motion in lung or liver cancers, a 4-dimensional (3D + time) computed tomography (4DCT) can be acquired. The 4DCT generally consists of eight to ten successive 3DCTs,

[1] a voxel represents a value in a 3-dimensional regular grid. It is the counterpart of a pixel in 3 dimensions.

[2] ability of radiowave and X-rays to pass through a particular material.

representing variations of the anatomy during a breathing period. The 3DCTs each represent a particular breathing phase of an average breathing period. They are obtained by sorting hundreds of radiographic images gathered during several minutes into their respective phase using an external monitoring device recording the breathing amplitude.

7.1.2 Contouring and prescription

After acquiring a CT scan of the patient, radiation oncologists delineate the tumor and organs at risk (OARs) on the CT scan. Then, different target volumes encompassing the tumor are defined via successive expansion of the volume to account for different types of uncertainties, as described in Figure 7.2.

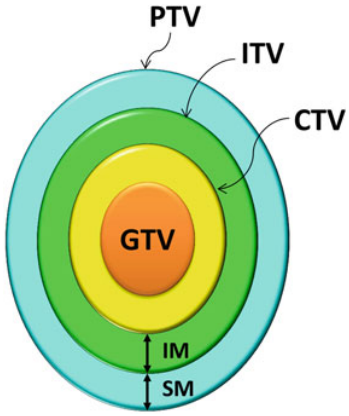


Fig. 7.2 Target volumes. Image taken from [ASH+17] with permission.

GTV: gross tumor volume , defined as the visible tumor volume in the CT scan.

CTV: clinical target volume , defined as the GTV + subclinical/invisible invasion

ITV: internal target volume , defined as the CTV + IM (internal margins to account for organ motion)

PTV: planning target volume , defined as the ITV + SM (setup margin for setup errors)

The ITV is usually obtained by taking the union of the CTVs in each phase of a 4DCT. The PTV extends the ITV (in case of a moving tumor) or the CTV (for a static tumor) to account for setup errors due to imperfect patient positioning. For photon therapy, the PTV is the target volume used for optimizing a treatment plan. In proton therapy, the concept of security margins in ITV and PTV is usually insufficient, as already mentioned in Section 1.3 of the introduction. A more sophisticated technique is robust optimization, which takes into account a set of uncertainty scenarios and uses the CTV as the target volume of reference for the optimization. This approach is detailed in Section 7.1.3.

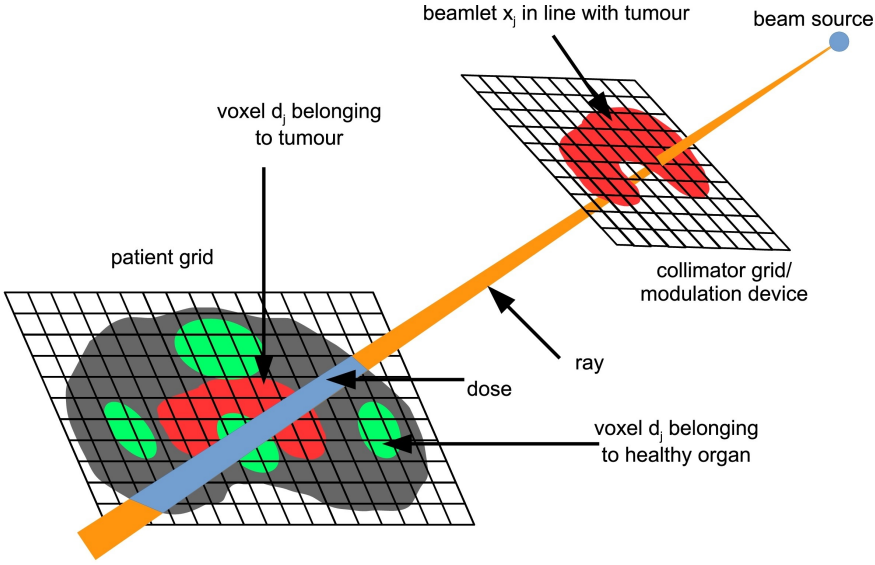


Fig. 7.3 Radiotherapy problem decomposition. Top part: Ionizing radiations are emitted from a beam source, going through a modulation device (in pencil beam scanning, no modulation device is required). The beam is discretized in *beamlets*. Bottom part: the patient is discretized into voxels. Delivering multiple shapes or different beamlet time exposure allows intensity modulation. Image adapted from [BCVHH19] with permission.

Once the target volume and organs at risks are defined, the oncologist decides the dose to be prescribed in the target volume and the maximum admissible dose in the surrounding OARs. Usually, they are described as inequality constraints based on the mean, maximum, or minimum dose in the volumes of interest. The number of fractions in which the treatment will be administered is also decided. Indeed, RT treatments are usually not administered in a single shot but rather fractionated in the course of multiple days or weeks where a fraction of the prescription is delivered. The reason for this fractionated scheme is to allow time for healthy tissues to repair themselves between treatment fractions, therefore reducing side effects [HMZ14].

7.1.3 Treatment optimization

The decomposition of a conventional radiotherapy problem is described in Figure 7.3. In the figure, the beam is discretized into *beamlets*, the fun-

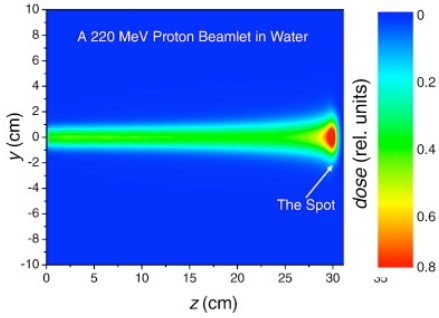


Fig. 7.4 Example of a 220 MeV proton beamlet in water [LSJ21].

damental decision variables of the optimization problem, after passing through the modulation device. In photon therapy, the modulation device is a multi-leaf collimator, while in proton therapy using passive scattering, the modulation devices are the range modulator, collimator, and range compensator (see Figure 1.4a from the introduction). In proton therapy using pencil beam scanning, no modulation device is required; the beam source emitted can directly be considered as a *beamlet* (also called *pencil beam* in this case). In the rest of this section, we will only focus on treatment optimization for proton therapy using pencil beam scanning. For treatment optimization regarding photon therapy, the reader can refer to [BCVHH19] while for proton therapy using passive scattering, the reader can refer to [BA03] and [Pag18, Chapter 15].

The beamlets can be viewed as a basis of elementary functions that can be weighted and summed up to obtain the total dose. The size of the pencil beams are adjustable between $\sigma = 2.5$ mm and $\sigma = 10$ mm [MPB+00]. An example of a 220 MeV proton beamlet in water is shown in Figure 7.4. These weighted Bragg peak "spots" are placed throughout the target volume to obtain a 3D conformal dose. Beamlets can be pre-computed in advance by a dose engine. Usually, a matrix of beamlets is computed for every spot in the target volume plus a small margin. A lateral and depth spacing (i.e. energy spacing) between the spots is chosen by the user, which influences the size of the beamlet matrix and the optimization problem. Formally, the total dose can be expressed as

$$\mathbf{d} = \mathbf{B}\mathbf{x} \tag{7.1}$$

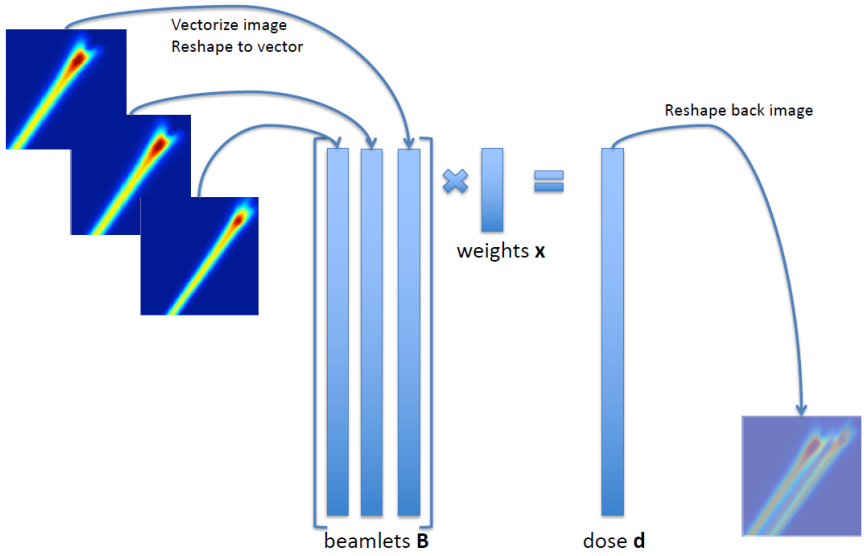


Fig. 7.5 Representation of the total dose computation. The beamlet images are reshaped in vector form in the beamlet matrix before being multiplied by the weight vector to obtain the total dose and then be converted back to an image. Image taken from [LSJ21].

where $\mathbf{d} \in \mathbb{R}^V$ is the total dose, with V the number of voxels in the image, $\mathbf{B}$ is a $V \times N$ beamlet matrix with N the number of beamlets, and $\mathbf{x} \in \mathbb{R}^N$ are the weights to optimize. A representation of equation (7.1) is depicted in Figure 7.5.

Finding the optimal weight vector $\mathbf{x}$ is the purpose of the treatment optimization. A typical formulation of the objective function of the optimization problem is via a weighted sum of asymmetric quadratic penalties. In Section 7.1.2, we mentioned that prescriptions are usually expressed via inequality constraints on target volume and organs at risks. However, it is rarely possible to achieve all desired constraints simultaneously. Instead, they are defined in the objective functions as penalties. The usual constraints used are described in Table 7.1. Usually, a minimum and maximum constraint are applied to the target volume, while maximum and mean constraints are applied to the OARs. It is common to use a minimum and maximum constraint with the same prescription p, which can then be written as a simple penalty function $\frac{1}{|V|}\sum_{i\in V}(d_i - p)^2$. Note that

Type	Constraint	Penalties
Maximum	$\max_{i \in V} d_i < p$	$\frac{1}{\lvert V \rvert} \sum_{i \in V} \max(0, d_i - p)^2$
Minimum	$\min_{i \in V} d_i > p$	$\frac{1}{\lvert V \rvert} \sum_{i \in V} \min(0, d_i - p)^2$
Mean	$\text{mean}_{i \in V}(d_i) < p \Leftrightarrow \frac{\sum_{i \in V} d_i}{\lvert V \rvert} < p$	$\max(0, \text{mean}_{i \in V}(d_i) - p)^2 \Leftrightarrow \max\left(0, \frac{\sum_{i \in V} d_i}{\lvert V \rvert} - p\right)^2$

Table 7.1 Constraints and objective function penalties relationship in treatment optimization. V is the volume on which the constraint is defined, $d_i = \sum_j B_{ij} x_j$ is the dose in voxel i, and p is the prescription.

the constraint $\text{mean}_{i \in V}(d_i) > p$ is omitted in Table 7.1 because it is never used in practice. The min and max penalties are divided by the number of voxels in the volume of interest to reflect an equal contribution of each constraint. In the end, the objective function can be written as a sum of weighted penalties. An example of an objective function with one target volume and two OARs with respectively a max and a mean constraint is given below:

$$f(\mathbf{x}) = \frac{w_1}{|TV|} \sum_{i \in V} \left(\sum_j B_{ij} x_j - p_{TV} \right)^2 + \frac{w_2}{|OAR_1|} \sum_{i \in V} \max\left(0, \sum_j B_{ij} x_j - p_{OAR_1}\right)^2 + w_3 \max\left(0, \frac{\sum_{i \in OAR_2} \sum_j B_{ij} x_j}{|OAR_2|} - p_{OAR_2}\right)^2 \quad (7.2)$$

where TV is the target volume, OAR_1 and OAR_2 are the volumes formed by two organs at risks, and w_1, w_2, w_3 weights to be defined by the user describing the relative importance of each objective. Finally, the optimization problem to solve is

$$\min_{\mathbf{x}} f(\mathbf{x}) \quad (7.3)$$

$$\text{s.t. } x_j \geq l \text{ OR } x_j = 0 \text{ for all } j \in \mathcal{J} \quad (7.4)$$

$$x_j \leq u \text{ for all } j \in \mathcal{J} \quad (7.5)$$

where l and u are respectively a lower and upper bound on the value of each x_j. These bound constraints represent the minimum and maximum intensity the machine can deliver. Note that the lower bound constraint l is only active if beamlet j is selected; otherwise $x_j = 0$. This disjunctive constraint can be handled in two steps[3]. First, the optimization problem is solved for $\mathbf{x} \geq 0$, then only the spots satisfying $\mathbf{x} \geq l$ are kept, and the optimization is refined by applying $\mathbf{x} \geq l$ as a hard constraint to the spots kept from the first step. The good news is that the optimization problem is *convex*, meaning that efficient algorithms can be used to solve the problem and that every local minimum is a global minimum. However, the size of the optimization problem can be quite big. The number of beamlets and decision variables is usually in the order of 10,000, while the number of voxels is in the order of 10 million, although the number of voxels actually used for the optimization is in the order of 100,000.

Single field vs. multi-field optimization

Proton therapy treatment plans are usually delivered with multiple fields from different directions. Indeed, as in photon therapy, having multiple fields improves the overall dose conformity and increases robustness. However, there are two ways to optimize a treatment plan with multiple fields. Either the fields are optimized independently, leading to a homogeneous dose in each field, or concurrently leading to a heterogeneous dose in each field but a globally more conformal dose. The former is called *single field optimization* (SFO) while the latter is called *multi-field optimization* (MFO). Hence, SFO does not refer to a single field being delivered but to single fields being optimized independently. Therefore, SFO is less conformal but more robust, while the inverse is true for MFO. Most PBS plans are MFO nowadays, which is the approach we take for the remaining of the thesis.

Robust optimization

As mentioned in Section 7.1.2 and Chapter 1, proton therapy is especially affected by uncertainties such as range and setup errors. An optimal plan needs to be *robust* against slight variations from the expected plan due to those uncertainties during treatment delivery, without affecting the general outcome of the treatment as long as the variations stay within the as-

[3]Note that this could be handled in a single step by introducing a binary variable, which would make the problem a mixed-integer nonlinear program which is much harder to solve. The proposed approach on the other hand is only a heuristic that appears to work well in practice.

sumed levels. Different robust optimization strategies have been developed over the past decade to address this problem. Most methods assume that the uncertainties can be well represented by a set of discrete error scenarios that may happen. We define $\mathcal{S}$ as a set of scenarios representing the nominal case plus a number of plausible deviations from the nominal case. Those generally include setup errors, modeled as a shift of the patient or a shift of the beamlets, and range errors modeled as a density perturbation on the CT scan. A simple model of setup errors would contain seven scenarios: a nominal case plus a certain shift in the anterior, posterior, inferior, superior, and left and right directions. A simple model for the range errors would contain three scenarios: the nominal scenario plus a range overshoot or a range undershoot of a specific amount, for instance, by modifying the Hounsfield unit of the CT scan by a small percentage. Different strategies can directly incorporate this set of error scenarios into the optimization problem. Three main approaches can be distinguished:

Probabilistic approach: Minimizing the expected value of the objective function where each objective is associated with a probability of happening p_s. Mathematically, that is $\min_{\mathbf{x}} \sum_{s\in\mathcal{S}} p_s f(\mathbf{x}; \mathbf{B}_s)$.

Minimax approach: It consists of minimizing the worst-case scenario. Mathematically, that is $\min_{\mathbf{x}} \max_{s\in\mathcal{S}} f(\mathbf{x}; \mathbf{B}_s)$.

Worst-case distribution: A variation of the minimax approach where we consider the maximum with respect to the error scenarios of each objective independently. Mathematically that is $\min_{\mathbf{x}} \sum_k \max_{s\in\mathcal{S}} f_k(\mathbf{x}; \mathbf{B}_s)$. There is also another variation with a voxel-wise maximum where we consider the maximum with respect to the error scenarios of each voxel independently. Mathematically, that is $\min_{\mathbf{x}} \sum_{i\in V} \max_{s\in\mathcal{S}} f_i(\mathbf{x}; \mathbf{B}_s)$.

In this work, we use the minimax approach. Formally, we aim to solve the optimization problem:

$$\min_{\mathbf{x}} \max_{s\in\mathcal{S}} f(\mathbf{x}; \mathbf{B}_s) \tag{7.6}$$

$$x_j \geq l \text{ OR } x_j = 0 \text{ for all } j \in \mathcal{J} \tag{7.7}$$

$$x_j \leq u \text{ for all } j \in \mathcal{J} \tag{7.8}$$

Depending on the number of scenarios taken into account, the size of the optimization problem quickly grows and can become an issue. A beamlet matrix $\mathbf{B}_s$ must, in principle, be computed for each scenario s although

some approximations are usually made to avoid recomputing a completely new beamlet matrix. This optimization problem still remains convex (provided that the disjunctive constraint is handled as described earlier). We can reformulate the problem by the epigraph technique as a constrained optimization problem:

$$\min_{\mathbf{x},t} t \tag{7.9}$$

$$t \geq f(\mathbf{x}; \mathbf{B}_s) \text{ for all } s \in \mathcal{S} \tag{7.10}$$

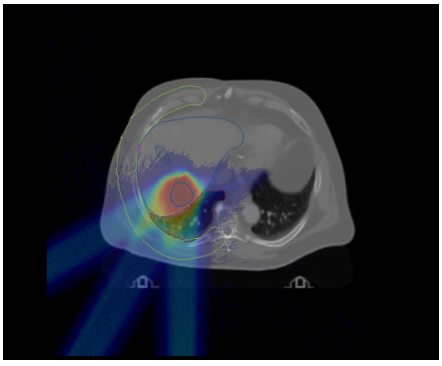
$$x_j \geq l \text{ OR } x_j = 0 \text{ for all } j \in \mathcal{J} \tag{7.11}$$

$$x_j \leq u \text{ for all } j \in \mathcal{J} \tag{7.12}$$

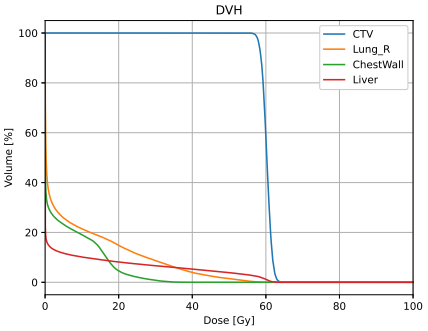
Although optimization problem 7.6 was also constrained, there were only bound constraints easily taken into account by simple methods such as projected gradient methods. This new optimization problem requires dedicated constrained optimization solvers. Sequential quadratic programming has been used (Raystation, Raysearch laboratories) as well as augmented Lagrangian methods (Pinnacle, Philips Healthcare) [Pag18]. For unconstrained optimization (with the exception of bound constraints), solvers typically used for nonlinear objectives are quasi-Newton methods such as the limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) algorithm. If linear objectives and constraints are used, the linear programming framework can be used.

Multi-criteria optimization

The design of the objective function in radiotherapy involves manually selecting weights assigned to each penalty, describing their importance. If the optimization yields unsatisfactory results, re-optimization might be needed with a different set of weights on the objectives. This process can be very time-consuming. Multi-criteria optimization (MCO) aims to solve this issue by aiding the decision process. Two strategies for MCO have been investigated for radiotherapy treatment planning: lexicographic optimization and the Pareto surface approach. In lexicographic optimization, objectives are ordered in terms of importance by solving a sequence of optimization problems while constraining the previous ones to stay within a tolerance of their optimal solutions. In Pareto surface optimization, every objective is treated equally, resulting in multiple optimal plans that trade off the objectives in various ways. On top of the optimization part, one needs to design smooth navigation on the Pareto surface with possibly



(a) Dose distribution on a slice of the CT scan (axial view)



(b) Dose-volume histogram.

Fig. 7.6 Example of treatment verification. On the left, the physician can observe the spatial dose distribution on a slice of the CT scan. On the right, the dose-volume histogram of some VOI is depicted.

many dimensions. For a detailed review of MCO applied to radiotherapy, the reader can refer to [BCVHH19].

7.1.4 Treatment verification

Once a treatment is optimized, it needs to be verified and approved. Physicians use several tools to assess the quality of a treatment plan. The spatial distribution of the dose can be observed by overlaying the dose distribution on the CT scan, as depicted in Figure 7.6a. Another tool used by physicians to verify the quality of the treatment is a dose-volume histogram (DVH). A DVH is a cumulative histogram of the radiation dose received in a volume of interest (VOI), e.g. the target volume or an OAR. It is displayed as a curve with the x-axis being the dose value and the y-axis being the percentage of considered volume receiving this dose value. An example of DVH is shown in Figure 7.6b. Note that the cumulative distribution is from right to left, unlike conventional cumulative distributions. Multiple VOIs can be displayed on the same graph since the y-axis is described in volume percentage. This allows for summarizing the treatment plan quality on a single graph. Finally, a third more quantitative assessment of the treatment plan quality is done via DVH metrics. The metrics generally used are the mean, maximum, or minimum dose in a given volume but also a dose received in a volume percentage. For instance, the metric D_{95} is defined as the minimum dose received in at least 95% of the

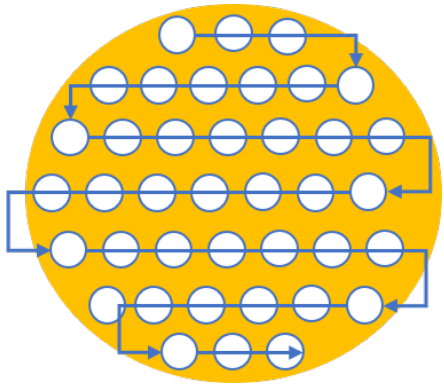


Fig. 7.7 Example of a usual spot pattern for delivering a PBS plan.

volume. Metrics D_{95} and D_5 are usual metrics used by radiation oncologists and medical physicists to compare and approve plans.

7.1.5 Treatment delivery

Once the plan has been approved, the plan will usually be delivered in multiple fractions. The spot weights defined previously are scaled according to the number of fractions. Usually, the plan is delivered in the course of multiple days or weeks and can be adapted if the patient's anatomy changes too much, which would induce a significant dosimetric change. The plan typically only contains instructions on the spot location (position and energy) and intensity (weight). The machine usually handles the order in which the spots are shot. Generally, the spots are delivered layer by layer, moving from the highest to the lowest energy, and the lateral displacement of the pencil beam follows a serpentine pattern as outlined in Figure 7.7.

7.2 Specificities of mobile tumors

Tumors located in the thoracic region, such as lung or liver tumors, are subject to a breathing-induced motion. Special considerations are required for treating those tumors. Two types of motions are encountered clinically:

Inter-fraction motion: Motion happening on a large time scale, across multiple fractions in the course of days or weeks. For instance, this can be due to a displacement of some organs, a change in anatomy due to

weight loss or gain, or an expansion or shrinkage of the tumor size. This type of motion does not strictly concern thoracic tumors.

Intra-fraction motion: Motion happening in a small time scale, in the scale of (a fraction of) seconds during the treatment. For instance, the motion is due to breathing or, to a lesser extent, to cardiac movements.

Those types of motions can deteriorate the treatment quality if they are not adequately taken into account. This thesis focuses on intra-fraction motion and how we can handle and mitigate its negative dosimetric effect. The consequence of this motion affects both the treatment quality of photon and proton therapy, although it affects proton therapy to a bigger extent, as discussed in Section 1.3 and illustrated in Figure 1.2. Two issues arise in proton therapy using pencil beams scanning with tumor motion:

Proton range variation: The effect of tumor motion can lead to a density variation in the beam path, thereby shifting the expected proton range and deteriorating the overall dose distribution. This effect is illustrated in Figure 1.2.

Interplay effect: The tumor and the scanning beam motion happen at the same time scale, leading to an *interplay effect* that can lead to hot and cold spots in the targets, deteriorating the overall dose distribution [SRTP09]. This process is illustrated in Figure 7.8.

7.2.1 Evaluation of tumor motion's outcome on treatment delivery

How can we analyze the impact of motion on the resulting dose distribution before delivery? The answer to this question is via simulations of the treatment delivery based on the 4DCT. Two complementary types of simulations can be conducted: a 4D dose simulation and a 4D dynamic dose simulation. The dose deposition in tissues is generally computed via a Monte-Carlo algorithm simulating beam-matter interactions inside voxelized geometries. An example of an open-source dose engine is MCsquare [SLS16], which is used to carry out simulations in this thesis.

4D dose simulation

A 4D dose (4DD) simulation consists of independently simulating the dose delivery on each motion phase of the 4DCT and computing the average of those doses. Thus, each motion phase is considered *static* and contributes

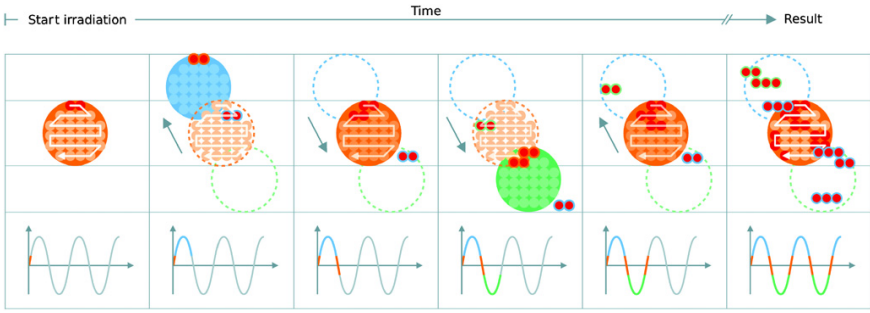


Fig. 7.8 Illustration of the interplay effect. In the presence of tumor motion (bottom signal), the scanned beam delivery results in a deterioration of the dose. During beam delivery, the target moves, represented by the blue, red, and green circles, while the pencil beam is delivered to stationary coordinates. The target's motion thus changes the relative position of the pencil beams in the target. This results in a deteriorated dose, as shown in the last subfigure. Image taken from [BGR08] with permission.

equally to the total dose. Mathematically the 4D dose is given by

$$\mathbf{F}_i = \mathrm{DIR}(\mathbf{CT}_i, \mathbf{CT}_{\mathrm{MidP}}) \text{ for all } i = 1, ..., P \tag{7.13}$$

$$\mathbf{d}_{tot} = \frac{1}{P} \sum_{i=1}^{P} \mathbf{d}_i^{\mathrm{stat}}(\mathbf{v} + \mathbf{F}_i(\mathbf{v})) \text{ for all positions } \mathbf{v} \in \mathbb{R}^3 \text{ in the image.} \tag{7.14}$$

where P is the number of motion phases (generally 8 or 10), $\mathbf{d}_i^{\mathrm{stat}}$ is the dose resulting from the treatment plan simulation on the 3DCT of motion phase i, $\mathrm{DIR}(\cdot)$ is the deformable image registration operator, $\mathbf{F}_i$ is the deformation field resulting from the registration, $\mathbf{CT}_i$ is planning CT image i and $\mathbf{CT}_{\mathrm{MidP}}$ is the Mid-position CT. This resulting dose allows us to analyze the impact of the proton range variation when motion is present. Generally, if this resulting dose is unsatisfactory, a re-optimization with a new set of robustness parameters is needed. In this analysis, we suppose that the entire dose was delivered statically to each motion phase (although scaled by $\frac{1}{P}$); thus, the interplay effect is neglected.

4D dynamic dose simulation

A 4D dynamic dose (4DDD) simulation consists of simulating the dose delivery dynamically. By dynamic, we mean simulating the delivery of beam spots one by one, via a time delivery model, on a sequence of 3DCTs. The sequence of 3DCTs is generally the planning 4DCT, but it could be a

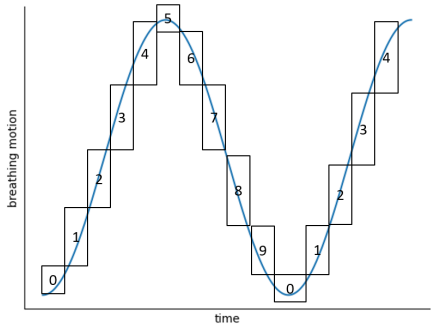


Fig. 7.9 Example of motion phases (0-9) associated with a breathing signal. The motion phases are distributed evenly along one period. A beam spot can be associated with a particular motion phase depending on its delivery timing.

richer set of synthetic CTs representing multiple breathing motions. The time delivery model is a model that associates a timing with the delivery of each spot in the treatment plan. This model is machine-specific and gives an expected delivery timing depending on several parameters such as the delay between the delivery of two adjacent spots, the time required for switching energy, etc. Knowing the delivery timing of each spot in the treatment plan, we can create a breathing model and know exactly in which motion phase each spot will be delivered. The breathing model has two parameters in the simplest case: the breathing period and the number of motion phases of the 4DCT. Mathematically the total dose is given by

$$\mathbf{F}_i = \mathrm{DIR}(\mathbf{CT}_i, \mathbf{CT}_{\mathrm{MidP}}) \text{ for all } i = 1, ..., P \tag{7.15}$$

$$\mathbf{d}_{tot} = \sum_{i=1}^{P} \mathbf{d}_i^{\mathrm{dyn}}(\mathbf{v} + \mathbf{F}_i(\mathbf{v})) \text{ for all positions } \mathbf{v} \in \mathbb{R}^3 \text{ in the image.} \tag{7.16}$$

where $\mathbf{d}_i^{\mathrm{dyn}}$ is the dose associated with the motion phase i that includes all spots predicted to be delivered in that motion phase. This process is illustrated in Figure 7.9. Repeating the simulation by starting the delivery at different motion phases allows for quantifying the uncertainty of the final dose distribution. Note that we consider that the motion phase is a static image; hence we assume that motion is negligible in the time frame $\frac{\text{breathing period}}{\text{\# motion phases}}$. Contrary to the 4DD simulation, each motion phase is only

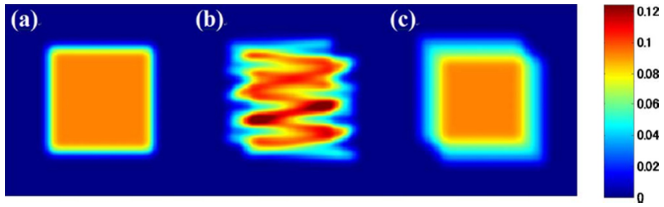


Fig. 7.10 Illustration of the interplay effect with and without rescanning. (a) static dose, (b) dose deteriorated by the interplay effect without rescanning, and (c) blurred distribution by rescanning the target multiple times. Image taken from [FIS+10] with permission.

associated with a subset of the spots of the treatment plan. The 4DDD simulation allows for analyzing the impact of the interplay effect on the treatment outcome.

7.2.2 Motion monitoring and mitigation techniques

Several solutions have been developed in the last decades to handle mobile tumors and mitigate their negative dosimetric effects. Motion mitigation techniques can be classified as online or offline. Online techniques refer to methods that are handled *during* treatment delivery, while offline techniques refer to methods that are handled *before* treatment delivery. They are detailed in the following sections.

Offline motion mitigation techniques

Rescanning

Rescanning, also referred to as *repainting*, refers to delivering the dose in multiple iterations within a fraction in order to smooth the inhomogeneities resulting from the interplay effect by achieving a statistical averaging of the motion effects. Rescanning is generally used when fractionation does not provide enough repetitions to blur the interplay effect [SRTP09]. This process is illustrated in Figure 7.10. Two types of rescanning are distinguished: volumetric rescanning and layered rescanning. In volumetric rescanning, the dose is delivered by multiple rescanning on the entire target volume. In contrast, in layered rescanning, each energy layer is rescanned multiple times before switching to the next energy layer.

Optimizing the beam delivery parameters

The beam delivery parameters such as the spot size, spot spacing, and delivery speed impact the interplay effect. Dowdell et al. investigated the relationship between beam scanning parameters and the interplay effect in [DGSP13]. Large spot sizes ($\sigma \sim 9 - 16mm$) resulted in an improved dose homogeneity in the target compared to smaller spot sizes ($\sigma \sim 2 - 4mm$). Reducing the spot spacing also improved the homogeneity in the target. Longer treatment times generally led to reduced interplay effects.

Another study investigated the optimal spot delivery sequence for reducing the dose uncertainty induced by respiratory motion [LZZ15]. By increasing the area the proton beam covers in a certain period (and therefore reducing the fluence delivered to any given spot in that period), the authors observed that they could reduce the maximum absolute dose error by more than 80%.

In a more recent study, Engwall et al. investigated the inclusion of uncertainties in the time structures of the delivery in the robust optimization framework [EFG18]. They did so by considering multiple scenarios in which the beam spots are distributed to the different respiratory phases of the planning 4DCT. It is an extension of the 4D robust optimization presented in the next section. This approach reduced interplay effects, especially for large tumor motions, when combined with rescanning.

4D robust optimization

Four-dimensional (4D) robust optimization extends robust optimization to account for intra-fraction anatomy changes on top of setup and range uncertainties. Compared to 3D robust optimization, where the target volume is defined as the ITV, 4D robust optimization uses the CTVs of each phase of the planning 4DCT as the target volumes used for the optimization. There are different ways in which the different 3DCTs composing the 4DCT can be handled simultaneously in the optimization. One solution proposed by the original paper on 4D robust optimization [LSC+16] is averaging the dose received by the voxels in each 3DCTs. Mathematically, that is

$$d_i^s = \sum_{p=1}^{N} d_{i_p}^s \tag{7.17}$$

where $d^s_{i_k}$ represents the dose received by voxel i in motion phase p under uncertainty s. The uncertainty set of scenarios is, in this case, the same as for the 3D robust optimization of Section 7.1.3 and the problem can be optimized similarly. Another possibility is to represent each 3DCT as a scenario. Then worst-case robust optimization can also be used similarly with this extended uncertainty set. However, this solution yields a very large uncertainty set if combined with the usual range and setup errors. Assuming we have 7 scenarios for the setup errors, 3 scenarios for range errors, and 10 scenarios for the 4DCT, this results in 210 scenarios, which is computationally challenging. Instead, what is generally performed is to use three scenarios for the anatomical changes, the mid-position CT[4] and the two extreme phases (max exhale and max inhale). This would results in 63 scenarios in our previous example.

Online motion monitoring and mitigation techniques

This section is inspired by the review of Mori et al. on motion management in particle therapy [MKU18].

Online motion mitigation techniques require continuous monitoring of the motion of the patient's anatomy. Two types of monitoring are possible: indirect monitoring based on an external surrogate or direct monitoring based on imaging. In surrogate-based techniques, an external device is used to measure the displacement of the body surface, such as an optical surface imaging system. A correlation model is then necessary to map the skin surface motion and internal tumor motion. While this approach has the advantage of being non-invasive, it lacks the accuracy of direct methods because tumor motion is not always well correlated with surface motion.

Direct methods rely on an imaging device to monitor the motion of the tumor and internal organs. The most used imaging modality is X-ray fluoroscopic imaging, consisting of continuously taking radiographs of the patient's body during treatment delivery. Motion monitoring using imaging can be further divided into two types of tracking methods: fiducial marker tracking and markerless tracking. In fiducial marker tracking, a fiducial marker, built from a material easily observable on X-ray images, is inserted

[4]Synthetic CT representing the time-averaged position of the 4DCT.

on or near the tumor, and 3D coordinates can be retrieved on the X-ray images via a triangulation method. Markerless tracking refers to tracking the tumor without fiducial markers. The technique is obviously more challenging and requires accurate computer vision algorithms to retrieve the tumor location, but it has the advantage of being non-invasive. Markerless tracking is already commercially available for photon therapy (CyberKnife; Accuray, Inc.; Sunnyvale CA, USA) and particle therapy (Toshiba, Tokyo, Japan).

Other imaging modalities include ultrasound (US) imaging and magnetic resonance imaging (MRI). Ultra-sound imaging is a cost-effective solution for providing good soft tissue contrast and is increasingly used in radiotherapy for setup verification and inter-fraction motion [FVdMB+15]. It has the advantage of being non-irradiant, contrary to fluoroscopy imaging. For intra-fraction motion, the technology is in its early stage, and the translation into the clinic is underway [OBF+16]. US imaging has also been shown to be feasible for particle therapy [PKS+14].
MRI is a non-irradiant imaging modality that provides excellent soft-tissue contrast (the best among all aforementioned imaging modalities). MR-linac machines (a machine combining an MRI scanner and a radiotherapy linac) are already commercially available for photon therapy. However, this is not the case yet for proton therapy where this type of machine brings additional engineering challenges due to the effect of the magnetic field on particle beams.

In the next sections, we detail the online motion mitigation techniques currently used in clinical practice.

Abdominal compression

Abdominal compression is a technique for which a compression device is placed on the patient's abdomen that applies a constant force to it in order to reduce the diaphragm motion. An example of such a device is shown in Figure 7.11. It has been shown to reduce organ motion in liver tumors [EPS+11] and some (but not all) cases of lung tumors [BAR+13]. Abdominal compression was also proven effective in reducing the interplay effect for proton beam scanning of liver tumors [SGK+16]. The diaphragm motion is generally verified during treatment by fluoroscopy.

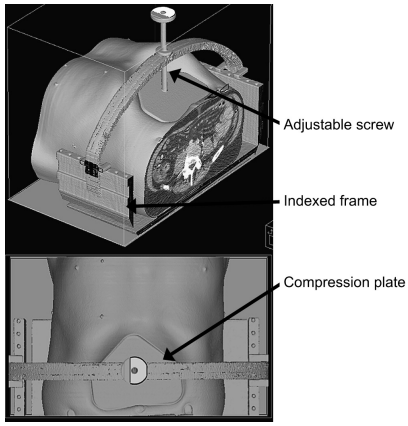


Fig. 7.11 Example of a patient with an abdominal compression device. Image taken from [EPS+11] with permission.

Respiratory gating

Respiratory gating is a method that aims at delivering the treatment beam during a specific respiratory window based on the phase or amplitude of a breathing signal [LBS+07]. The treatment beam is synchronized with a gating signal controlling the beam delivery, which is obtained via imaging or an external surrogate. The beam is thus switched ON only during a specific gating window at the end of the inhale or exhale phase. The gating window can be defined on a single or multiple phases, for instance, representing 30% of the duty cycle, thus trading off between delivery efficiency and dose conformity [MKU18]. Two types of gating exist: phase-based gating and amplitude-based gating. In phase-based gating, the gating window is defined as a part of the respiratory period, while in amplitude-based gating, the gating window is defined as a part of the breathing amplitude or within a specific position defined during treatment planning. Thus, in phase-based gating, there is a risk of constantly missing the target if the amplitude changed or there was a baseline drift. The clinical advantages of respiratory gating are higher dose conformity and better sparing of OARs with the disadvantage of an increased treatment time.

Breath-hold and active breathing control

Deep inspiration breath-hold (DIBH) is a technique that can be used for both moving targets in conjunction with gating in an attempt to decrease tumor motion and for breast or lung cancer to maximize the distance be-

tween the tumor and OARs. DIBH can be achieved in two ways: by repeating voluntary breath-hold or with a computer-controlled device that can assist breath-hold via airway blocking and feedback approaches (e.g. [VODSL+19]). For an in-depth review of clinical applications, the reader can refer to [BHKSC+16].

Tracking

Beam tracking refers to automatically compensating for tumor motion by tracking the tumor and adapting the beam delivery. This would be the optimal motion mitigation technique should it be implemented one day in proton therapy because it would not lead to target expansion (by adding margins to the target volume) or result in an increased treatment time [GRL+06]. Even though adapting the scanning pattern of PBS to follow the tumor motion in real-time might seems doable, beam tracking in proton therapy is very sensitive to position uncertainties and requires hardware adaptations to be able to switch the energy quickly to compensate for range changes [VdWKZ+09]. Moreover, tracking would only be effective if real-time accurate 3D information are available. Hence, tracking in particle therapy still remains a research topic and will take some time before being implemented clinically [MKU18].

8

A library of treatment plans approach for mobile tumors

This chapter is adapted from an article submitted to the Medical Physics journal in 2022 (first revision under review) [HSD+22].

8.1 Introduction

Proton therapy offers a physical advantage over conventional radiotherapy in terms of dose conformity and normal tissue sparing due to the unique depth-dose characteristics of protons, thus potentially improving tumor control while at the same time reducing toxicity [MG17]. However, this precision comes at the cost of high vulnerability to uncertainties. This is especially the case for thoracic tumors, where breathing-induced tumor motion can lead to density variation in the beam path, thereby missing the target or shifting the expected proton range and deteriorating the overall dose distribution. Moreover, with pencil beam scanning (PBS) proton therapy, on top of the proton range variation issue, another dose deterioration can occur due to the interference between the scanning beam motion and the anatomical motion, known as the interplay effect, which can lead to hot and cold spots in the target [SRTP09].

The current state-of-the-art treatment planning approach for dealing with intra-fractional motion is 4D robust optimization which is designed to be robust against small changes in the anatomy by optimizing the worst-case scenario. On top of the usual 3D robustness scenarios, such as range and setup errors, 4D robust optimization is designed to be robust against multiple anatomical variations present in the 4DCT. While this approach is efficient and is the current best practice for treating thoracic cancer [CZK+17], two criticisms can be made: (i) the treatment is only designed to be robust against motions seen in the planning 4DCT, and (ii) robustness against breathing phases increase the dose to surrounding organs at risks (OARs).

Another passive motion mitigation method is rescanning. While rescanning is interesting to mitigate the interplay effect, it does not solve the range variation issue. When tumor motion amplitude is high, active techniques such as gating, abdominal compression, or breath-hold using optical or fluoroscopy imaging are generally used [CEK+21]. A breath-hold approach has the advantage of being simple and easily implemented, with reduced tumor motion and increased tumor-to-OAR distance [HDM+99]. However, this method requires coaching, and not all patients can hold their breath for a long period. Abdominal compression is also a relatively simple method to implement and has been shown to reduce tumor motion [SGK+16]. However, it only mitigates the dose deterioration to a certain degree. Another common active motion mitigation technique is respiratory gating. This has the advantage of increasing dose conformation in the target and sparing OARs. However, this method requires synchronization with tumor motion, which is challenging and increases the treatment time.

Our proposed approach is somewhat an extension of the respiratory gating approach where the beam is ON only when the tumor position is at a close distance to one particular phase of the planning 4DCT. However, instead of optimizing one global treatment plan, our approach consists in optimizing ten separate treatment plans corresponding to each phase of the 4DCT. Then, during treatment, the plans associated with the planning CT phases whose tumor position is closest to the current tumor position deliver their spots.

A similar idea was considered by Graeff [Gra14] and subsequently implemented in a prototype system at GSI Helmholtzzentrum für Schwerionenforschung [LDN+20], where they also use a library of treatment plans.

However, their delivery process is different. They take their decision based on a 1D signal from an external surrogate with a phase-based approach and irradiate in a continuous fashion. Our approach consists in taking decisions based on the frequent acquisition of images and the use of a particular distance metric between the current image and the 4DCT and irradiating in an intermittent fashion. A drawback of the phase-based approach is that the beam might miss the target because the target position in the phase space might not be the same, even though the corresponding respiratory phase is the same. In our method, we base our decision on the distance between the current tumor position and the tumor position in the planning 4DCT. The distance metrics tested in this paper are the Euclidean distance between the center of mass of the tumors and the Dice similarity index between the tumors. A maximum distance (or minimum Dice) threshold needs to be set by the user, which will trade off between dose conformity and treatment time. Several thresholds are compared in this paper and confronted with inexact tumor positions obtained by adding noise of various amplitudes.

We compare state-of-the-art 4D robust treatment plans against our approach in Section 8.3 by simulating the treatment on a continuous sequence of synthetic 3DCTs generated from cine-MRI sequences of five liver patients. This has the benefit of being as close as possible to a real breathing scenario that can be encountered in practice and that does not necessarily follow the breathing pattern seen in the original 4DCT.

8.2 Materials & methods

8.2.1 Patient data

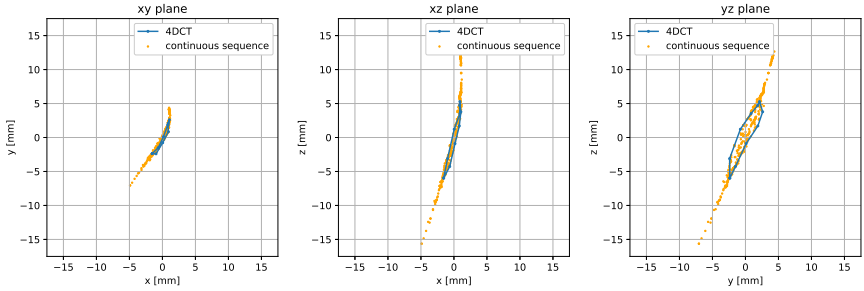
The study includes five patients with liver cancer that were treated with photon radiotherapy for which a planning 4DCT was acquired with abdominal compression and audio-coaching to regularize the breathing pattern. Those patients were replanned in this study for proton therapy with the characteristics described in Section 8.2.2. The particularity of this study is that on top of the planning 4DCT, the patients underwent a dynamic MRI scan. The MRI sequence was acquired the same day and under the same conditions as the 4DCT, with abdominal compression and audio-coaching at a rate of 1.85Hz during 2 minutes. The 2D dynamic MRI sequence can then be used to generate a continuous sequence (CS) of synthetic 3DCTs

Pt. nb.	Tumor volume [cm^3]	P2P motion amplitude in 4DCT [mm]			P2P motion amplitude in CS [mm]: Mean ± std (max)		
		LR	AP	CC	LR	AP	CC
1	29.73	7.5	8.4	10.7	2.8±0.6 (4.2)	2.7±0.6 (4.0)	3.5±0.9 (5.0)
2	9.41	1.2	3.4	7.9	1.6±0.1 (2.0)	3.1±0.2 (3.6)	10.6±0.8 (12.7)
3	8.12	0.8	5.0	8.6	0.7±0.1 (0.8)	4.3±1.5 (5.7)	10.5±2.2 (13.1)
4	82.40	1.1	7.4	7.8	1.2±0.8 (3.8)	3.5±2.9 (12.0)	8.5±5.1 (19.8)
5	22.62	2.6	5.0	11.3	2.9±0.9 (5.8)	5.4±1.1 (9.0)	13.0±2.5 (20.3)

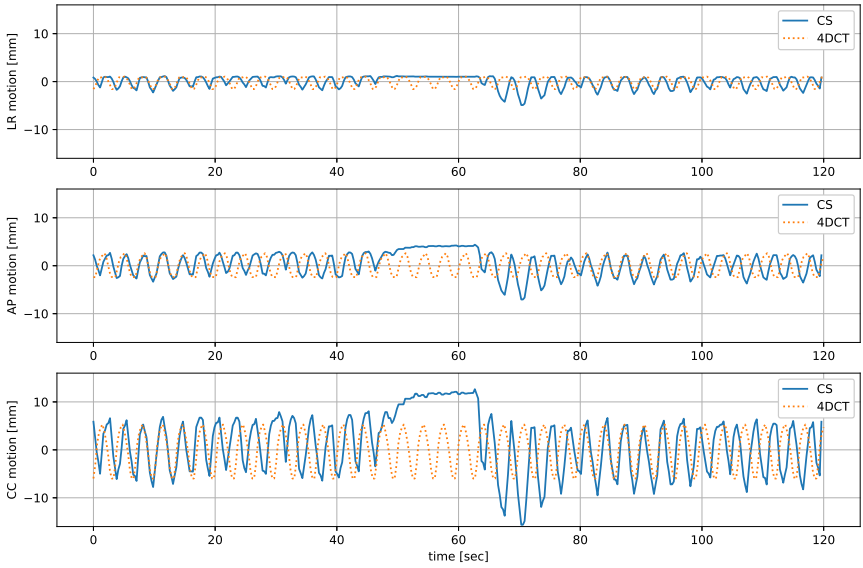
Table 8.1 Information on the tumor volume and peak-to-peak (P2P) motion amplitudes for the five patients in the case study. The motion amplitudes are given for both the 4DCT and continuous sequence (CS) in the three main directions: Left-right (LR) direction, Anterior-Posterior (AP) direction, and Cranio-Caudal (CC) direction.

with the method of Dasnoy et al. [DSSVOM20, DSASM22]. This continuous sequence represents a real breathing pattern of the patient, on which we can simulate the treatment and compute the accumulated dose. Patients' tumor information are detailed in Table 8.1. We can make several observations from those data. Patient 1's motion amplitude observed in the continuous sequence is much lower than the motion amplitude observed in the 4DCT. Indeed, the motion amplitude in the 4DCT overestimates the motion by roughly a factor of 3. Motion amplitudes for the rest of the patients remain globally similar on average (within 2-2.5mm) between what is observed in the 4DCT and the continuous sequence. However, the maximum peak-to-peak amplitudes can be quite different (e.g. for patients 4 and 5). Those differences in motion amplitudes are not uncommon. Indeed, in a recent study, the authors observed a maximum amplitude deviation from the 4DCT $>$ 5 mm for liver tumors in 67% of the cases and a maximum amplitude deviation from the 4DCT $>$ 10 mm in 22% of the cases in the CC direction [DVB$^+$18]. However, the average amplitude deviation from the 4DCT $>$ 5 mm (and $>$ 10 mm) was observed for only 6% of the liver tumors, so patient 1 shows a less common amplitude variation.

An example of tumor motion that we can observe in one of the patients is depicted in Figure 8.1. We observe that the tumor location in the continuous sequence does not necessarily stay within the path formed by the 4DCT motion. Hence, some motions will not necessarily be taken into account even with a 4D robust optimized treatment plan. The tumor motions for the four other patients are available as supplementary materials for completeness (Figures B.1-B.4 in appendix B).



(a) Motion of the tumor in two dimensions in the three principal planes (xy plane, xz plane, and yz plane).



(b) Motion of the tumor in 1 dimension as a function of time in the three principal directions: left-right (LF) direction, anterior-posterior (AP) direction, and cranio-caudal (CC) direction.

Fig. 8.1 Motion of the center of mass of the tumor in the 4DCT (looped over two minutes) and the continuous sequence (CS) for patient 5.

8.2.2 Treatment planning

The conventional approach for designing a treatment plan is to encode desired physical constraints in an objective function to be minimized using a dose-influence matrix calculated on a reference CT image. Since proton dose deposition is subject to uncertainties due to density variations in the beam path and positional errors, we must make the optimization robust to those uncertainties by optimizing the worst-case scenario. Taking into account range errors and setup errors is generally referred to as 3D robust optimization, while additionally taking into account anatomical variations from the 4DCT is referred to as 4D robust optimization.

In our approach, instead of optimizing a single treatment plan on a reference CT, we optimize 10 treatment plans, one on each phase of the 4DCT. There are several ways one can achieve that. The optimal way would be to concurrently optimize the ten plans so that the most optimal beam configuration is delivered in each phase to take advantage of the movement characteristics and the different tumor-to-OARs configurations. This approach leads to heterogeneous dose distribution in each plan that sums up to the theoretically best achievable dose distribution. Although this approach sounds appealing, it is a relatively dangerous idea as it would lead to a heterogeneous dose distribution if some anatomical configurations do not eventually appear during treatment, not even mentioning the additional computational burden of optimizing a huge plan while handling deformable image registration during the optimization. Instead, we propose to seek a homogeneous dose in each plan by optimizing a plan independently on each phase of the 4DCT with the same objective function, which is also one of the proposed methods by Graeff [Gra14]. The downside of this approach is that the total number of spots to deliver is increased since we optimize 10 plans instead of one, but this also leads to an intrinsic rescanning, which can mitigate the interplay effect.

The library of treatment plans can be defined mathematically for each CT_i ($i = 1, .., P$) in the planning 4DCT as

$$\min_{\mathbf{x}^{\mathrm{CT}_i}} \max_{s \in \mathcal{S}} f(\mathbf{x}^{\mathrm{CT}_i}; \mathbf{B}_s^{\mathrm{CT}_i}) \tag{8.1}$$

$$x_j^{\mathrm{CT}_i} \geq P \cdot n_f \cdot m \text{ OR } x_j^{\mathrm{CT}_i} = 0 \text{ for all } j \in \mathcal{J} \tag{8.2}$$

where f is the objective function common to the ten problems, $\mathbf{B}_s^{\mathrm{CT}_i}$ is the beamlet matrix associated with the i^{th} CT under scenario s and $x_j^{\mathrm{CT}_i}$ the variables to optimize for that CT, which are the weights associated with beamlet inDices $j \in \mathcal{J}$. This is the common worst-case robust optimization formulation for radiation therapy; only this has to be solved on each CT. This paper uses a set of scenarios $\mathcal{S}$ of $3 \times 7 = 21$ scenarios: those combine 3 density variations (nominal and $\pm 3\%$) and 7 setup variations (nominal and a shift of $\pm 2.5mm$ in each of the three dimensions). Other uncertainty scenarios could be included such as uncertainties linked to the beam characteristics, possibly obtained from the calibration results of Chapter 6. However, the more uncertainty scenarios are included, the more computationally challenging the planning becomes. The weights associated with each beamlet must also either be zero or satisfy a lower bound constraint, which corresponds to the number of plans ($P = 10$ in our case) multiplied by the number of fractions n_f and the minimum number of monitor units m, the machine can deliver. Note that this lower bound constraint is more restrictive than for a conventional treatment plan (10 times greater) since each plan is designed to deliver 10% of the dose in our case. However, this more restrictive constraint will lead to a sparser treatment plan since more weights will need to be set to zero by the optimizer to achieve higher weights elsewhere. The disjunctive constraint (8.2) is handled in two steps. First, the optimization problem is solved for $\mathbf{x} \geq 0$, then only the spots satisfying $\mathbf{x} \geq P \cdot n_f \cdot m$ are kept, and the optimization is refined by applying $\mathbf{x} \geq P \cdot n_f \cdot m$ as a hard constraint to the spots kept from the first step.

The physical constraints expressed as penalties in the objective function f to guide the optimization are defined in this paper as follows:

- Target: $\min \mathbf{d}_T \geq p$ and $\max \mathbf{d}_T \leq p$ where $\mathbf{d}_T$ is the dose on each voxel in the target and p is the prescription, which in our paper is chosen to be $p = 60$Gy.
- OARs: $\max \mathbf{d}_{OAR} \leq \frac{2}{3} p$ for all organs at risk.

Furthermore, we emphasize the target constraints by weighting the target constraints 10 times more than the OARs constraints. The constraints in the objective functions are usually encoded as asymmetric quadratic penalties, although this depends on the treatment planning system (TPS) used.

Concerning the grid spacing parameter to solve equation (8.1), we choose a

constant spot spacing of 5 mm and a constant layer spacing of 5 mm (water equivalent distance). A special consideration for our approach is that the treatment plans need to share the same energy layers to avoid constantly switching energy during delivery, which would significantly increase the treatment time. Hence, after the first optimization is carried out, we enforce the next plans to reuse the same energy layers. Nevertheless, energy layers that would be needed outside the current range of energies in other plans can still be added, and energy layers not required in a plan can be removed in that plan. In the end, most energy layers will be shared across plans, with only a few used only by extreme phases.

In Section 8.3, we compare the static and dynamic dose delivered by a 4D robust plan against our approach. For the library of plans approach, the static dose is defined as the sum of optimized doses from each phase plan deformed on the reference CT. Mathematically, this is

$$\mathbf{F}_i = \mathrm{DIR}(\mathbf{CT}_i, \mathbf{CT}_{\mathrm{MidP}}) \text{ for all } i = 1, ..., P \tag{8.3}$$

$$\mathbf{D}_{\mathrm{MidP}} = \sum_{i=1}^{P} \mathbf{D}_i(\mathbf{v} + \mathbf{F}_i(\mathbf{v})) \text{ for all positions } \mathbf{v} \in \mathbb{R}^3 \text{ in the image.} \tag{8.4}$$

where $\mathrm{DIR}(\cdot)$ is the deformable image registration operator, $\mathbf{F}_i$ is the deformation field resulting from the registration, $\mathbf{CT}_i$ is planning CT image i, $\mathbf{CT}_{\mathrm{MidP}}$ is the Mid-position CT and $\mathbf{D}_i$ is the optimized dose computed on $\mathbf{CT}_i$. The reference CT is, in our case, the Mid-position (MidP) CT [WSVHD08]. This gives the theoretically best achievable dose by our library of treatment plans.

8.2.3 Treatment delivery

Proton treatment plans are usually delivered without any synchronization, i.e. all the spots are delivered in a continuous fashion layer by layer in a serpentine pattern regardless of the changes in anatomy until all spots are delivered. Our approach is an active technique that irradiates only when the current target position is close to its position in a phase of the planning 4DCT. It is essentially an improved version of respiratory gating with multiple plans computed on each 3DCT. A flowchart representing the treatment delivery framework is depicted in Figure 8.2.

First, an image is acquired from which the tumor location is retrieved. Our

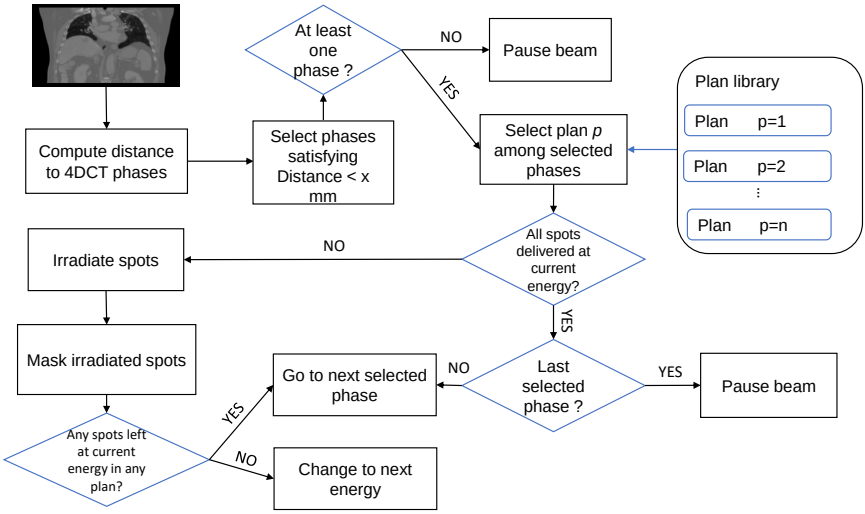


Fig. 8.2 Treatment delivery process of our library of treatment plans approach.

approach is agnostic of the modality used to obtain the tumor location information. This could be via fluoroscopy imaging with a fiducial marker [MMA+19], markerless tracking [HSTM19], or even magnetic resonance imaging. As long as we receive accurate information on the location of the tumors a few times every second, our approach is applicable. We could also eventually use an external surrogate with a tumor correlation model, although the indirect measurements make this solution less accurate. Hence, we suppose that we get a possibly noisy observation of the tumor location. We then compute a distance between the current tumor position and the tumor positions in each phase of the 4DCT. Two distance metrics are detailed in Section 8.2.4 for this purpose. The phases in which the target position is closer to the current target position than a certain threshold are selected, and the plans deliver their spots at the current energy until the next image is acquired or until no more spots remain in the selected plans at that particular energy. The order in which those plans are delivered depends on their completion progress, i.e. the plan with the most remaining spots will be delivered first. The reason for prioritizing the less delivered plan instead of the more similar one is to decrease treatment time. Once a new image is acquired, the process restarts. When all plans have delivered all their spots at the current energy, the energy is lowered to the next energy layer. If no plan satisfies the distance constraint,

the beam is paused until the next image acquisition. Finally, to avoid unreasonable treatment time, if no spots were irradiated for more than 10 consecutive seconds, the distance threshold is gradually increased. In our simulation, we choose to increase the distance threshold to 5mm after 10 seconds and 10 mm after 20 seconds and decrease it again to the original distance threshold after the current energy layer is completed.

Initialization phase

The tumor motion observed in the planning 4DCT might be different than the one observed during the day of the treatment. There might be a difference in breathing amplitude, or there could be a baseline shift [ZWZ+22]. To account for the true breathing pattern of the day, we implemented an initialization phase during which we observe the tumor motion 30 seconds before starting the beam. During this period, images are acquired in the same way they are acquired during treatment, i.e. a few images per second. For each image, the tumor location is computed, and the distances to the ones from the 4DCT are also computed. The phases whose distance is below the user-defined threshold are recorded for all images in the initialization phase. Finally, only the phases that appear regularly are kept for the treatment delivery, and their weights are rescaled accordingly to meet the dose prescription. The occurrence of phase p_i during the initialization phase is computed as follows:

$$O_{p_i} = \frac{100}{N} \cdot \#\left\{\text{dist}\left(c\left(\text{im}_j\right), c(p_i)\right) < D, i = 1, ..., N\right\} \tag{8.5}$$

where N is the number of images acquired during the initialization period, $c(\cdot)$ is a function computing the location of the center of mass of the tumor, dist is the Euclidean distance, im_j is the image j acquired during the initialization phase and $\#\{\cdot\}$ is the counting operator. If $O_{p_i} > 5\%$, the plan corresponding to phase p_i is selected for the treatment delivery, that is with at least half of the *normal* occurrence of a phase in the 4DCT.

After the selection process, the weights of the selected plans must be rescaled to reach the prescription dose. Let n_p be the number of selected plans from the initialization phase. Then the weights of each of the selected plans are rescaled as follows:

$$\mathbf{x}^{CT_i} \leftarrow \frac{10}{n_p}\mathbf{x}^{CT_i} \tag{8.6}$$

Indeed, since the dose in each plan is homogeneous, we can simply rescale all the weights by a constant so that their sum reaches the prescription dose.

8.2.4 Choosing the distance metric

What information do we need or should we use for deciding which plan to choose for irradiating the patient at time t? A straightforward measure to use is the Euclidean distance between the center of mass of the tumor in the current image and the 4DCT. In this study, we consider the distance between the center of mass of the tumors, but this could also very well be the location of a fiducial marker for marker-based tracking. The most accurate information is naturally 3-dimensional, but we can also consider a 2-dimensional or even 1-dimensional location which would include the principal amplitude variations. In this study, we consider a 3D information with various noise amplitudes. Another theoretically more accurate distance metric is the Dice coefficient. The Dice similarity coefficient, first introduced by Dice [Dic45] and subsequently adapted for segmentation [ZDMP94], is used as a measure of similarity between two binary masks, i.e. the segmentation of the tumor in the current image and the segmentation of the tumor in each phase of the 4DCT. It gives a score between zero and one (0 meaning no overlap and 1 means complete overlap) and is computed as

$$\text{Dice} = \frac{2|A \cap B|}{|A| + |B|} \tag{8.7}$$

where A and B are two binary masks. The two metrics are compared in Section 8.3.2.

8.2.5 What image acquisition frequency do we need?

Every time we receive an image, we can make a decision: whether or not to irradiate and which plans to shoot. Therefore, we need to know the optimal frequency to acquire images. Too few acquisitions would lead to less accurate and potentially longer treatments, while too many acquisitions might be undesirable in terms of radiation exposure for imaging modalities such as fluoroscopy and even unnecessary in terms of performance. Indeed, the tumor movement becomes negligible at smaller time scales. On top of the decision-making, the frequency rate has an impact on the accuracy of the treatment delivery simulation. Different acquisition frequencies are tested in Section 8.3.3, some requiring upsampling the continuous

sequence of CTs. The upsampling is done by applying a deformation field $\mathbf{F}_i$, previously computed on each pair of images that follow each other in time, multiplied by a constant $c \in]0,1[$. Mathematically, that is

$$\mathbf{F}_i = \mathrm{DIR}(\mathbf{CT}_i, \mathbf{CT}_{i+1}) \tag{8.8}$$

$$\mathbf{V}_i = \log(\mathbf{F}_i + Id) \tag{8.9}$$

$$\mathbf{CT}_{i+c} = \mathbf{CT}_i(\mathbf{v} + \exp(c\mathbf{V}_i(\mathbf{v}))) \text{ for all } \mathbf{v} \in \mathbb{R}^3 \tag{8.10}$$

where $\mathbf{CT}_i$ is image i, $\mathbf{CT}_{i+c}$ is an interpolated image between i and $i+1$, $\mathbf{V}_i$ the velocity field between phase i and $i+1$, and Id the identity transform ($Id(\mathbf{v}) = \mathbf{v}$). The log and exp are operations on the vector fields (rather than element-wise operations) as described in [VPPA09].

8.2.6 Simulation

Treatment plans were designed with Raystation 10B [Bod18], while the simulation of the delivery of the treatment plans on the continuous sequence of synthetic CT was carried out in our in-house treatment planning system. The computation of the spot delivery timings was done with the IBA ScanAlgo simulation tool emulating the delivery timings on an IBA C230 cyclotron, while the simulation of the dose deposition on the continuous sequence of 3DCTs was done with the Monte Carlo dose engine MCsquare [SLS16], which was recently validated for clinical use [DYS+20]. The dose distributions in the CTs of the continuous sequence are then accumulated on the MidP CT via deformable image registration, similarly to equations (8.3-8.4).

We compare the dose deposited by a 4D-robust plan against our library of treatment plans method, both simulated on a continuous sequence of around 400 3DCTs (see Section 8.2.1) representing 2 minutes of breathing. If the duration of the treatment is longer than 2 minutes, we assume that the breathing restarts in a loop from the first image. Our method is tested under various distance thresholds and noise levels as well as different starting points on the continuous sequence to assess the statistical variability. We assess the performance of our method based on three complementary metrics: the homogeneity in the target, the mean liver-CTV dose, and the treatment time. Homogeneity is defined as

$$\text{homogeneity} = 1 - \frac{D_5 - D_{95}}{\text{prescription}} \tag{8.11}$$

where D_{95} and D_5 are the lowest dose received at least 95% and 5% of the target volume respectively, and the prescription is in this study fixed at 60Gy. This metric gives a score between 0 and 1, where 1 represents a (nearly) perfect homogeneity in the target. The OAR sparing is in this study expressed as the mean liver-CTV dose where liver-CTV represents the volume subtraction between the liver and the CTV. Indeed, according to the consensus report from the Miami Liver Proton Conference, minimizing the mean liver dose and the volume of uninvolved tumor is of extreme importance for any liver radiotherapy [CKK+19]. Finally, treatment time must also be taken into account as a trade-off with the target coverage.

Simulating noise on the location of the tumor To simulate imperfect information on the position of the tumor, we add Gaussian noise to the location of the center of mass of the tumor taken from the images of the continuous sequence. The noise is designed to represent an average distance error to the actual true position. Those noises are designed to follow a multivariate normal distribution $\mathbf{e} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$ of zero mean and covariance matrix $\mathbf{\Sigma} = \sigma^2\mathbf{I}$ with $\sigma = Cd$ where d represents the average distance error and $C = \frac{\sqrt{\pi}}{2\sqrt{2}}$ a normalizing factor that is needed to convert a 3D positional error to a distance error [1] The impact of different noise amplitudes on the final dose distribution is given in Section 8.3.4.

8.3 Results

8.3.1 Treatment plan optimization

For each of the five patients detailed in Table 8.1, we compute one 4D robust treatment plan and ten 3D robust treatment plans on each phase of the 4DCT. We report in Table 8.2 the total number of spots in the 4D robust plan and the library of treatment plans, as well as key dose-volume histogram (DVH) metrics on the static doses.

From those results, we can already conclude that the theoretically best achievable multi-gating solution surpasses the conventional 4D robust plan in both the target homogeneity and the surrounding OAR dose reduction. However, the total number of spots is, on average, roughly increased by a factor of 3, which remains acceptable considering that there are ten plans

[1] The derivation of C is given in appendix A.

Pt. nb.	4D robust					Library of plans				
	# spots	D_{95} [%]	D_5 [%]	Homog-eneity [%]	Liver-CTV D_{mean} [Gy]	# spots	D_{95} [%]	D_5 [%]	Homog-eneity [%]	Liver-CTV D_{mean} [Gy]
1	3936	97.3	104.8	92.5	3.9	11271	97.6	102.8	94.9	3.0
2	2140	98.0	103.3	94.7	7.5	6292	99.0	102.4	96.6	6.5
3	2178	98.5	103.7	94.9	2.1	6573	100.1	103.7	96.5	1.6
4	7615	97.2	105.0	92.2	6.6	21226	97.5	103.0	94.5	5.4
5	4574	98.2	104.2	93.9	3.2	12009	99.8	101.6	98.2	3.0

Table 8.2 Optimized treatment plans characteristics. Comparison between the 4D robust approach and the library of plans static dose characteristics and the number of spots.

in the library. In the following sections, we simulate our approach dynamically on a continuous sequence of CTs to evaluate its capabilities.

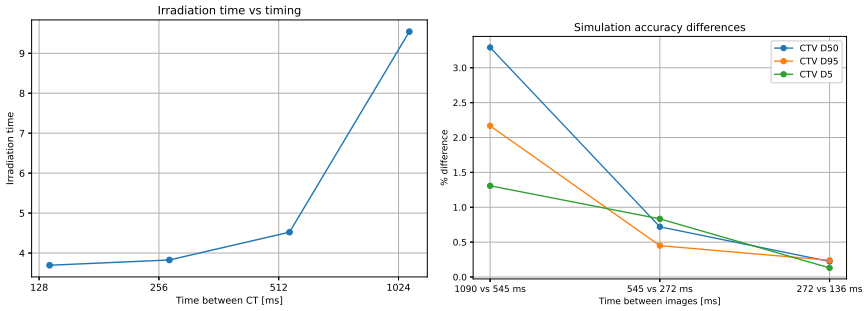
8.3.2 Choosing the distance metric

We compare two metrics: the Euclidean distance and Dice similarity coefficient in Table 8.3 under perfect knowledge of the tumor position. First of all, we observe that our approach has a better dose homogeneity in the target as well as a lower mean liver-CTV dose in all cases compared to the 4D robust plan. However, treatment time is greater, as expected, for two reasons. First, the duty cycle of the beam will always be lower than the one of the 4D robust approach, which is 100% since the plan is delivered without interruption. Secondly, there are more spots to deliver in our approach since there are multiple plans. However, while there are more spots to deliver, there is also less dose to deliver per spot and less dose to deliver globally (since margins are smaller). It is therefore likely that as we go towards more hypofractionation, the time gained from a plan with fewer spots will decrease considerably while benefiting from a plan with less dose to deliver.

Concerning the comparison between the use of the Euclidean distance between centers of mass and the use of the Dice similarity measure, the two approaches give qualitatively similar results. It is hard to give a definitive answer as the decision parameter, i.e. the Euclidean distance threshold and the Dice threshold, are not directly comparable. However, no approach seems to be clearly superior to the other. Hence, because the distance to

	CTV D_{95}	CTV D_5	Homogeneity [%]	Mean Liver-CTV dose	Treatment time [min]
Distance 1mm	59.0	61.89	95.18	2.52	15.08
Distance 2mm	58.88	61.60	95.46	2.55	4.51
Distance 3mm	58.41	61.89	94.20	2.59	3.83
Dice 0.75	57.98	62.95	91.72	2.57	3.51
Dice 0.85	58.53	61.62	94.85	2.64	3.90
Dice 0.95	58.97	61.53	95.73	3.49	21.37
4D robust	56.76	64.21	87.58	3.33	2.19

Table 8.3 Comparison between distance to the center of mass and Dice similarity measure for decision-making. Units are in Gy unless otherwise stated. The last row corresponds to the standard 4D robust model for completeness.



(a) Treatment time analysis **(b)** Simulation accuracy analysis

Fig. 8.3 (a) Treatment timing as a function of the image rate. (b) Percentage difference between the values of three DVH metrics for different image rates.

the center of mass is a simpler metric, more easily implemented clinically, and has better threshold interpretability, we choose to keep this distance metric throughout the rest of the paper.

8.3.3 What image acquisition frequency do we need?

To answer this question, we compute three DVH metrics: D_{95}, D_{50}, and D_5 at different image acquisition frequencies. The period of acquisition was, in our case, 545 ms between images. We downsampled to 1090 ms and upsampled to 222.5 and 111.25 ms to study the impact of the acquisition rate.

The results for the different image acquisition rates are depicted in Figure 8.3. In the right figure, we observe a difference of less than 1% for the three DVH metrics between a period of 545 ms and 272ms. Therefore, 545

ms is a good enough image period for the simulation accuracy. However, in the left figure, we observe that an image rate of 545 ms leads to a longer treatment time than a 222.5 ms rate. This is because more decisions are taken, effectively reducing the duty cycle of the beam. Further increasing the imaging rate only decreases the treatment time by a tiny margin, most likely because the tumor movement becomes small, which does not induce a change in the plans selected. Therefore, a period of around 225ms seems to be a good trade-off and will be kept for the rest of the simulations.

8.3.4 Robustness to noise: analysis of one patient

In this section, we study the impact of imperfect information on the tumor position as well as different distance thresholds for decision-making on the final dose. We compare five distance thresholds (1, 2, 3, 4, 5mm), used to select which plans will be delivered, under four distance error scenarios (0, 1, 2, 3mm) on patient 5 and report the results in Figure 8.4 (the results for the other patients are available in the supplementary materials: Figures B.5-B.8 in appendix B). For each threshold-error pair, we ran the simulations from five different starting images within the continuous sequence, giving us a statistical approximation of the uncertainty represented in the box plots. Three metrics are compared: D_{95} and D_5 given in percentage of the prescription and the treatment time.

Looking at the noise-free scenario, we observe that increasing the distance threshold decreases the treatment time and slowly decreases the homogeneity (represented by the gap between the D_{95} and D_5 metric). We also observe that even at a distance of 5mm, we are still delivering a more homogeneous dose to the target compared to the 4D robust plan. Moreover, the uncertainty in the final dose due to the different starting times is low compared to the 4D robust plan.

Looking at the impact of the noise amplitude on the outcome of the simulated treatment, we observe that the dose is slightly deteriorating but staying at a reasonable target coverage, still better than the 4D robust plan. However, the treatment time is impacted by the noise, especially for the 1 and 2mm distance thresholds. Hence, we should aim for a distance threshold of at least 3mm for these parameters. For the location error, depending on the type of tracking chosen, we can expect different levels of noise amplitudes. In the case of fiducial-based tumor tracking, we can ex-

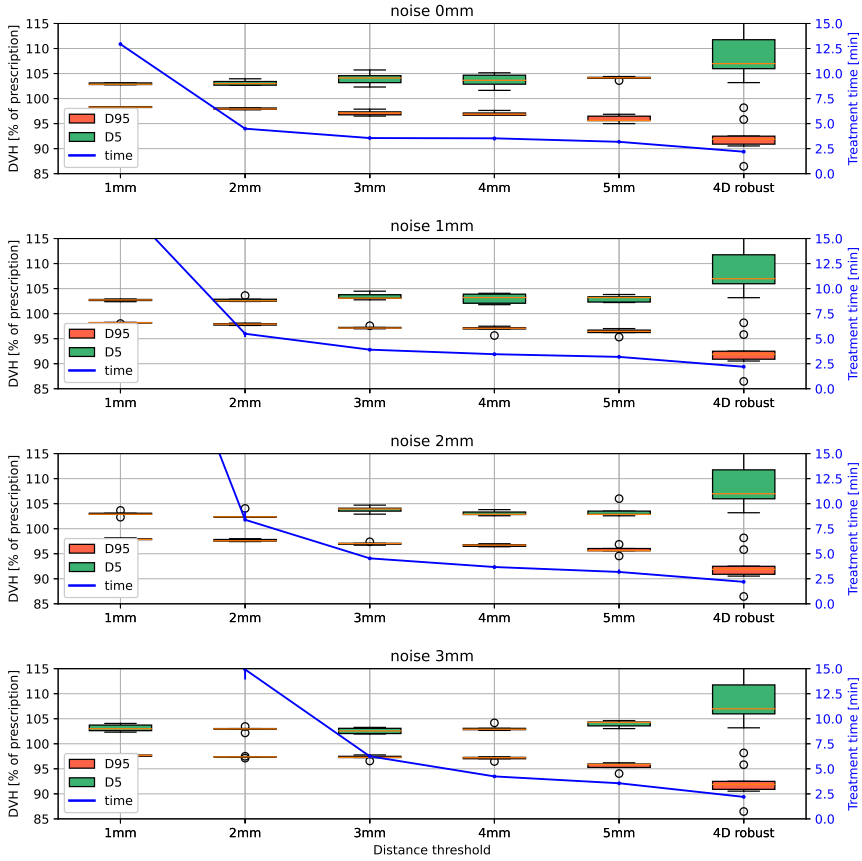
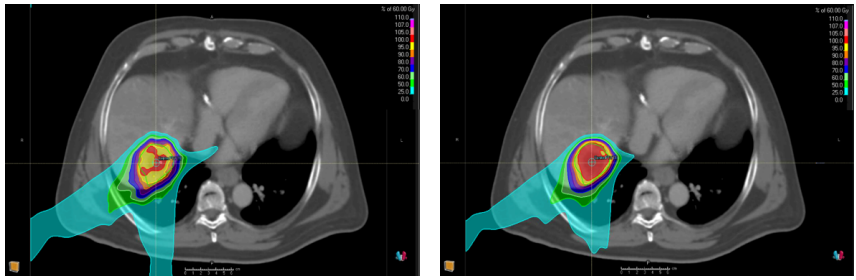


Fig. 8.4 Patient 5 results (DVH characteristics and treatment time) for increasing noise amplitudes on the location of the tumor (top to bottom) and increasing distance thresholds (left to right) for selecting the plans in the library. The distance thresholds are only the initial thresholds, and they follow the increase scheme described in Section 8.2.3, that is, the distance thresholds are increased to 5mm if no spot were irradiated in the last 10 seconds and further increased to 10mm if no spots were irradiated in the last 20 seconds. The 4D robust approach results are repeated on the far right of each subplot. They are not affected by the noise amplitudes since the plan is not synchronized with tumor motion. The boxplots and small vertical bars on the blue solid line account for uncertainties linked to the different starting times for the simulation on the CS.

pect a sub-millimeter accuracy on 2D images and an error within 1mm for three-dimensional calculation error, as reported in a recent study on fidu-



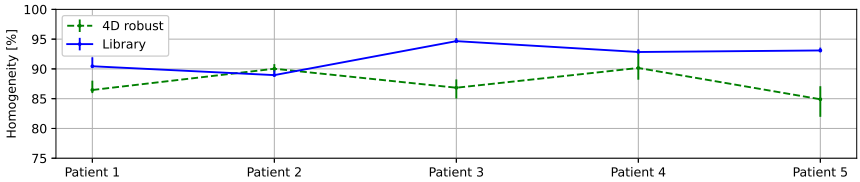
(a) 4D robust approach **(b)** Library of plans approach

Fig. 8.5 Comparison of spatial dose distributions for patient 5 under a noise amplitude of 2mm and distance threshold of 3mm.

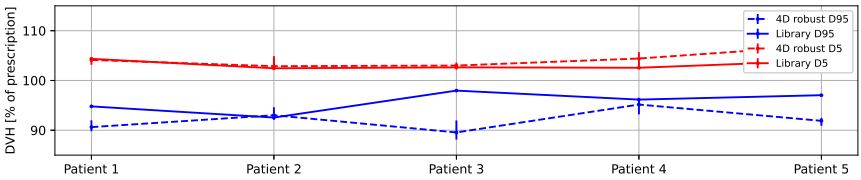
cial markers [MMA+19], hence a noise amplitude $<$ 2mm. For markerless tracking, a recent paper achieved a tracking accuracy of $1.64 \pm 0.73mm$ [HSTM19]; hence, a noise amplitude of 2mm seems to be a correct expected error for today's tumor tracking accuracy. For the patient analyzed in Figure 8.4, this would lead to a treatment time below 5 minutes, which still seems reasonable. The spatial distribution of the dose for this patient under a noise amplitude of 2mm is compared to the 4D robust approach in Figure 8.5

8.3.5 Comparison between all patients

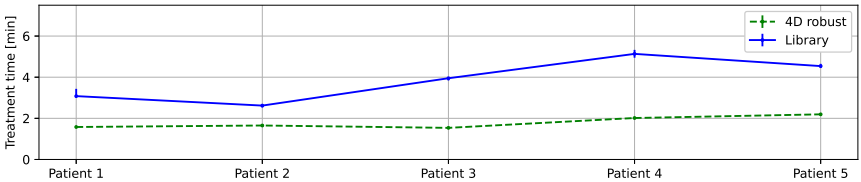
In this section, we compare the results of the simulation study on the five patients described in Section 8.2.1 with a distance threshold of 3 mm and a noise amplitude of 2 mm. During the initialization phase, the number of plans selected, satisfying the constraint $O_{p_i} > 5\%$ (see equation 8.5), are respectively 4, 10, 7, 5, and 10 for patients 1 to 5. Results of the treatment delivery are depicted in Figure 8.6 (Figures B.9-B.13 in appendix B provide additional details on the cumulative DVH results). We observe that the dose is more homogeneous in the target for all patients with our approach except for patient 2, for which the homogeneity is similar (4D robust plan is 1% better). On average, the homogeneity increase was 5% for the library of treatment plans approach. This is not surprising as the interplay effect is greatly reduced with our approach. Moreover, motions not present in the planning 4DCT, and therefore not taken into account by the 4D robust approach, are encountered during treatment delivery, resulting in dose deterioration for the 4D robust approach.



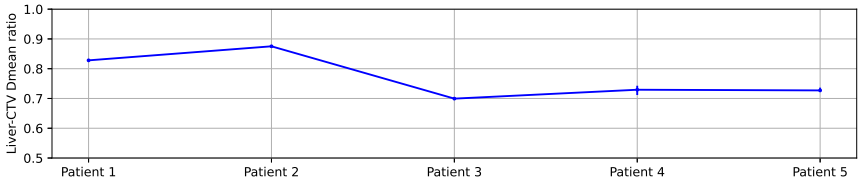
(a) Homogeneity in the target volume (computed according to equation (8.11)). The vertical bars describe the interquartile range (IQR), and the continuous line represents the median value. The closer we are to 100%, the better.



(b) D_{95} and D_5 metrics in the target volume in percentage of the prescription.



(c) Time required for delivering the treatment.



(d) Ratio between the mean doses delivered in the Liver-CTV volume (CTV subtracted from the liver) of the library of plans and the 4D robust approach. A value below 1 indicates an improvement compared to the 4D robust approach.

Fig. 8.6 Simulation results of the treatment delivery on five patients for the 4D robust approach and the library of plans approach. A noise amplitude on the distance to the tumor of 2mm is applied, and a distance threshold of 3mm is chosen. The vertical bars indicate uncertainty linked to the different starting times for the simulation on the CS.

Since margins are smaller with our approach, we also expect that the dose

to the organs at risk will be reduced, which is confirmed in the last plot in Figure 8.6. We computed the ratio between the mean Liver-CTV dose of our approach and the mean Liver-CTV dose of the 4D robust model. We observe that for all patients, the mean dose of uninvolved liver is reduced by 23% on average and up to 30%.

Finally, the treatment time is inevitably increased in our approach. On average, it is increased by roughly a factor of 2, which lead to a mean treatment time of 3.8 minutes and a maximum treatment time of 5.1 minutes, which still seems reasonable. In this treatment time, we also included the 30-second initialization phase described in Section 8.2.3.

8.4 Discussion

Adapting treatment in real-time In this study, we have seen that real-time adaptation of the treatment based on the current anatomy using a library of treatment plans helps to achieve a better dose homogeneity in the target and a significant decrease in the surrounding OARs compared to a state-of-the-art 4D robustly optimized treatment plan. This comes at the expense of an increased treatment time. On average, we manage to limit this increase to a factor of 2. Considering that we have ten treatment plans and an active beam delivery, i.e. a duty cycle strictly lower than 1, this is considered a quite good performance. This was possible because of sparser treatment plans enforced by equation (8.2). Nevertheless, the tumors in this case study were relatively small, and big tumors might pose a problem in terms of increased treatment time. Improvements on treatment plan sparsity are still possible, hence further decreasing the treatment time. One of the possibilities would be regularizing the objective function (8.1) by adding a penalty term on the weights via a ℓ_1 norm. We leave this for a future study. Another approach that could increase the duty cycle is a relative weighting of the plans according to the time spent in each phase. Indeed, it is possible that one phase is available for a longer amount of time than another phase, which could be evaluated during the 30-second initialization phase. This could also provide a gain in dose conformity because our approach prioritizes the less-delivered plan over the most similar one.

Tumor tracking Although we did not restrict ourselves in this study to a specific imaging modality, tracking tumors seems to be best handled with fluoroscopy in today's technology, mainly because this technology

is already mostly integrated into today's proton therapy systems. Indeed, nowadays, proton therapy machines are usually either equipped with a Cone-beam CT imaging system or 2D orthogonal radiographs and can be used for real-time image guidance [Kru18, Kor15, VSV+18]. Moreover, the results obtained in Section 8.3.5 assume a distance error of 2 mm, which is what today's tumor tracking technology can already achieve [MMA+19, HSTM19]. Should our approach be implemented in a proton therapy system, no hardware changes would be needed for the imaging system. However, some changes would be necessary for the treatment delivery system, which would need to be synchronized with the anatomical motion and selection of treatment plans. Such a modular dose delivery system prototype has already been successfully implemented for scanned ion beam [LDN+20] and characterized at GSI Helmholtzzentrum für Schwerionenforschung GmbH and the Centro Nazionale di Adroterapia Oncologica (CNAO).

System latency and tumor motion prediction We have empirically demonstrated that an image acquisition with a period of roughly 250ms was enough for making decisions on the active delivery of the treatment. Decreasing this period only marginally decreases treatment time below this value. However, we neglected the latency of the system, defined as the duration between the start of the image acquisition and the beam delivery or beam off. The system latency includes the image acquisition duration, time for processing the image, time for selecting the treatment plan based on the distance to the current tumor position, and time for switching the beam ON or off. If this time lag is too large, this could affect the accuracy of the dose delivery by irradiating while the target has already moved out of the beam path. AAPM Task Group 264 for the safe clinical implementation of MLC tracking in radiotherapy recommends as a minimum requirement an average system latency $\leq$ 500ms [KSB+21]. This is well over system latencies currently obtained for gated photon and proton therapy treatments. In proton therapy using PBS with real-time imaging and amplitude-based gating, Hokkaido University Hospital PT center reported a system latency of 66ms [SMM+14]. Another group recently commissioned a fluoroscopic-based real-time markerless tumor tracking system for carbon-ion PBS and reported an overall system latency of 70-110ms [MSH+19]. For surrogate-based tracking with optical and electromagnetic technologies, the worst reported system latency was 31 ms [FSC+17]. In our approach, we could expect a similar latency as PBS with gating based

on fluoroscopy imaging, i.e. 100ms up to 200ms if the selection of treatment plans takes more time. Because tumor motion is predictable [VWL+10], the system latency can be compensated with a prediction filter predicting future motion. A larger number of predictive models have been developed in the last decade [VWL+10, EDSS13, JEG+20, LZL+19]. In the latest study [JEG+20] comparing several predictive models, the authors reported a normalized RMSE[2] < 0.05 for a prediction horizon of 160 and 480 ms with a linear filter, meaning that prediction is 95% better than without a prediction (1 corresponds to no improvement and 0 to a perfect prediction). This corresponds to a sub-mm accuracy even for high motion amplitudes. However, the prediction filters in the study used a sampling frequency of 25Hz, while we used a sampling frequency $\sim$ 4Hz. A hybrid motion tracking using fluoroscopy and an external surrogate might help achieve a higher sampling rate and increase the accuracy of motion prediction.

Distance metrics We found in Section 8.3.2, that the center of mass of the tumor was a good proxy to decide which plans to select when acquiring an image. We also considered the Dice similarity index, which provides richer information on the tumor location but did not significantly improve the performance. However, those two metrics only look at tumor information, not the anatomy globally. Other measures that take the global anatomy into account could be tested, such as the mutual information between the current image and the images of the 4DCT. We leave this as future work.

Application to lung cancers In this simulation study, we only looked at liver cancers simply because we had real breathing MRI acquisitions for those patients at our disposal. However, we expect our approach to display even better performance in lung cancer patients because the effect of motion is amplified due to the lower tissue densities encountered for those cancers, making a single treatment plan more prone to dose heterogeneities than our library of treatment plans approach.

Choosing the number of breathing phases: The number of breathing phases and treatment plans in our approach was simply determined by the number of phases in the planning 4DCT. However, we did not investigate the outcome of a higher or lower number of phases. Taking more phases would probably not be a good idea in terms of the negative impact

[2]The normalized RMSE is computed as $\frac{RMSE(y_{orig}, y_{pred})}{RMSE(y_{orig}, y_{delayed})}$

on the treatment time (because more plans would need to be delivered), but fewer phases might offer a better trade-off between treatment time and dose conformity.

Comparison with gated delivery on maximum exhalation or inhalation Our approach is essentially an extension of amplitude-based gating for multiple plans. If the breathing motion during treatment delivery is similar to the one of the 4DCT, then gated delivery on maximum exhalation or inhalation and our approach should both lead to approximately the same treatment time (for the same choice of parameters). However, our approach should lead to a better homogeneity in the target volume because of the intrinsic volumetric rescanning induced by delivering multiple plans. To obtain the same dosimetric outcome, a conventional gated delivery would need several rescanning, which would increase the treatment time considerably. Another disadvantage of gating is the possibility that the plan can never be delivered if there is a baseline shift between planning and delivery. This is the case for patient 1 (see Figure B.1 in the appendix B) in our case study. A treatment plan optimized on max inhalation would be unable to be delivered in that case, while our method allows adapting the plan to be delivered during the initialization phase. Finally, for a gated plan where delivery window is set across several motion phases with the target defined as an ITV, the difference with our approach would be larger margins for the gating solution and therefore higher dose for surrounding organs.

Full concurrent optimization In this work, we built each plan in the library independently. Concurrently optimizing all the treatment plans to take advantage of the movement characteristics would be less robust, especially if a certain motion phase never appears during delivery, as mentioned in Section 8.2.2. However, there would be a gain in terms of treatment time because fewer spots would need to be delivered, and in terms of sparing organs at risks because the optimizer would be able to choose the best tumor-to-OAR configurations to deliver the treatment. The homogeneity in the target would however likely decrease because no intrinsic rescanning would be performed. Quantifying the gain and loss of a joint optimization should be addressed in a future study.

8.5 Conclusion

We developed a real-time image-guided approach for treating moving tumors with a library of treatment plans optimized on each phase of a planning 4DCT. Our approach, simulated with the current accuracy of tumor tracking technology, allows to reduce the dose in the surrounding OARs and improve the dose homogeneity in the target compared to a state-of-the-art 4D robust treatment plan for the five patients in this study. This comes at the expense of an increased treatment time but remains below an acceptable level of 4 minutes on average. Future works include validating our approach on more patients to obtain a higher statistical significance before a possible clinical validation.

9

Conclusions and perspectives

Proton therapy provides a clear dosimetric advantage compared to conventional radiotherapy but is costly and vulnerable to uncertainties. Increasing cost-effectiveness and treatment quality is therefore essential to allow widespread adoption. This thesis focused on three very different problems based on data-driven models to meet some of these objectives.

The first part of this thesis aimed to apply predictive maintenance techniques to predict incoming failures of a central component in the RF system of a proton therapy synchrocyclotron: a rotating condenser. This piece of machinery is one of the main components prone to failure because of its high-speed rotation. The bearing system is replaced on a regular basis, but unforeseen failures might still happen, requiring specialized technicians to quickly intervene for replacing the damaged system. Treatments cannot be performed during the time elapsed between failure and replacement. Hence, they must either be rescheduled or redirected to photon therapy, impacting both the patients and the hospital. Predicting a failure avoids this catastrophic scenario by giving time to prepare for a replacement and reducing the system downtime to a minimum. This objective was achieved in Chapter 4 where 90% of the failures were correctly detected (more than 75% of those detections being between 2 and 10 days ahead of the fail-

ure) with a relatively low false alarm rate ($\pm 5\%$) for the best model, which is currently used in production. The scientific contributions for this part of the thesis include a new feature selection approach for prognostic applications that does not require class labels, which outperforms conventional and specialized feature selection techniques, and a framework for predictive maintenance applications that conjointly uses a learning algorithm with a decision-making step. Future perspectives in this field are twofold. From a theoretical point of view, it would be nice to attempt to merge the two-level predictive maintenance approach of Chapter 4 into a single level of learning, possibly with a stochastic evaluation of the decisions. From a practical point of view, an obvious perspective is applying the predictive algorithms developed for the rotating condenser to other components of the proton therapy system.

In the second part of the thesis, we developed a derivative-free optimization approach for automatically calibrating a proton therapy beamline in order to gain time and save labor costs when installing a new system. The objective was achieved in Chapter 6 where we managed to considerably decrease the time required to calibrate the magnet currents of the beamline. On top of formulating the optimization problem and solving it, the scientific contribution for this part is a transfer learning approach for Bayesian optimization that reduces the number of iterations needed to find a solution for a different configuration of the problem that would typically require a completely new and lengthy optimization procedure.

In the third part of the thesis, we developed a novel approach for treating mobile tumors in proton therapy, using a library of treatment plans where the beam delivery is synchronized with the tumor motion. Because of the significant impact of tumor motion on the uncertainty of the dose deposition in proton therapy, thoracic tumors are difficult to treat within the safety constraints of surrounding organs at risk and tumor control. With the developed approach, the dose deposited in surrounding organs can be reduced to a minimum while maintaining tumor control. In Chapter 8, our approach allowed a decrease of 23% of the mean dose delivered in the liver while providing a more homogeneous dose than the state-of-the-art methods. Further improvements regarding our library of plans approach would be to find a way to compensate for a plan not frequently selected, or even never selected, for instance due to a baseline shift of the tumor, with another plan of the library. However, this is not trivial since the algorithm

would need to adapt the plan's weights in an online way and recompute the dose to ensure its correctness.

The end goal of the data-driven methods developed throughout this thesis should be to aid and augment human capabilities for making decisions, rather than replacing the human component. The success of those methods can only be achieved if they are understood, and if humans are involved in their daily use and operations. In this context, the different approaches proposed in this thesis must integrate this human component, with the final decision incumbent to the user, by providing the necessary tools to aid him or her. For the predictive maintenance of the cyclotron, one way to do that is by providing the health indicator output by the model to the user, and superimposing the optimized threshold on the indicator. This is currently the way it is used, on top of warning notifications sent to the users when the health indicator reaches the threshold. For the beamline calibration, the user should choose the algorithm he sees fit for the current task and be able to tune it when needed, for instance by enforcing tighter or relaxed constraints on the beam. Finally, concerning the treatment of mobile tumors with a library of plans, the radiation oncologist should validate the treatment plans of the library in advance and have the final word on delivering the treatment. The place the human operator occupies is central to suggest improvements, retraining or corrections in case errors are detected.

Even though proton therapy is better at targeting tumors and sparing organs at risks compared to photon therapy, there is still room for improvement to fully exploit the physical advantages of protons. Improvements in planning and delivery are still needed. Indeed, proton therapy lags behind photon therapy; for instance, proton arc therapy[1] is still in a research phase while it is commercially available for photon therapy. Reducing uncertainties is another key aspect that needs to be further addressed. Improving imaging and image-guided proton therapy, which also lags behind conventional radiotherapy, will reduce some of these uncertainties. Finally, there is still a need to further decrease the total cost of proton therapy; this includes decreasing the cost of the accelerator but also improving the efficiency to reduce operating costs. The methods developed in this thesis provide some contributions toward these objectives, helping to achieve efficient and effective cancer treatments based on proton therapy.

[1]Proton arc therapy is a treatment method where the proton beam is delivered continuously as the gantry rotates around the patient.

Appendices

A Multivariate Gaussian noise and distance error

To add a Gaussian noise on a N-dimensional position that represents an average distance error, we need to look at the expected value of the norm of the noise vector.

Let us define $\mathbf{p} \in \mathbb{R}^N$, a position in N dimensions. If we add a Gaussian noise $\mathbf{e} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$ with $\mathbf{\Sigma} = \sigma^2\mathbf{I}$ to $\mathbf{p}$, we obtain a noisy position

$$\mathbf{y} = \mathbf{p} + \mathbf{e}$$

How can we choose σ so that the average distance error is equal to d? We need to look at the expected value of the norm of the noise vector $\mathbf{e}$. Mathematically, that is

$$\begin{aligned}
\mathbb{E}_{\mathbf{e}\sim\mathcal{N}(\mathbf{0},\sigma^2\mathbf{I})}\left\{\|\mathbf{e}\|_2\right\} &= \mathbb{E}_{\bar{\mathbf{e}}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left\{\|\sigma\bar{\mathbf{e}}\|_2\right\} && (\star.1)\\
&= \sigma\mathbb{E}_{\bar{\mathbf{e}}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left\{\|\bar{\mathbf{e}}\|_2\right\} && (\star.2)\\
&= \sigma\mathbb{E}_{\bar{\mathbf{e}}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left\{\sqrt{\sum_{i=1}^{N} e_i^2}\right\} && (\star.3)
\end{aligned}$$

We thus have the square root of a sum of squares of N standard normal random variables, which is a Chi-distribution for which the mean is given by [EHPF11]

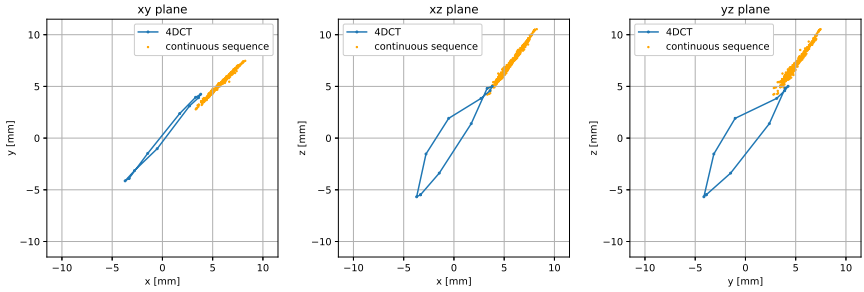
$$\mu = \sqrt{2}\frac{\Gamma\left(\frac{N+1}{2}\right)}{\Gamma\left(\frac{N}{2}\right)} \qquad (\star.4)$$

where Γ is the gamma function and N the dimension. For $N = 3$, we have $\mu = \frac{\sqrt{2}\Gamma(2)}{\Gamma(3/2)} = \frac{2\sqrt{2}}{\sqrt{\pi}}$. Hence, for a 3-dimensional position, if we want the average distance error to be equal to d, we must set $\sigma = \frac{\sqrt{\pi}}{2\sqrt{2}}d$.

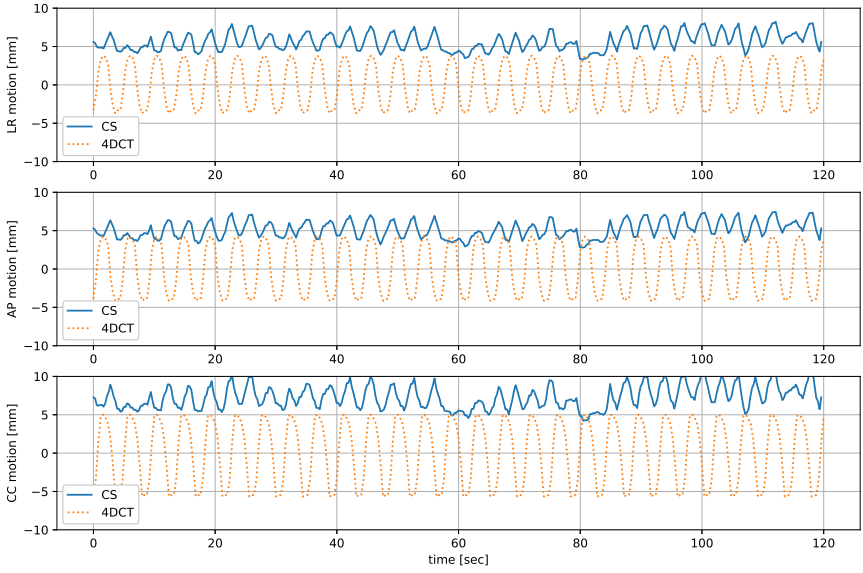
B Additional materials for Chapter 8

Tumor motion

In this section, we provide tumor motion information in the same way as Figure 8.1 for the four other patients in this study.

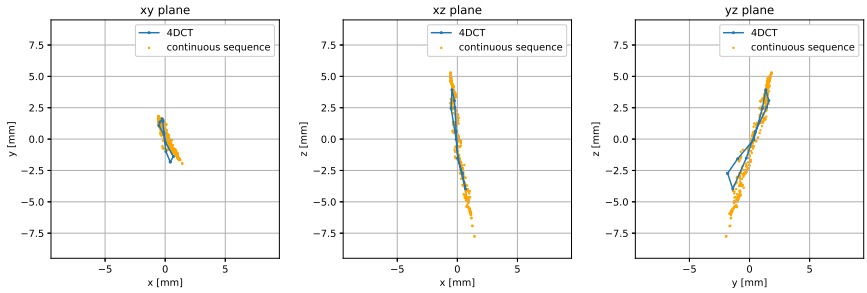


(a) Motion of the tumor in two dimensions in the three principal planes (xy plane, xz plane, and yz plane).

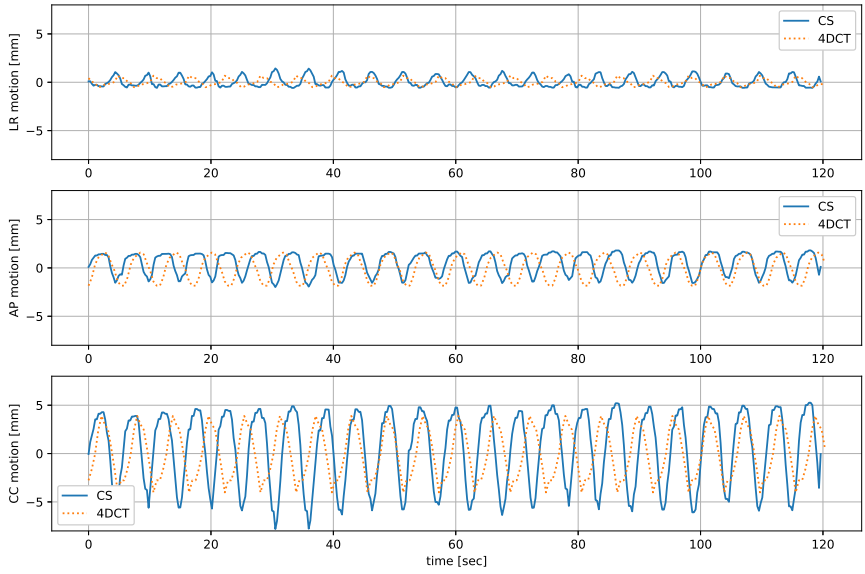


(b) Motion of the tumor in 1 dimension as a function of time in the three principal directions: left-right (LF) direction, anterior-posterior (AP) direction, and cranio-caudal (CC) direction.

Fig. B.1 Motion of the center of mass of the tumor in the 4DCT (looped over two minutes) and the continuous sequence (CS) for patient 1.

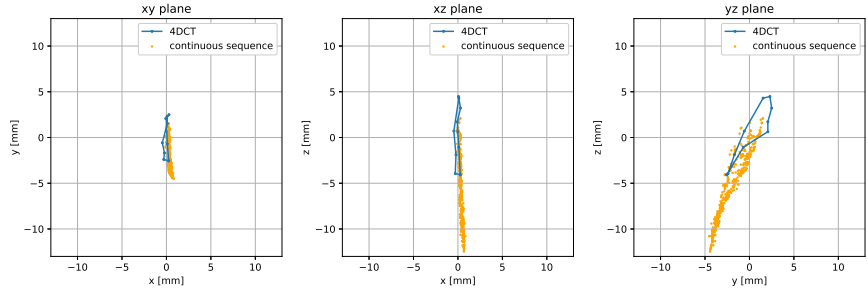


(a) Motion of the tumor in two dimensions in the three principal planes (xy plane, xz plane, and yz plane).

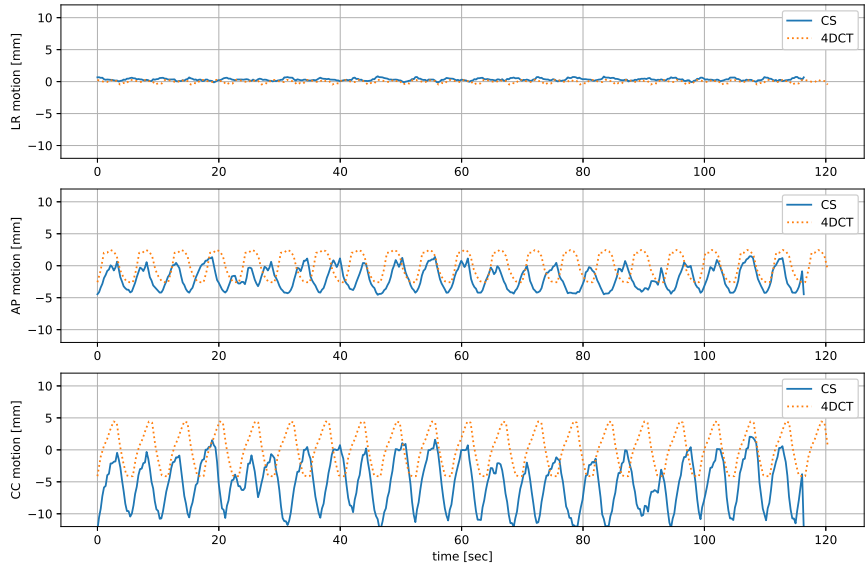


(b) Motion of the tumor in 1 dimension as a function of time in the three principal directions: left-right (LF) direction, anterior-posterior (AP) direction, and cranio-caudal (CC) direction.

Fig. B.2 Motion of the center of mass of the tumor in the 4DCT (looped over two minutes) and the continuous sequence (CS) for patient 2.

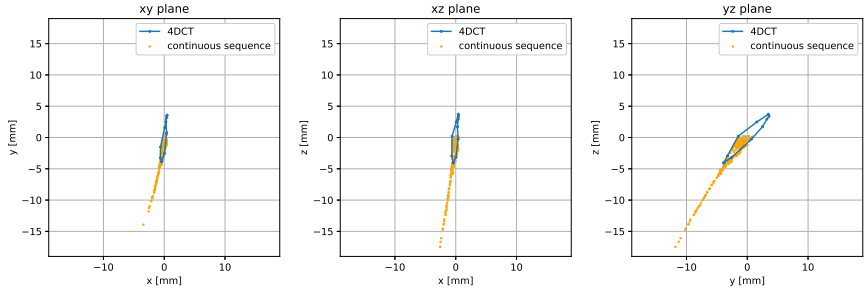


(a) Motion of the tumor in two dimensions in the three principal planes (xy plane, xz plane, and yz plane).

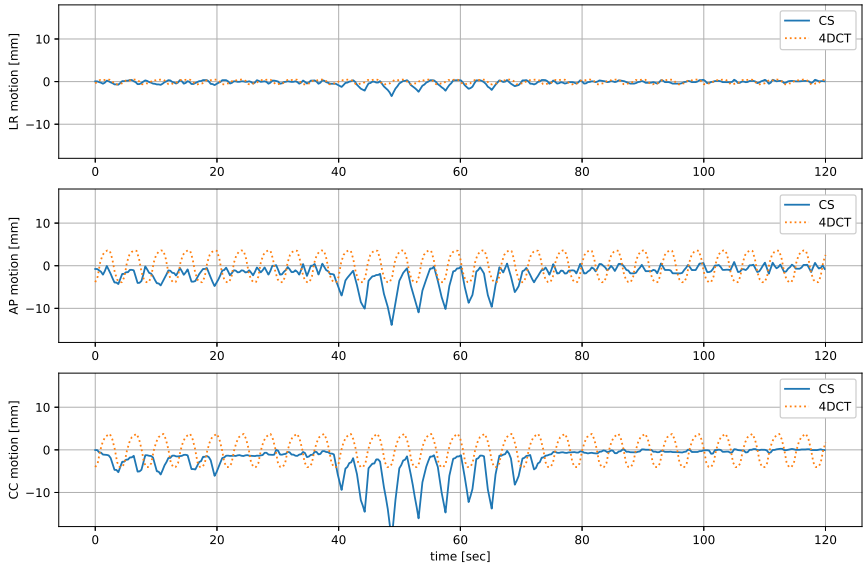


(b) Motion of the tumor in 1 dimension as a function of time in the three principal directions: left-right (LF) direction, anterior-posterior (AP) direction, and cranio-caudal (CC) direction.

Fig. B.3 Motion of the center of mass of the tumor in the 4DCT (looped over two minutes) and the continuous sequence (CS) for patient 3.



(a) Motion of the tumor in two dimensions in the three principal planes (xy plane, xz plane, and yz plane).



(b) Motion of the tumor in 1 dimension as a function of time in the three principal directions: left-right (LF) direction, anterior-posterior (AP) direction, and cranio-caudal (CC) direction.

Fig. B.4 Motion of the center of mass of the tumor in the 4DCT (looped over two minutes) and the continuous sequence (CS) for patient 4.

Variations of simulation parameters and impact on the treatment outcome

In this section, we vary the simulation parameters, namely the noise on the tumor location and the distance thresholds. We repeat the analysis of Figure 8.4 for the four other patients in this study.

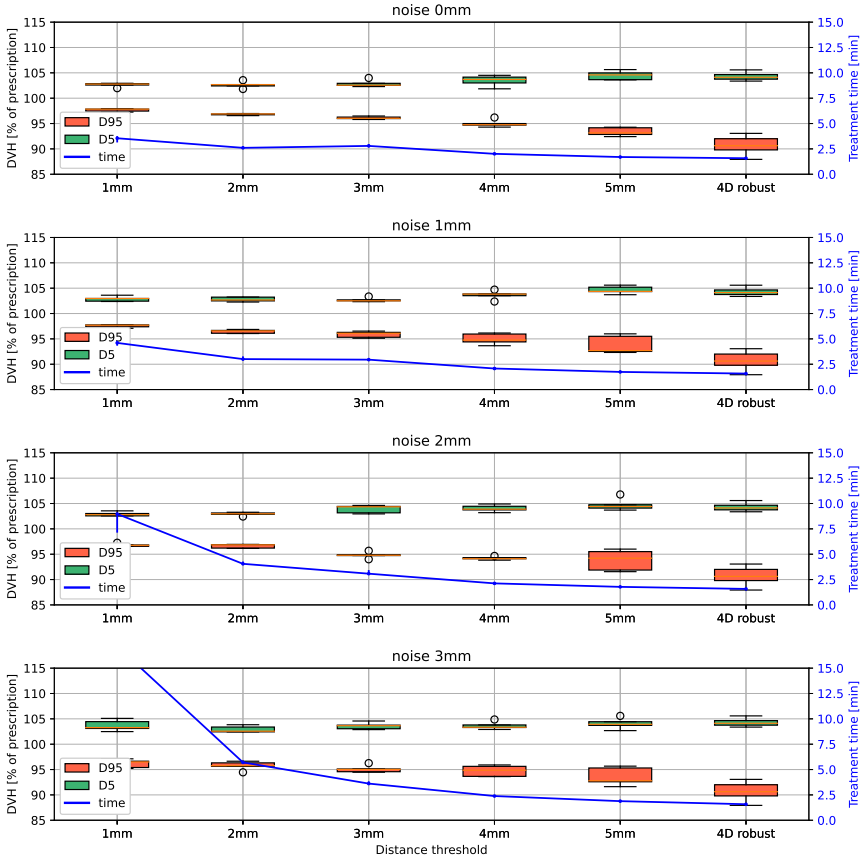


Fig. B.5 Patient 1 results for each noise (top to bottom) and distance thresholds (left to right).

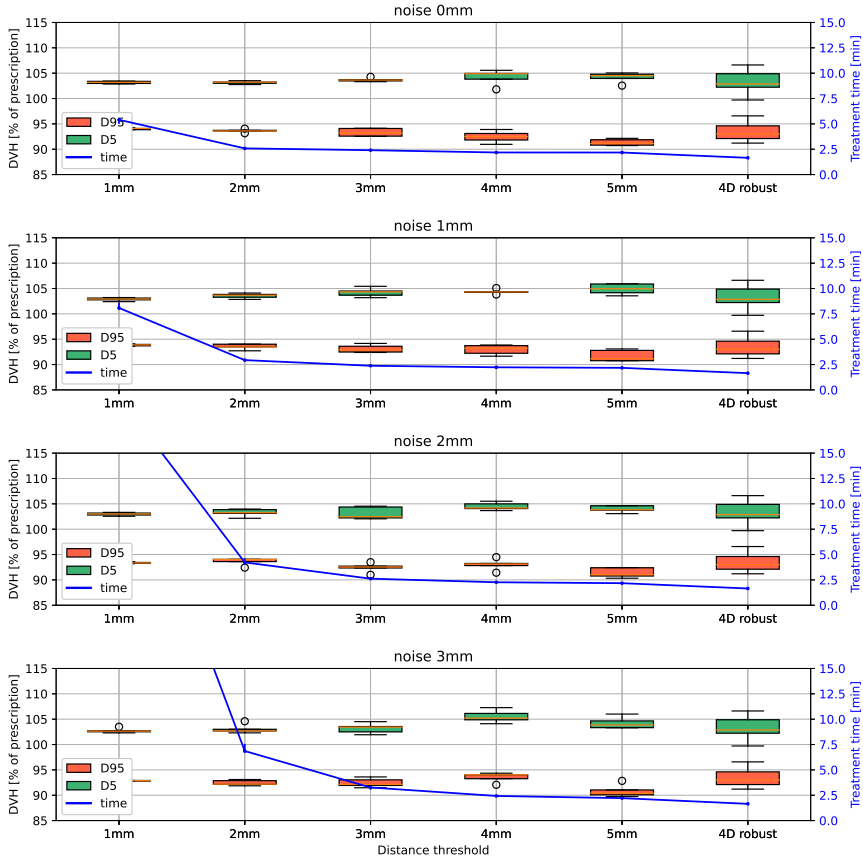


Fig. B.6 Patient 2 results for each noise (top to bottom) and distance thresholds (left to right).

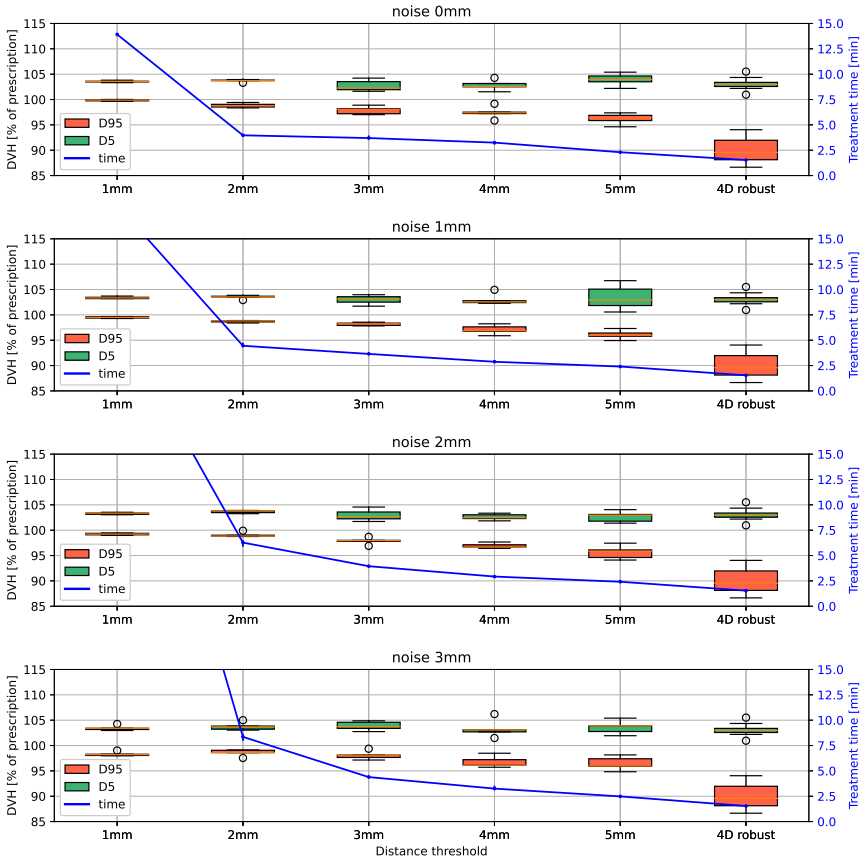


Fig. B.7 Patient 3 results for each noise (top to bottom) and distance thresholds (left to right).

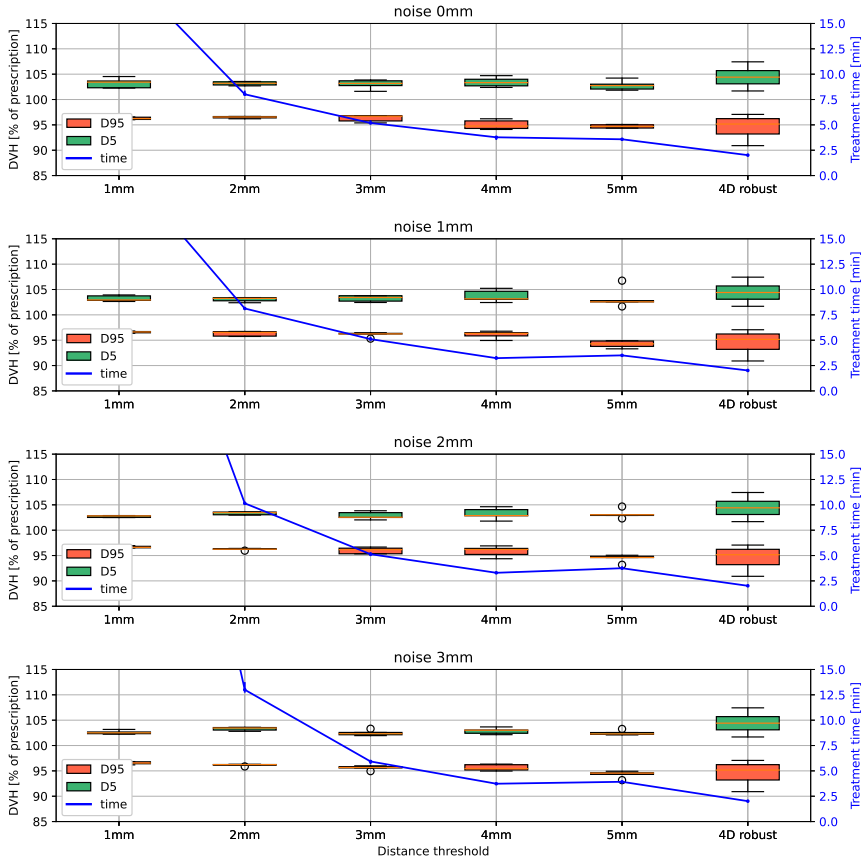
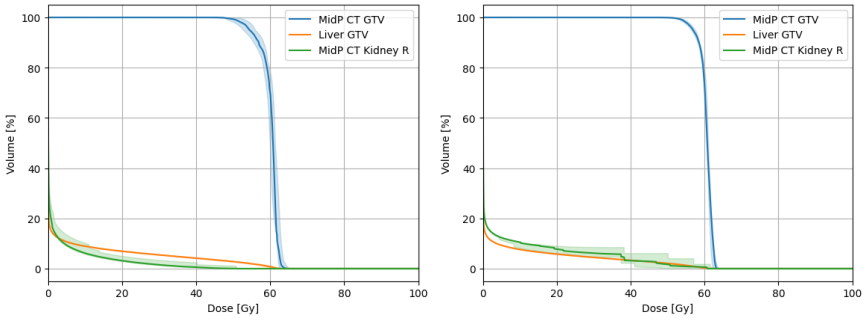


Fig. B.8 Patient 4 results for each noise (top to bottom) and distance thresholds (left to right).

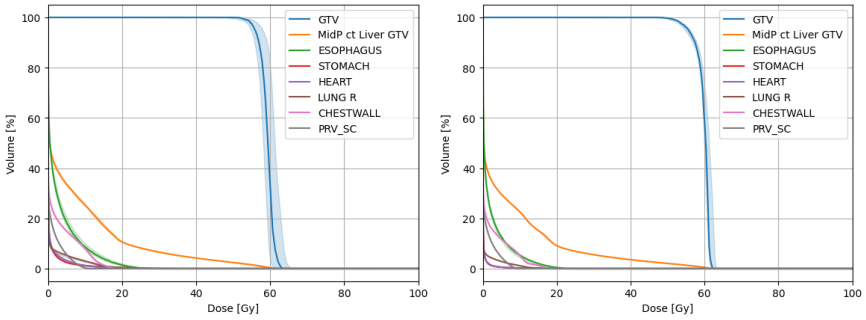
Cumulative DVH results

In this section, we provide the cumulative dose-volume histograms for the simulation of the 4D robust treatment and the library of treatment plans on the continuous sequence for the set of parameters chosen in Section 8.3.5, i.e. a noise amplitude of 2mm and a distance threshold of 3mm.



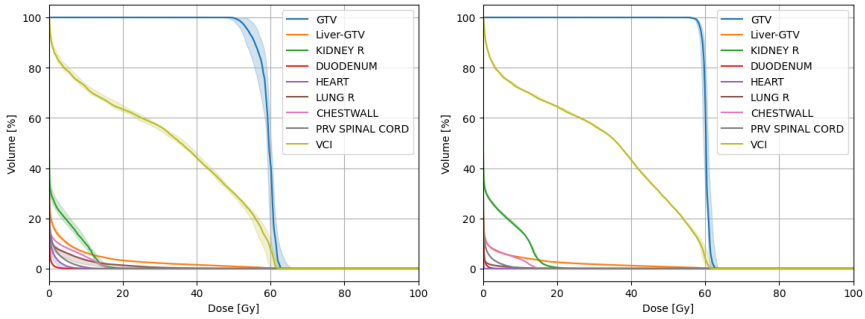
(a) 4D robust approach. **(b)** Library of plans.

Fig. B.9 Cumulative DVH results for Patient 1.



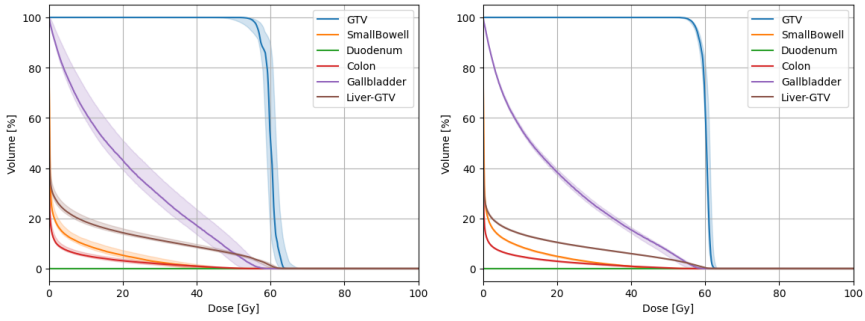
(a) 4D robust approach. **(b)** Library of plans.

Fig. B.10 Cumulative DVH results for Patient 2.



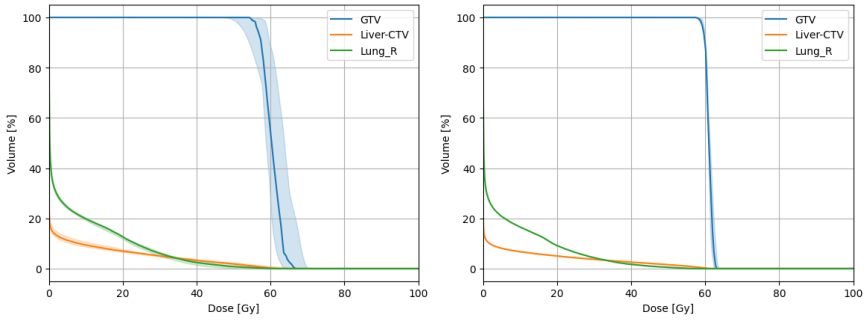
(a) 4D robust approach. **(b)** Library of plans.

Fig. B.11 Cumulative DVH results for Patient 3.



(a) 4D robust approach. **(b)** Library of plans.

Fig. B.12 Cumulative DVH results for Patient 4.



(a) 4D robust approach. **(b)** Library of plans.

Fig. B.13 Cumulative DVH results for Patient 5.

List of publications

The following articles are either published or submitted to peer-reviewed journals and conference proceedings:

- Valentin Hamaide and François Glineur. Predictive maintenance of a rotating condenser inside a synchrocyclotron. In *BNAIC/BENELEARN*, 2019.

- Valentin Hamaide and François Glineur. Transfer learning in bayesian optimization for the calibration of a beamline in proton therapy. In *ESANN 2021, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2021.

- Valentin Hamaide and François Glineur. Unsupervised minimum redundancy maximum relevance feature selection for predictive maintenance: Application to a rotating machine. *International Journal of Prognostics and Health Management*, 12(2), 2021.

- Valentin Hamaide, Denis Joassin, Lauriane Castin, and François Glineur. A two-level machine learning framework for predictive maintenance: comparison of learning formulations. Submitted to *Mechanical Systems and Signal Processing*.

- Valentin Hamaide, Kevin Souris, Damien Dasnoy, François Glineur, and Benoît Macq. Real-time image-guided treatment of mobile tumors in proton therapy by a library of treatment plans: a simulation study. Submitted to *Medical Physics*.

Published articles not related to this thesis:

- Ha Bui-Van, Valentin Hamaide, Christophe Craeye, François Glineur, and Eloy de Lera Acedo. Direct deterministic nulling techniques for large random arrays including mutual coupling. *IEEE Transactions on Antennas and Propagation*, 2018.
- Valentin Hamaide, Bertrand Hamaide, and Justin C. Williams. Nature reserve optimization with buffer zones and wildlife corridors for rare species. *Sustainability Analytics and Modeling*, 2022.

Bibliography

[AMC+20] Vepa Atamuradov, Kamal Medjaher, Fatih Camci, Noureddine Zerhouni, Pierre Dersin, and Benjamin Lamoureux. Machine health indicator construction framework for failure diagnostics and prognostics. *Journal of Signal Processing Systems*, 92(6):591–609, 2020.

[App] Ion Beam Applications. Proteus®one. Access: 2022-14-06. URL: `https://www.iba-worldwide.com/pt/proton-therapy/proton-therapy-solutions/proteus-one`.

[ARC+12] Asaad Abdollahzadeh, Alan Reynolds, Mike Christie, David Corne, Brian Davies, and Glyn Williams. Bayesian optimization algorithm applied to uncertainty quantification. *SPE Journal*, 17(03):865–873, 2012.

[ASH+17] Hidetaka Arimura, Yusuke Shibayama, Mohammad Haekal, Ze Jin, and Koujiro Ikushima. *Computer-Assisted Target Volume Determination*, pages 87–109. Springer, 2017. `doi:10.1007/978-981-10-2945-5_5`.

[B+13] Rémi Bardenet et al. Collaborative hyperparameter tuning. In *International conference on machine learning*, pages 199–207, 2013.

[BA03] Marc R Bussière and Judith A Adams. Treatment planning for conformal proton radiation therapy. *Technology in cancer research & treatment*, 2(5):389–399, 2003.

[BAR+13] Gauthier Bouilhol, Myriam Ayadi, Simon Rit, Sheeba Thengumpallil, Joël Schaerer, Jef Vandemeulebroucke, Line Claude, and David Sarrut. Is abdominal compression useful in lung stereotactic body radiation therapy? a 4dct and dosimetric lobe-dependent study. *Physica Medica*, 29(4):333–340, 2013.

[BBBK11] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, 24, 2011.

[BCDF10] Eric Brochu, Vlad M Cora, and Nando De Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*, 2010.

[BCVHH19] Sebastiaan Breedveld, David Craft, Rens Van Haveren, and Ben Heijmen. Multi-criteria optimization and decision-making in radiotherapy. *European Journal of Operational Research*, 277(1):1–19, 2019.

[BGR08] Christoph Bert, Sven O Grözinger, and Eike Rietzel. Quantification of interplay effects of scanned particle beams and moving targets. *Physics in Medicine & Biology*, 53(9):2253, 2008.

[BGV92] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM, 1992.

[BHKSC+16] Judit Boda-Heggemann, Antje-Christin Knopf, Anna Simeonova-Chergou, Hansjörg Wertz, Florian Stieler, Anika Jahnke, Lennart Jahnke, Jens Fleckenstein, Lena Vogel, Anna Arns, et al. Deep inspiration breath hold—based radiation therapy: a clinical review. *International Journal of Radiation Oncology* Biology* Physics*, 94(3):478–492, 2016.

[BKO+16] Michael Baumann, Mechthild Krause, Jens Overgaard, Jürgen Debus, Søren M Bentzen, Juliane Daartz, Christian Richter, Daniel Zips, and Thomas Bortfeld. Radiation oncology in the era of precision medicine. *Nature Reviews Cancer*, 16(4):234–249, 2016.

[BL17] Thomas R Bortfeld and Jay S Loeffler. Three ways to make proton therapy affordable. *Nature*, 549(7673):451–453, 2017.

[Bla22] Blackbaze hard drive data and stats. `https://www.backblaze.com/b2/hard-drive-test-data.html`, 2022. Accessed: 2022-06-08.

[BLYY12] Rajamanickam Baskar, Kuo Ann Lee, Richard Yeo, and Kheng-Wei Yeoh. Cancer and radiation therapy: current advances and future directions. *International journal of medical sciences*, 9(3):193, 2012.

[Bod18] Dayna Bodensteiner. Raystation: External beam treatment planning system. *Medical Dosimetry*, 43(2):168–176, 2018.

[can22] Treatment types. American Cancer Society, 2022. `https://www.cancer.org/treatment/treatments-and-side-effects/treatment-types.html` Accessed: May 2022.

[CEK+21] Katarzyna Czerska, Frank Emert, Renata Kopec, Katja Langen, Jamie R McClelland, Arturs Meijers, Naoki Miyamoto, Marco Riboldi, Shinichi Shimizu, Toshiyuki Terunuma, et al. Clinical practice vs. state-of-the-art research and future visions: Report on the 4d treatment planning workshop for particle therapy–edition 2018 and 2019. *Physica Medica*, 82:54–63, 2021.

[CGLRML20] Jair Cervantes, Farid Garcia-Lamont, Lisbeth Rodríguez-Mazahua, and Asdrubal Lopez. A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing*, 408:189–215, 2020.

[CH09] Jamie Coble and J Wesley Hines. Identifying optimal prognostic parameters from data: a genetic algorithms approach. In *Annual conference of the prognostics and health management society*, volume 27, 2009.

[CKK+19] Michael D Chuong, Adeel Kaiser, Fazal Khan, Parag Parikh, Edgar Ben-Josef, Christopher Crane, Thomas Brunner, Toshiyuki Okumura, Niek Schreuder, Søren M Bentzen, et al. Consensus report from the miami liver proton therapy conference. *Frontiers in Oncology*, 9:457, 2019.

[CL11] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–27, 2011.

[CMZN13] Fatih Camci, Kamal Medjaher, Noureddine Zerhouni, and Patrick Nectoux. Feature evaluation for effective bearing prognostics. *Quality and reliability engineering international*, 29(4):477–486, 2013.

[Cob10] Coble, J. B. Merging data sources to predict remaining useful life–an automated method to identify prognostic parameters, 2010. PhD thesis.

[CPV14] Emile Contal, Vianney Perchet, and Nicolas Vayatis. Gaussian process optimization with mutual information. In *International Conference on Machine Learning*, pages 253–261, 2014.

[CS14] Girish Chandrashekar and Ferat Sahin. A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1):16–28, 2014.

[CSV09] Andrew R Conn, Katya Scheinberg, and Luis N Vicente. *Introduction to derivative-free optimization*. SIAM, 2009.

[CZD+15] Jesus A Carino, Daniel Zurita, M Delgado, JA Ortega, and RJ Romero-Troncoso. Remaining useful life estimation of ball bearings by means of monotonic score calibration. In *2015 IEEE International Conference on Industrial Technology (ICIT)*, pages 1752–1758. IEEE, 2015.

[CZK+17] Joe Y Chang, Xiaodong Zhang, Antje Knopf, Heng Li, Shinichiro Mori, Lei Dong, Hsiao-Ming Lu, Wei Liu, Shahed N Badiyan, Stephen Both, et al. Consensus guidelines for implementing pencil-beam scanning proton therapy for thoracic malignancies on behalf of the ptcog thoracic and lymphoma subcommittee. *International Journal of Radiation Oncology* Biology* Physics*, 99(1):41–50, 2017.

[DBK+97] 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.

[DGSP13] S Dowdell, C Grassberger, GC Sharp, and H Paganetti. Interplay effects in proton scanning for lung: a 4d monte carlo study assessing the impact of tumor and beam delivery parameters. *Physics in Medicine & Biology*, 58(12):4137, 2013.

[Dic45] Lee R Dice. Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302, 1945.

[DP05] Chris Ding and Hanchuan Peng. Minimum redundancy feature selection from microarray gene expression data. *Journal of bioinformatics and computational biology*, 3(02):185–205, 2005.

[dPGH22] Félix de Patoul, François Glineur, and Valentin Hamaide. Predictive maintenance: predict upcoming failures via machine learning. Master thesis, UCLouvain, 2022.

[DRLJ+12] Dirk De Ruysscher, M Mark Lodge, Bleddyn Jones, Michael Brada, Alastair Munro, Thomas Jefferson, and Madelon Pijls-Johannesma. Charged particles in radiotherapy: a 5-year update of a systematic review. *Radiotherapy and Oncology*, 103(1):5–7, 2012.

[DSASM22] D Dasnoy-Sumell, A Aspeel, K Souris, and B Macq. Locally tuned deformation fields combination for 2d cine-mri-based driving of 3d motion models. *Physica Medica*, 94:8–16, 2022.

[DSSVOM20] Damien Dasnoy-Sumell, Kevin Souris, Geneviève Van Ooteghem, and Benoit Macq. Continuous real time 3d motion reproduction using dynamic mri and precomputed 4dct deformation fields. *Journal of Applied Clinical Medical Physics*, 21(8):236–248, 2020.

[DSUS16] Alberto Degiovanni, Pierpaolo Stabile, Donatella Ungaro, and ADAM SA. Light: a linear accelerator for proton therapy. *Proceedings of NAPAC2016, Chicago, USA*, 2016.

[DVB+18] Jennifer Dhont, Jef Vandemeulebroucke, Manuela Burghelea, Kenneth Poels, Tom Depuydt, Robbe Van Den Begin, Cyril Jaudet, Christine Collen, Benedikt Engels, Truus Reynders, et al. The long-and short-term variability of breathing induced tumor motion in lung and liver over the course of a radiotherapy treatment. *Radiotherapy and Oncology*, 126(2):339–346, 2018.

[DYS+20] Wei Deng, James E Younkin, Kevin Souris, Sheng Huang, Kurt Augustine, Mirek Fatyga, Xiaoning Ding, Marie Cohilis, Martin Bues, Jie Shan, et al. Integrating an open source monte carlo code "mcsquare" for clinical use in intensity-modulated proton therapy. *Medical physics*, 47(6):2558–2574, 2020.

[EDSS13] F Ernst, R Dürichen, A Schlaefer, and A Schweikard. Evaluating and comparing algorithms for respiratory motion prediction. *Physics in Medicine & Biology*, 58(11):3911, 2013.

[EFG18] Erik Engwall, Albin Fredriksson, and Lars Glimelius. 4d robust optimization including uncertainties in time structures can reduce the interplay effect in proton pencil beam scanning radiation therapy. *Medical physics*, 45(9):4020–4029, 2018.

[EHPF11] Merran Evans, Nicholas Hastings, Brian Peacock, and Catherine Forbes. *Statistical distributions*. John Wiley & Sons, 2011.

[EPS+11] Cynthia L Eccles, Ritesh Patel, Anna K Simeonov, Gina Lockwood, Masoom Haider, and Laura A Dawson. Com-

parison of liver tumor motion with and without abdominal compression using cine-magnetic resonance imaging. *International Journal of Radiation Oncology* Biology* Physics*, 79(2):602–608, 2011.

[FCBC+19] Marta Fernandes, Alda Canito, Verónica Bolón-Canedo, Luís Conceição, Isabel Praça, and Goreti Marreiros. Data analysis and feature selection for predictive maintenance: A case-study in the metallurgic industry. *International journal of information management*, 46:252–262, 2019.

[FEL+20] J Ferlay, M Ervik, F Lam, M Colombet, L Mery, and M et al. Piñeros. Global cancer observatory: Cancer today. `https://gco.iarc.fr/today`, 2020. Accessed: May 2022.

[FFMRFRAB13] Diego Fernandez-Francos, David Martinez-Rego, Oscar Fontenla-Romero, and Amparo Alonso-Betanzos. Automatic bearing fault diagnosis based on one-class ν-svm. *Computers & Industrial Engineering*, 64(1):357–365, 2013.

[FIS+10] Takuji Furukawa, Taku Inaniwa, Shinji Sato, Toshiyuki Shirai, Shinichiro Mori, Eri Takeshita, Kota Mizushima, Takeshi Himukai, and Koji Noda. Moving target irradiation with fast rescanning and gating in particle therapy. *Medical physics*, 37(9):4874–4879, 2010.

[Fra18] Peter I Frazier. A tutorial on bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018.

[FSC+17] Giovanni Fattori, Sairos Safai, Pablo Fernández Carmona, Marta Peroni, Rosalind Perrin, Damien Charles Weber, and Antony John Lomax. Monitoring of breathing motion in image-guided pbs proton therapy: comparative analysis of optical and electromagnetic technologies. *Radiation oncology*, 12(1):1–11, 2017.

[FVdMB+15] Davide Fontanarosa, Skadi Van der Meer, Jeffrey Bamber, Emma Harris, Tuathan O'Shea, and Frank Verhaegen. Review of ultrasound image guidance in external beam radiotherapy: I. treatment planning and inter-fraction mo-

tion management. *Physics in medicine & biology*, 60(3):R77, 2015.

[G+09] Toni Giorgino et al. Computing and visualizing dynamic time warping alignments in r: the dtw package. *Journal of statistical Software*, 31(7):1–24, 2009.

[Gar15] Roman Garnett. Lecture notes on bayesian methods in machine learning, Spring 2015.

[GCD15a] Zhiwei Gao, Carlo Cecati, and Steven X Ding. A survey of fault diagnosis and fault-tolerant techniques—part 1 : Fault diagnosis with model-based and signal-based approaches. *IEEE transactions on industrial electronics*, 62(6):3757–3767, 2015.

[GCD15b] Zhiwei Gao, Carlo Cecati, and Steven X Ding. A survey of fault diagnosis and fault-tolerant techniques—part 2 : Fault diagnosis with knowledge-based and hybrid/active approaches. *IEEE Transactions on Industrial Electronics*, 62(6):3768–3774, 2015.

[Gra14] Christian Graeff. Motion mitigation in scanned ion beam therapy through 4d-optimization. *Physica Medica*, 30(5):570–577, 2014.

[GRG+20] Stewart Greenhill, Santu Rana, Sunil Gupta, Pratibha Vellanki, and Svetha Venkatesh. Bayesian optimization for adaptive experimental design: A review. *IEEE access*, 8:13937–13948, 2020.

[GRL+06] Sven Oliver Grözinger, Eike Rietzel, Qiang Li, Christoph Bert, Thomas Haberer, and Gerhard Kraft. Simulations to design an online motion compensation system for scanned particle beams. *Physics in Medicine & Biology*, 51(14):3517, 2006.

[GY59] IM Gel'Fand and AM Yaglom. Calculation of the amount of information abouta random function contained in another such function. *Eleven Papers on Analysis, Probability and Topology*, 12:199, 1959.

[HA06] Kenneth R Hogstrom and Peter R Almond. Review of electron beam therapy physics. *Physics in Medicine & Biology*, 51(13):R455, 2006.

[Han18] Jay Hancock. For cancer centers, proton therapy's promise is undercut by lagging demand. 2018.

[HDM+99] Joseph Hanley, Marc M Debois, Dennis Mah, Gikas S Mageras, Adam Raben, Kenneth Rosenzweig, Borys Mychalczak, Lawrence H Schwartz, Paul J Gloeggler, Wendell Lutz, et al. Deep inspiration breath-hold technique for lung tumors: the potential value of target immobilization and reduced lung density in dose escalation. *International Journal of Radiation Oncology* Biology* Physics*, 45(3):603–611, 1999.

[HDO+98] Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4):18–28, 1998.

[HG19] Valentin Hamaide and François Glineur. Predictive maintenance of a rotating condenser inside a synchrocyclotron. In *BNAIC/BENELEARN*, 2019.

[HG21a] Valentin Hamaide and François Glineur. Transfer learning in bayesian optimization for the calibration of a beam line in proton therapy. In *ESANN 2021, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2021.

[HG21b] Valentin Hamaide and François Glineur. Unsupervised minimum redundancy maximum relevance feature selection for predictive maintenance: Application to a rotating machine. *International Journal of Prognostics and Health Management*, 12(2), 2021.

[HJCG22] Valentin Hamaide, Denis Joassin, Lauriane Castin, and François Glineur. A two-level machine learning framework for predictive maintenance: comparison of learning formulations. submitted to *Mechanical Systems and Signal Processing*, 2022.

[HMZ14] Turid Hellevik and Iñigo Martinez-Zubiaurre. Radiotherapy and the tumor stroma: the importance of dose and fractionation. *Frontiers in oncology*, 4:1, 2014.

[HSD+22] Valentin Hamaide, Kevin Souris, Damien Dasnoy, François Glineur, and Benoît Macq. Real-time image-guided treatment of mobile tumors in proton therapy by a library of treatment plans: a simulation study. submitted to *Medical Physics*, 2022.

[HSQ+20] Qin Hu, Xiao-Sheng Si, Ai-Song Qin, Yun-Rong Lv, and Qing-Hua Zhang. Machinery fault diagnosis scheme using redefined dimensionless indicators and mrmr feature selection. *IEEE Access*, 8:40313–40326, 2020.

[HSTM19] Ryusuke Hirai, Yukinobu Sakata, Akiyuki Tanizawa, and Shinichiro Mori. Real-time tumor tracking using fluoroscopic imaging with deep neural network analysis. *Physica Medica*, 59:22–29, 2019.

[J+19] Tinu Theckel Joy et al. A flexible transfer learning framework for bayesian optimization with convergence guarantee. *Expert Systems with Applications*, 115:656–672, 2019.

[Jäk09] Oliver Jäkel. Medical physics aspects of particle therapy. *Radiation protection dosimetry*, 137(1-2):156–166, 2009.

[JEG+20] Alexander Jöhl, Stefanie Ehrbar, Matthias Guckenberger, Stephan Klöck, Mirko Meboldt, Melanie Zeilinger, Stephanie Tanadini-Lang, and Marianne Schmid Daners. Performance comparison of prediction filters for respiratory motion tracking in radiotherapy. *Medical physics*, 47(2):643–650, 2020.

[Jer15] Martin Jermann. Particle therapy statistics in 2014. *International Journal of Particle Therapy*, 2(1):50–54, 2015.

[JGZN13] Kamran Javed, Rafael Gouriveau, Noureddine Zerhouni, and Patrick Nectoux. A feature extraction procedure based on trigonometric functions and cumulative descriptors to enhance prognostics modeling. In *2013 IEEE Conference on Prognostics and Health Management (PHM)*, pages 1–7. IEEE, 2013.

[JGZN14] Kamran Javed, Rafael Gouriveau, Noureddine Zerhouni, and Patrick Nectoux. Enabling health monitoring approach based on vibration data for accurate prognostics. *IEEE Transactions on Industrial Electronics*, 62(1):647–656, 2014.

[JLB06] Andrew KS Jardine, Daming Lin, and Dragan Banjevic. A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical systems and signal processing*, 20(7):1483–1510, 2006.

[JSQ+16] Xiaohang Jin, Yi Sun, Zijun Que, Yu Wang, and Tommy WS Chow. Anomaly detection and fault prognosis for bearings. *IEEE Transactions on Instrumentation and Measurement*, 65(9):2046–2054, 2016.

[KAF+13] WJGM Kleeven, M Abs, E Forton, S Henrotin, Y Jongen, V Nuttens, Y Paradis, E Pearson, S Quets, J Van de Walle, et al. The IBA superconducting synchrocyclotron project S2C2. In *Proc. Cyclotrons*, pages 115–119, 2013.

[KCMM+16] Racha Khelif, Brigitte Chebel-Morello, Simon Malinowski, Emna Laajili, Farhat Fnaiech, and Noureddine Zerhouni. Direct remaining useful life estimation based on support vector regression. *IEEE Transactions on industrial electronics*, 64(3):2276–2285, 2016.

[Kor15] SS Korreman. Image-guided radiotherapy and motion management in lung cancer. *The British journal of radiology*, 88(1051):20150100, 2015.

[Kra21] Martin Krasser. Bayesian machine learning. `https://github.com/krasserm/bayesian-machine-learning`, 2021. GitHub repository.

[Kru18] Jon Kruse. Proton image guidance. In *Proton Therapy Physics*, pages 615–641. CRC Press, 2018.

[KSB+21] Paul J Keall, Amit Sawant, Ross I Berbeco, Jeremy T Booth, Byungchul Cho, Laura I Cerviño, Eileen Cirino, Sonja Dieterich, Martin F Fast, Peter B Greer, et al. Aapm task group 264: The safe clinical implementation of mlc

tracking in radiotherapy. *Medical physics*, 48(5):e44–e64, 2021.

[KSG04] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical review E*, 69(6):066138, 2004.

[LBS+07] Hsiao-Ming Lu, Robert Brett, Gregory Sharp, Soiros Safai, Steve Jiang, Jay Flanz, and Hanne Kooy. A respiratory-gated treatment system for proton therapy. *Medical physics*, 34(8):3273–3278, 2007.

[LDN+20] Michelle Lis, Marco Donetti, Wayne Newhauser, Marco Durante, Joyoni Dey, Ulrich Weber, Moritz Wolf, Timo Steinsberger, and Christian Graeff. A modular dose delivery system for treating moving targets with scanned ion beams: Performance and safety characteristics, and preliminary tests. *Physica Medica*, 76:307–316, 2020.

[LDS18] Xiang Li, Qian Ding, and Jian-Qiao Sun. Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliability Engineering & System Safety*, 172:1–11, 2018.

[LFL+13] Wei Liu, Steven J Frank, Xiaoqiang Li, Yupeng Li, Peter C Park, Lei Dong, X Ronald Zhu, and Radhe Mohan. Effectiveness of robust optimization in intensity-modulated proton therapy planning for head and neck cancers. *Medical physics*, 40(5):051711, 2013.

[LLG+16] Yaguo Lei, Naipeng Li, Szymon Gontarz, Jing Lin, Stanislaw Radkowski, and Jacek Dybala. A model-based method for remaining useful life prediction of machinery. *IEEE Transactions on Reliability*, 65(3):1314–1326, 2016.

[LLG+18] Yaguo Lei, Naipeng Li, Liang Guo, Ningbo Li, Tao Yan, and Jing Lin. Machinery health prognostics: A systematic review from data acquisition to rul prediction. *Mechanical Systems and Signal Processing*, 104:799–834, 2018.

[LLLL14] Naipeng Li, Yaguo Lei, Zongyao Liu, and Jing Lin. A particle filtering-based approach for remaining useful life

predication of rolling element bearings. In *2014 International Conference on Prognostics and Health Management*, pages 1–8. IEEE, 2014.

[LMP+14] Wei Liu, Radhe Mohan, Peter Park, Zhong Liu, Heng Li, Xiaoqiang Li, Yupeng Li, Richard Wu, Narayan Sahoo, Lei Dong, et al. Dosimetric benefits of robust treatment planning for intensity modulated proton therapy for base-of-skull cancers. *Practical radiation oncology*, 4(6):384–391, 2014.

[LMW19] Jeffrey Larson, Matt Menickelly, and Stefan M Wild. Derivative-free optimization methods. *Acta Numerica*, 28:287–404, 2019.

[LN89] Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.

[LQY+07] J Lee, H Qiu, G Yu, J Lin, et al. Rexnord technical services: Bearing data set. *Moffett Field, CA: IMS, Univ. Cincinnati. NASA Ames Prognostics Data Repository, NASA Ames*, 2007.

[LSC+16] Wei Liu, Steven E Schild, Joe Y Chang, Zhongxing Liao, Yu-Hui Chang, Zhifei Wen, Jiajian Shen, Joshua B Stoker, Xiaoning Ding, Yanle Hu, et al. Exploratory study of 4d versus 3d robust optimization in intensity modulated proton therapy for lung cancer. *International Journal of Radiation Oncology* Biology* Physics*, 95(1):523–533, 2016.

[LSJ21] John Lee, Edmond Sterpin, and Guillaume Janssens. Engineering challenges in proton therapy. Université Catholique de Louvain, 2021.

[LWB+07] Daniel J Lizotte, Tao Wang, Michael H Bowling, Dale Schuurmans, et al. Automatic gait optimization with gaussian process regression. In *IJCAI*, volume 7, pages 944–949, 2007.

[LYJ+20] Yaguo Lei, Bin Yang, Xinwei Jiang, Feng Jia, Naipeng Li, and Asoke K Nandi. Applications of machine learning to

machine fault diagnosis: A review and roadmap. *Mechanical Systems and Signal Processing*, 138:106587, 2020.

[LYL+17] Yongbo Li, Yuantao Yang, Guoyan Li, Minqiang Xu, and Wenhu Huang. A fault diagnosis scheme for planetary gearboxes using modified multi-scale symbolic dynamic entropy and mrmr feature selection. *Mechanical Systems and Signal Processing*, 91:295–312, 2017.

[LZL+19] Hui Lin, Wei Zou, Taoran Li, Steven J Feigenberg, Boon-Keng K Teo, and Lei Dong. A super-learner model for tumor motion prediction and management in radiation therapy: development and feasibility evaluation. *Scientific reports*, 9(1):1–11, 2019.

[LZLL17] Guoliang Lu, Yiqi Zhou, Changhou Lu, and Xueyong Li. A novel framework of change-point detection for machine monitoring. *Mechanical Systems and Signal Processing*, 83:533–548, 2017.

[LZP+15] Heng Li, Xiaodong Zhang, Peter Park, Wei Liu, Joe Chang, Zhongxing Liao, Steve Frank, Yupeng Li, Falk Poenisch, Radhe Mohan, et al. Robust optimization in intensity-modulated proton therapy to account for anatomy changes in lung cancer patients. *Radiotherapy and Oncology*, 114(3):367–372, 2015.

[LZQ16] Zhiliang Liu, Ming J Zuo, and Yong Qin. Remaining useful life prediction of rolling element bearings based on health state assessment. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 230(2):314–330, 2016.

[LZX13] Zhiliang Liu, Ming J Zuo, and Hongbing Xu. Fault diagnosis for planetary gearboxes using multi-criterion fusion feature selection framework. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 227(9):2064–2076, 2013.

[LZZ15] Heng Li, X Ronald Zhu, and Xiaodong Zhang. Reducing dose uncertainty for spot-scanning proton beam therapy of moving tumors by optimizing the spot delivery

sequence. *International Journal of Radiation Oncology* Biology* Physics*, 93(3):547–556, 2015.

[MAB+17] Mark V Mishra, Sameer Aggarwal, Soren M Bentzen, Nancy Knight, Minesh P Mehta, and William F Regine. Establishing evidence-based indications for proton therapy: an overview of current clinical trials. *International Journal of Radiation Oncology* Biology* Physics*, 97(2):228–235, 2017.

[Mat21] MathWorks. Signal features - matlab & simulink, 2021. URL: `https://mathworks.com/help/predmaint/ug/signal-features.html`.

[MG17] Radhe Mohan and David Grosshans. Proton therapy–present and future. *Advanced drug delivery reviews*, 109:26–44, 2017.

[MGH20] Fabio Mercurio, François Glineur, and Valentin Hamaide. Neural networks and gradient boosting for predictive maintenance of a proton therapy machine. Master thesis, UCLouvain, 2020.

[MKU18] Shinichiro Mori, Antje-Christin Knopf, and Kikuo Umegaki. Motion management in particle therapy. *Medical Physics*, 45(11):e994–e1010, 2018.

[MMA+19] Naoki Miyamoto, Kenichiro Maeda, Daisuke Abo, Ryo Morita, Seishin Takao, Taeko Matsuura, Norio Katoh, Kikuo Umegaki, Shinichi Shimizu, and Hiroki Shirato. Quantitative evaluation of image recognition performance of fiducial markers in real-time tumor-tracking radiation therapy. *Physica Medica*, 65:33–39, 2019.

[MPB+00] B Marchand, D Prieels, B Bauvir, R Sépulchre, and M Gérard. Iba proton pencil beam scanning: an innovative solution for cancer treatment. In *Proceedings of EPAC*, pages 2539–2541, 2000.

[MS09] Sankar Mahadevan and Sirish L Shah. Fault detection and diagnosis in process data using one-class support vector machines. *Journal of process control*, 19(10):1627–1639, 2009.

[MSH+19] Shinichiro Mori, Yukinobu Sakata, Ryusuke Hirai, Wataru Furuichi, Kazuki Shimabukuro, Ryosuke Kohno, Woong Sub Koom, Shigeru Kasai, Keiko Okaya, and Yasushi Iseki. Commissioning of a fluoroscopic-based real-time markerless tumor tracking system in a superconducting rotating gantry for carbon-ion pencil beam scanning treatment. *Medical physics*, 46(4):1561–1574, 2019.

[Mur12] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[MZ14] Timur Mitin and Anthony L Zietman. Promise and pitfalls of heavy-particle therapy. *Journal of Clinical Oncology*, 32(26):2855, 2014.

[NGM+12] Patrick Nectoux, Rafael Gouriveau, Kamal Medjaher, Emmanuel Ramasso, Brigitte Chebel-Morello, Noureddine Zerhouni, and Christophe Varnier. Pronostia: An experimental platform for bearings accelerated degradation tests. In *IEEE International Conference on Prognostics and Health Management, PHM'12.*, pages 1–8. IEEE Catalog Number: CPF12PHM-CDR, 2012.

[NM65] John A Nelder and Roger Mead. A simplex method for function minimization. *The computer journal*, 7(4):308–313, 1965.

[OBF+16] Tuathan O'Shea, Jeffrey Bamber, Davide Fontanarosa, Skadi van der Meer, Frank Verhaegen, and Emma Harris. Review of ultrasound image guidance in external beam radiotherapy part ii: intra-fraction motion management and novel applications. *Physics in Medicine & Biology*, 61(8):R90, 2016.

[Oh19] Dongryul Oh. Proton therapy: the current status of the clinical evidences. *Precision and Future Medicine*, 3(3):91–102, 2019.

[OLJ16] Hywel Owen, Antony Lomax, and Simon Jolly. Current and future accelerator technologies for charged particle

therapy. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 809:96–104, 2016.

[Ott08] Karl Otto. Volumetric modulated arc therapy: Imrt in a single gantry arc. *Medical physics*, 35(1):310–317, 2008.

[Pag18] Harald Paganetti. *Proton therapy physics*. CRC press, 2018.

[PAM+12] Harald Paganetti, Basit S Athar, Maryam Moteabbed, Judith A Adams, Uwe Schneider, and Torunn I Yock. Assessment of radiation-induced second cancer risks in proton therapy and imrt for organs inside the primary radiation field. *Physics in Medicine & Biology*, 57(19):6047, 2012.

[PJ07] Åsa Palm and Karl-Axel Johansson. A review of the impact of photon and proton external beam radiotherapy treatment modalities on the dose distribution in field and out-of-field; implications for the long-term morbidity of cancer survivors. *Acta oncologica*, 46(4):462–473, 2007.

[PKB+19] T Pfeiler, D Ahmad Khalil, C Bäumer, O Blanck, M Chan, E Engwall, D Geismar, S Peters, S Plaude, B Spaan, et al. 4d robust optimization in pencil beam scanning proton therapy for hepatocellular carcinoma. In *Journal of Physics: Conference Series*, volume 1154, page 012021. IOP Publishing, 2019.

[PKS+14] Matthias Prall, R Kaderka, N Saito, C Graeff, C Bert, M Durante, K Parodi, J Schwaab, C Sarti, and J Jenne. Ion beam tracking using ultrasound motion detection. *Medical physics*, 41(4):041708, 2014.

[PLD05] Hanchuan Peng, Fuhui Long, and Chris Ding. Feature selection based on mutual information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, 2005. doi:10.1109/TPAMI.2005.159.

[RGFO17] Milos Radovic, Mohamed Ghalwash, Nenad Filipovic, and Zoran Obradovic. Minimum redundancy maximum

relevance feature selection approach for temporal gene expression data. *BMC bioinformatics*, 18(1):1–14, 2017.

[Ros14] Brian C Ross. Mutual information between discrete and continuous data sets. *PloS one*, 9(2):e87357, 2014.

[RS13] Luis Miguel Rios and Nikolaos V Sahinidis. Derivative-free optimization: a review of algorithms and comparison of software implementations. *Journal of Global Optimization*, 56(3):1247–1293, 2013.

[RS20] Behnoush Rezaeianjouybari and Yi Shang. Deep learning for prognostics and health management: State of the art, challenges, and opportunities. *Measurement*, 163:107929, 2020.

[RSVG16] Kishore K Reddy, Soumalya Sarkar, Vivek Venugopalan, and Michael Giering. Anomaly detection and fault disambiguation in large flight data: a multi-modal deep auto-encoder approach. In *Annual Conference of the PHM Society*, volume 8, 2016.

[RT18] Cédric Hernalsteens Robin Tesse, Kévin André. Optimization of hadron therapy beamlines using a novel fast tracking code for beam transport and beam-matter interactions, 2018. URL: `https://accelconf.web.cern.ch/icap2018/talks/supaf03_talk.pdf`.

[SFCOMT20] Saúl Solorio-Fernández, J Ariel Carrasco-Ochoa, and José Fco Martínez-Trinidad. A review of unsupervised feature selection methods. *Artificial Intelligence Review*, 53(2):907–948, 2020.

[SGK+16] Kevin Souris, Adam Glick, Minglei Kang, Guillaume Janssens, Edmond Sterpin, Habo Lin, James McDonough, Charles Simone, Timothy Solberg, Edgar Ben-Josef, et al. Su-f-t-121: Abdominal compression effectively reduces the interplay effect and enables pencil beam scanning proton therapy of liver tumors. *Medical Physics*, 43(6Part14):3489–3489, 2016.

[SGSE08] Abhinav Saxena, Kai Goebel, Don Simon, and Neil Eklund. Damage propagation modeling for aircraft engine run-to-failure simulation. In *2008 international conference on prognostics and health management*, pages 1–9. IEEE, 2008.

[SHKB95] Randy R Schoen, Thomas G Habetler, Farrukh Kamran, and RG Bartfield. Motor bearing damage detection using stator current monitoring. *IEEE transactions on industry applications*, 31(6):1274–1279, 1995.

[SLA12] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25:2951–2959, 2012.

[SLS16] Kevin Souris, John Aldo Lee, and Edmond Sterpin. Fast multipurpose monte carlo simulation for proton therapy using multi-and many-core cpu architectures. *Medical physics*, 43(4):1700–1712, 2016.

[SMH16] Parham Shahidi, Daniel Maraini, and Brad Hopkins. Railcar diagnostics using minimal-redundancy maximumrelevance feature selection and support vector machine classification. *International Journal of Prognostics and Health Management*, 7:2153–2648, 2016.

[SMM+14] Shinichi Shimizu, Naoki Miyamoto, Taeko Matsuura, Yusuke Fujii, Masumi Umezawa, Kikuo Umegaki, Kazuo Hiramoto, and Hiroki Shirato. A proton beam therapy system dedicated to spot-scanning increases accuracy with moving tumors by real-time imaging and gating and reduces equipment size. *PloS one*, 9(4):e94971, 2014.

[SMZ14] Abdenour Soualhi, Kamal Medjaher, and Noureddine Zerhouni. Bearing health monitoring based on hilbert–huang transform, support vector machine, and regression. *IEEE Transactions on Instrumentation and Measurement*, 64(1):52–62, 2014.

[SN09] S. Singer and J. Nelder. Nelder-Mead algorithm. *Scholarpedia*, 4(7):2928, 2009. revision #91557. doi:10.4249/scholarpedia.2928.

[SPST+01] Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471, 2001.

[SRTP09] Joao Seco, Daniel Robertson, Alexei Trofimov, and Harald Paganetti. Breathing interplay effects during proton beam scanning: simulation and statistical analysis. *Physics in Medicine & Biology*, 54(14):N283, 2009.

[SS04] Alex J Smola and Bernhard Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 14(3):199–222, 2004.

[SSP+14] Gian Antonio Susto, Andrea Schirru, Simone Pampuri, Seán McLoone, and Alessandro Beghi. Machine learning for predictive maintenance: A multiple classifier approach. *IEEE Transactions on Industrial Informatics*, 11(3):812–820, 2014.

[SSW+15] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2015.

[Ste99] Michael L Stein. *Interpolation of spatial data: some theory for kriging*. Springer, 1999.

[SWHZ11] Xiao-Sheng Si, Wenbin Wang, Chang-Hua Hu, and Dong-Hua Zhou. Remaining useful life estimation–a review on the statistical data driven approaches. *European journal of operational research*, 213(1):1–14, 2011.

[THG+20] Xianghong Tang, Qiang He, Xin Gu, Chuanjiang Li, Huan Zhang, and Jianguang Lu. A novel bearing fault diagnosis method based on gl-mrmr-svm. *Processes*, 8(7):784, 2020.

[TMMZT11] Diego A Tobon-Mejia, Kamal Medjaher, Noureddine Zerhouni, and Gérard Tripot. Hidden markov models for failure diagnostic and prognostic. In *2011 Prognostics and System Health Managment Confernece*, pages 1–8. IEEE, 2011.

[vDStH+16] Lisanne V van Dijk, Roel JHM Steenbakkers, Bennie ten Haken, Hans Paul van der Laan, Aart A van 't Veld, Johannes A Langendijk, and Erik W Korevaar. Robust intensity modulated proton therapy (impt) increases estimated clinical benefit in head and neck cancer patients. *PloS one*, 11(3):e0152477, 2016.

[VdWKZ+09] S Van de Water, R Kreuger, S Zenklusen, E Hug, and Antony J Lomax. Tumour tracking with scanned proton beams: assessing the accuracy and practicalities. *Physics in Medicine & Biology*, 54(21):6549, 2009.

[VMM16] Vivek Verma, Mark V Mishra, and Minesh P Mehta. A systematic review of the cost and cost-effectiveness studies of proton radiotherapy. *Cancer*, 122(10):1483–1501, 2016.

[VODSL+19] Geneviève Van Ooteghem, Damien Dasnoy-Sumell, Maarten Lambrecht, Grégory Reychler, Giuseppe Liistro, Edmond Sterpin, and Xavier Geets. Mechanically-assisted non-invasive ventilation: a step forward to modulate and to improve the reproducibility of breathing-related motion in radiation therapy. *Radiotherapy and Oncology*, 133:132–139, 2019.

[VPPA09] Tom Vercauteren, Xavier Pennec, Aymeric Perchant, and Nicholas Ayache. Diffeomorphic demons: Efficient non-parametric image registration. *NeuroImage*, 45(1):S61–S72, 2009.

[VSV+18] Sai Vemprala, Srikanth Saripalli, Carlos Vargas, Martin Bues, Yanle Hu, and Jiajian Shen. Real-time tumor tracking for pencil beam scanning proton therapy. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4434–4440. IEEE, 2018.

[VWL+10] Poonam Verma, Huanmei Wu, Mark Langer, Indra Das, and George Sandison. Survey: real-time tumor motion prediction for image-guided radiation treatment. *Computing in Science & Engineering*, 13(5):24–35, 2010.

[Wan02] Wenbin Wang. A model to predict the residual life of rolling element bearings given monitored condition information to date. *IMA Journal of management mathematics*, 13(1):3–16, 2002.

[WHO22] Cancer. World Health Organization, 2022. `https://www.who.int/news-room/fact-sheets/detail/cancer` Accessed: May 2022.

[WKW16] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3(1):1–40, 2016.

[WPZ+16] Yu Wang, Yizhen Peng, Yanyang Zi, Xiaohang Jin, and Kwok-Leung Tsui. A two-stage data-driven-based prognostic approach for bearing degradation problem. *IEEE Transactions on industrial informatics*, 12(3):924–932, 2016.

[WR06] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

[WSVHD08] Jochem WH Wolthaus, J-J Sonke, M Van Herk, and EMF Damen. Reconstruction of a time-averaged midposition ct scan for radiotherapy planning of lung cancer patients using deformable registration a. *Medical physics*, 35(9):3998–4011, 2008.

[WTM17] Dong Wang, Kwok-Leung Tsui, and Qiang Miao. Prognostics and health management: A review of vibration based bearing and gear health indicators. *Ieee Access*, 6:665–676, 2017.

[WVDSL+16] Joachim Widder, Arjen Van Der Schaaf, Philippe Lambin, Corrie AM Marijnen, Jean-Philippe Pignol, Coen R Rasch, Ben J Slotman, Marcel Verheij, and Johannes A

Langendijk. The quest for evidence for proton therapy: model-based approach and precision medicine. *International Journal of Radiation Oncology* Biology* Physics*, 95(1):30–36, 2016.

[WYD+18] Yuting Wu, Mei Yuan, Shaopeng Dong, Li Lin, and Yingqi Liu. Remaining useful life estimation of engineered systems using vanilla lstm neural networks. *Neurocomputing*, 275:167–179, 2018.

[YJ19] Xiaoan Yan and Minping Jia. Intelligent fault diagnosis of rotating machinery using improved multiscale dispersion entropy and mrmr feature selection. *Knowledge-Based Systems*, 163:450–471, 2019.

[YM14] Dani Yogatama and Gideon Mann. Efficient transfer learning method for automatic hyperparameter tuning. In *Artificial intelligence and statistics*, pages 1077–1085, 2014.

[YZQ19] Tai-Ze Yuan, Ze-Jiang Zhan, and Chao-Nan Qian. New frontiers in proton therapy: applications in cancers. *Cancer Communications*, 39(1):1–7, 2019.

[ZDMP94] Alex P Zijdenbos, Benoit M Dawant, Richard A Margolin, and Andrew C Palmer. Morphometric analysis of white matter lesions in mr images: method and validation. *IEEE transactions on medical imaging*, 13(4):716–724, 1994.

[Zen10] Silvan Marius Zenklusen. *Exploring the potential of advanced pencil beam scanning for treating moving targets with the new Gantry 2 at the Paul Scherrer Institut*. PhD thesis, ETH Zurich, 2010.

[ZNSM14] Junda Zhu, Tom Nostrand, Cody Spiegel, and Brogan Morton. Survey of condition indicators for condition monitoring systems. In *Annual Conference of the PHM Society*, volume 6, 2014.

[ZP16] Jiahua Zhu and Scott N Penfold. Dosimetric comparison of stopping power calibration with dual-energy ct and single-energy ct in proton therapy treatment planning. *Medical physics*, 43(6Part1):2845–2854, 2016.

[ZSL+18] Xiaoguang Zhang, Zhenyue Song, Dandan Li, Wei Zhang, Zhike Zhao, and Yingying Chen. Fault diagnosis for reducer via improved lmd and svm-rfe-mrmr. *Shock and Vibration*, 2018, 2018.

[ZWZ+22] Lu Zeng, Xin Wang, Jidan Zhou, Pan Gong, Xuetao Wang, Xiaohong Wu, Zhonghua Deng, Bin Li, Denghong Liu, and Renming Zhong. Analysis of the amplitude changes and baseline shifts of respiratory motion using intra-fractional cbct in liver stereotactic body radiation therapy. *Physica Medica*, 93:52–58, 2022.

[ZZLH13] Xiaomin Zhao, Ming J Zuo, Zhiliang Liu, and Mohammad R Hoseini. Diagnosis of artificially created surface damage levels of planet gear teeth using ordinal ranking. *Measurement*, 46(1):132–144, 2013.

[ZZX16] Bin Zhang, Lijun Zhang, and Jinwu Xu. Degradation feature selection for remaining useful life prediction of rolling element bearings. *Quality and Reliability Engineering International*, 32(2):547–554, 2016.