POLITECNICO MILANO 1863

Doctoral PhD program in

Mathematical Models and Methods in Engineering

Machine learning for precision medicine

A combination of data-driven and physics based models

Advisor Prof. **Paolo Zunino**

Coadvisors Prof. **Francesca Ieva**,
Prof. **Andrea Manzoni**,
Prof. **Anna Maria Paganoni**

Doctoral dissertation of
Nicola Rares Franco

Chair of the Doctoral program: Prof. **Michele Correggi**

Department of Mathematics
Year 2023 - XXXV cycle

Abstract

Precision medicine aims at improving the clinical treatment of patients by proposing subject-specific therapies, which are designed on the basis of individual characteristics, such as age and lifestyle, but also more complicated biological features. To this end, precision medicine mostly relies on *biomarkers*, complex indicators that characterize the genotype and phenotype of a patient. In pratical applications, such biomarkers are commonly derived by synthesizing the information coming from both data-driven and physics based models.

The aim of this Thesis is to explore a unified framework for quantitative methods in precision medicine, leveraging on Machine Learning tools. In particular, we focus on the case of personalized treatment planning of radiotherapy. Recently, many new research lines are being explored in this field, two of which are the main focus of this Thesis. The first one concerns the study of radiosensitivity as a genetic trait, and thus aims at identifying the genetic mutations associated with late toxicity in order to build suitable predictive biomarkers. The second line of research, instead, consists in the analysis of the cellular response to radiation by means of accurate and extensive numerical simulations. Both approaches present significant challenges, which in this Thesis are addressed through the development of new Machine Learning and Deep Learning algorithms.

In the first part of the Thesis, we focus on studying the connection between late toxicity and mutations in the DNA. There, the main difficulties arise from the presence of complex interactions among genetic loci and from the intrinsic class imbalance characterizing clinical data. To tackle these adversities, we take advantage of different Machine Learning tools, from deep autoencoders to data mining algorithms, ultimately developing a novel approach to polygenic risk scoring that enables the construction of interpretable interaction-aware biomarkers. Throughout the Thesis, we assess the scientific value of the proposed approach on both simulated and real data, showcasing the impact of our work on the clinical world.

Conversely, in the second part of the dissertation, we discuss how Deep Learning can be used to reduce the computational cost entailed by the numerical simulation of biophysical models relevant for radiotherapy, such as oxygen transfer models. In particular, we develop several strategies based on Deep Learning algorithms for replacing the original numerical solver with a cheaper, yet accurate, surrogate model. Thanks to these tools, the computational bottleneck entailed by using physics based numerical simulations in the complex workflow of biomarker discovery and validation can be completely resolved. From the methodological and, in particular, the mathematical standpoint, the proposed approaches are inspired by the flourishing literature of Reduced Order Modeling, but they also share unique benefits that distinguish them from state-of-art techniques, such as the ability of handling singularities, transport and mass propagation, in an extremely efficient way. In order to make our proposal mathematically sound, we also derive innovative theoretical results that support our reasoning. In particular, the Thesis contains innovative results about the latent dimension of autoencoders and the properties of convolutional neural networks. Finally, as a by-product of our studies, we also end up developing completely new tools, such as mesh-informed architectures, that, for their generality, stand out as independent topics of research.

Sommario

La medicina di precisione nasce con l'obiettivo di perfezionare i trattamenti clinici proponendo l'utilizzo di terapie personalizzate, le quali possono essere definite sulla base di caratteristiche individuali più o meno semplici, quali l'età e lo stile di vita, nonché altre caratteristiche biologiche più complesse. Per fare ciò, la medicina di precisione si affida principalmente ai *biomarcatori*, indicatori complessi in grado di sintetizzare il genotipo ed il fenotipo del paziente. Nella pratica, tali biomarcatori vengono generalmente ottenuti attraverso la sintesi di svariate fonti d'informazione, le quali possono provenire da dati sperimentali o da modelli fisico-matematici.

L'obiettivo di questa Tesi è quello di esplorare un contesto unificato dove sviluppare metodi quantitativi per la medicina di precisione, facendo leva, nello specifico, sugli strumenti offerti dal mondo del Machine Learning. Particolare attenzione è dedita alla personalizzazione dei trattamenti radioterapici, un settore che ha recentemente visto la nascita e lo sviluppo di diverse linee di ricerca, due delle quali costituiscono il focus principale di questa Tesi. La prima concerne lo studio della radiosensitività come tratto genetico, e mira dunque ad identificare quelle mutazioni genetiche che sono associate all'insorgenza della tossicità tardiva, con il fine ultimo di produrre opportuni biomarcatori predittivi della radiotossicità. La seconda linea di ricerca, invece, consiste nell'analisi della risposta cellulare alla radiazione, analisi resa possibile da un uso mirato ed estensivo di simulazioni numeriche altamenta accurate. Entrambi gli approcci presentano sfide particolarmente significative, le quali, all'interno di questa Tesi, vengono affrontate attraverso lo sviluppo di nuove tecniche di Machine Learning e di Deep Learning.

La prima parte della Tesi è focalizzata sullo studio del legame che sussiste tra tossicità tardiva e mutazioni del DNA. In questo contesto, le difficoltà principali nascono dal complesso meccanismo che regola l'interazione tra loci genetici, nonché dall'intrinseco sbilanciamento tipico del dato clinico. Nella Tesi, queste problematiche vengono affrontate adoperando svariati strumenti di Machine Learning, a partire dalle reti neurali autocodificanti fino agli algoritmi di estrazione dei dati. In definitiva, l'analisi messa in atto si configura nello sviluppo di un nuovo approccio al calcolo dei punteggi di rischio poligenico, portando, in ultima analisi, alla costruzione di biomarcatori capaci di preservare l'interpretabilità clinica e la capacità predittiva. Nel corso della Tesi, la metodologia proposta viene testata su dati reali afferenti ad un caso clinico ben preciso, nonché su diversi studi di simulazione, volendo così dimostrare come, in potenza, il lavoro presentato possa avere un riscontro immediato sul mondo clinico ed ospedaliero.

Nella seconda parte del trattato, invece, si discute di come le tecniche di Deep Learning possano coadiuvare la simulazione numerica di modelli biofisici utili alla radioterapia, quali i modelli di microcircolazione dell'ossigeno, grazie ad una sostanziale riduzione dei costi computazionali. Più precisamente, all'interno della Tesi, vengono sviluppate diverse strategie basate su algoritmi di Deep Learning, volte a rimpiazzare i solutori numerici con opportuni modelli surrogati, i quali possano essere meno dispendiosi ma, al contempo, sufficientemente accurati. Grazie a questi strumenti, l'onere computazionale implicato dall'utilizzo delle simulazioni numeriche nel complesso processo di ricerca e validazione dei biomarcatori, può essere completamente risolto.

Da un punto di vista prettamente metodologico e matematico, gli approcci proposti trovano ispirazione nella fiorente letteratura sulla Riduzione di Ordine di Modello; tuttavia, tali approcci godono anche di proprietà uniche che li distinguono dallo stato dell'arte, quali l'abilità di gestire in maniera estremamente efficiente fenomeni come il trasporto e la propagazione di massa, nonché la presenza di singolarità. Nel presentare tali approcci, particolare enfasi viene riposta nella loro controparte teorica, la quale, in più circostanze, viene avvalorata da opportune considerazioni matematiche. In particolare, la Tesi contiene diversi risultati innovativi sulla dimensione latente delle reti autocodificanti e sulle proprietà delle reti neurali convoluzionali. Infine, sulla scia dell'analisi proposta, vengono anche sviluppati strumenti interamente nuovi, quali le architetture neurali mesh-informed (letteralmente: "informate dalla griglia"): questi strumenti, in luce della loro generalità, possono essere considerati argomento indipendente di ricerca.

Contents

To improve the experience of the reader, we have marked those parts that we believe contain the most significant contributions, in terms of novelty and scientific value. We use "◆" to highlight novel theoretical results, while we use "◇" for those contributions with a more algorithmic/computational flavor.

Introduction

The purpose of precision medicine, or personalized medicine, is to enhance clinical treatments by proposing subject-specific therapies that are tailored at the patient level. By taking into account the different characteristics of each patient, precision medicine aims at ensuring the right treatment at the right time, with the intention of improving the final outcome of medical healthcare [107].

Clearly, the idea itself is far from being new. In fact, we may trace the origins of precision medicine back to the late eighteenth century, when statistical inference was firstly applied to design medical treatments [107]. However, things have changed a lot since then, especially during the last century, with the advent of genomics, computers and numerical modeling. Currently, precision medicine is an extremely active and cross-disciplinary area of research that counts a large number of successful applications, which definitely shows the importance of including patients characteristics in the development of treatment planning systems.

In some cases, even basic information about the patient, such as lifestyle, age or sex, turns out be relevant. For instance, we can think of how certain habits affect the outcome of medical treatments, as in the case of smokers with periodontitis [149], or how certain drugs produce different results in males and females, as in the case of serotonergic antidepressants [20]. Conversely, in other circumstances, more complicated genotypic or phenotypic traits are considered, such as the presence of mutations and their expression rates (see e.g. the case of epileptic disorders [47]), or the shape of an organ and its mechanical properties (see e.g. the case of coronaries interventions for myocardial ischaemia [198]).

One of the practical challenges of precision medicine is thus related with the management of incredibly large amounts of heterogeneous information, a scenario that, in the big data era, is increasingly important. To deal with such complexity, many approaches to precision medicine rely on the so-called *biomarkers*, suitable indicators that aim at synthesizing different aspects of a given phenomenon. For instance, polygenic risk scores are genetic biomarkers that summarize the impact of multiple genetic mutations over a specific phenotype. The discovery and development of such indicators is usually based on purely data-driven methodologies, with a possible integration of external biological knowledge.

In addition to data-driven biomarkers, which are undoubtedly of paramount importance, a significant contribution is also provided by a diametrically opposite class of indicators, which we may refer to as physics based biomarkers. Differently from the previous, the latter are derived from accurate physical and physio-biological models that aim at describing a given biological phenomenon in mathematical terms. In this case, researchers exploit physical models and numerical simulations to discover new biological mechanisms and to derive physically interpretable biomarkers that can aid the optimization of clinical treatments.

Nonetheless, the development of both data-driven and physics based biomarkers poses significant challenges to scientists and researchers. Among these, one of the fundamental barriers is generalizability. In the case of data-driven approaches, this corresponds to ensuring that the data at hand, over which a discovery or a methodology was developed, represents the rest of the population sufficiently well. For physics based biomarkers, instead, the difficulty is related to the reliability of the physical model itself, which, being an approximation of a real biological system, first needs to be validated. However, in addition to these, there are

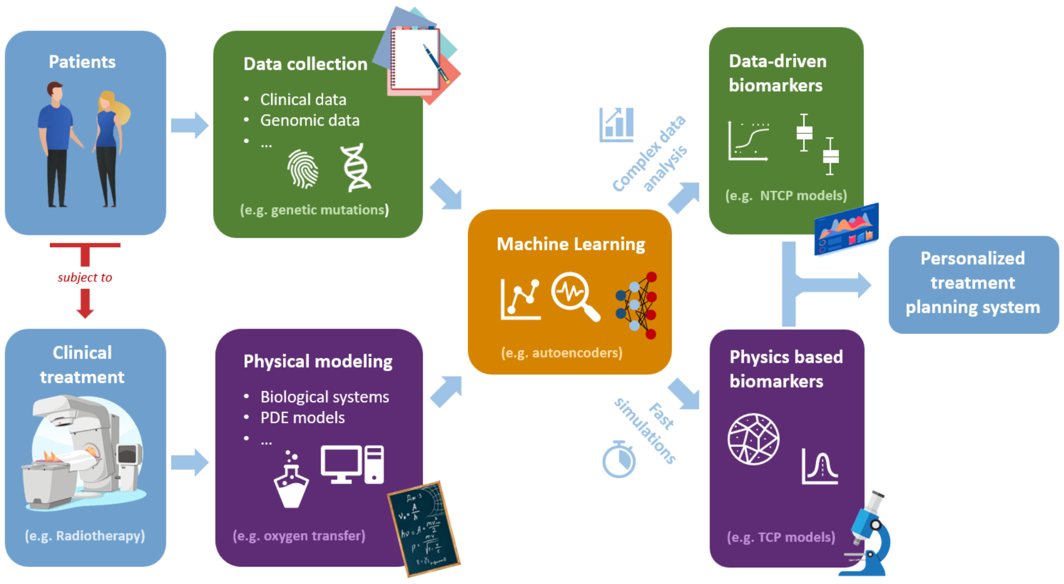


Figure 1: A unified framework for precision medicine where Machine Learning algorithms enable the fruitful combination of data-driven and physics based approaches.

also other problems that depend specifically on the context.

For instance, let us briefly mention the difficulties commonly encountered when dealing with genetic biomarkers. During the discovery phase, the main drawback is arguably given by the computational cost that is required to analyze the massive amount of information available in the DNA. Still, even after the identification of useful genetic variants/loci, the construction of predictive biomarkers is far from being straightforward. In fact, most of the times, the effect of single genetic loci over the phenotype is too small to grant any meaningful predictive power. Even when combining multiple genes, as in polygenic risk scores, one may end up with biomarkers that add little to nothing with respect to what is already known, leading health professionals to favor simpler and more standard clinical models, see e.g. [171, 175]. Nevertheless, these negative findings might be a hint that we are synthesizing genetic information in the wrong way, calling for novel approaches that are able to summarize complex genetic mechanisms.

In the case of physics based biomarkers, instead, one of the main difficulties is given by the calibration of model parameters, which may be patient specific and subject to uncertainty, ultimately leading to unreliable predictions. Addressing these issues requires a careful and extensive exploration of the physical model, a so-called *many-query scenario*, which in turn calls for numerous numerical simulations. Because this can lead to a dramatic increase in the overall computational cost, the applicability of these approaches can be severely limited. For this reason, many researchers (us included) are now working on novel ways for circumventing these drawbacks.

The purpose of this Thesis is to address these methodological and computational issues by leveraging on Machine Learning algorithms, ultimately developing a unified framework for the construction of predictive biomarkers. In particular, as we detail below, we shall focus on a specific application to precision medicine concerning the personalized treatment planning of radiotherapy for oncological patients. Nonetheless, thanks to the use of mathematical tools, we shall end up with completely general methodologies that can be easily adapted to other clinical settings.

Personalized treatment planning of radiotherapy

Radiotherapy, also known as radiation therapy, is a well-known medical treatment that is commonly employed for killing cancer cells in oncological patients. It is based on the use of ionizing radiation, which enables the effective eradication of tumor masses. In practical applications, the irradiation of the interested region can be carried out in several ways. One of the most common is external beam radiotherapy, where a machine is used to suitably target beams at cancer cells. However, other approches exist as well, such as brachytherapy, where the tumor is irradiated by placing a radioactive material next to it (inside the body). In general, though, radiotherapy is a treatment that is carried out over multiple sessions, usually covering a time interval of 5 to 8 weeks, with a suitable alternation between periods of treatment and days of rest according to the different national health system protocols.

As radiotherapy is one of the primary approaches in oncological healthcare, concerning up to 50-67% of the patients, there is a constant effort in trying to make it better and better. In particular, we may identify two main research directions that aim at improving the final outcome of radiotherapy. On the one hand, there is the interest in limiting long term complications, as the repeated use of radiations can cause mild to severe side-effects. On the other hand, there is the optimization of the treatment itself, which aims at a more efficient eradication of tumors, possibly reducing the total treatment time and the exposure to radiation. In both cases, though, it is unlikely that a substantial improvement will be obtained through the employment of more sophisticated or technologically up-to-date machines. Instead, precision medicine might actually lead to a significant breakthrough in the world of radiotherapy. We discuss how below.

Avoiding long term toxicity: NTCP models

In general, sources of radiation, such as X-rays, can be extremely aggressive on biological organisms. As a consequence, it is very common that healthy tissues nearby the tumoral region are damaged during radiotherapy. While some of these wounds recover within days, permanent damage is also likely to happen and, after years, the latter may give rise to long term complications (a phenomenon known as *late toxicity*). For instance, prostate cancer patients that have received radiation doses to the bladder may experience incontinence or presence of blood in the urine. Conversely, breast cancer patients may report oedemas or fibrosis near the mammary glands.

Ideally, these complications could be avoided if we could fire radiation beams with a very high precision. However, this is not the only way to prevent late toxicity. In fact, it is now acknowledged that patients have an individual subsceptibility to radiation, which is commonly referred to as *radiosensitivity*: while some people are more vulnerable to radiation and thus prone to experiencing side-effects, others are intrinsically more resistant. If we were able to know beforehand the radiosensitivity of each patient, we could tailor the treatment of radiotherapy to reduce the probability of late toxicity. For instance, for those patients at a higher risk, we could favor hyperfractionation (that is, a schedule with smaller doses of radiation given over a longer period of time), or we could suggest the daily use of much careful procedures such as IGRT (Image Guided Radiation Therapy). Conversely, highly resistant patients could be treated with higher doses to speed up their treatment cycle (hypofractionation).

In the literature, the probability of long term toxicities is estimated using the so-called Normal Tissue Complication Probability (NTCP) models. The latter aim at quantifying the radiosensitivity of a given patient on the basis of several quantities, including dosimetric variables (e.g. amount of dose per day) and individual characteristics (e.g. age, presence of comorbidities etc.). To this end, a classical approach to NTCP modeling is to use Logistic Regression models. Namely, if Y is the binary outcome standing for late toxicity, and $X_1, \dots, X_p$ are the model covariates, the probability of long term complications is estimated as

$$\mathbb{P}(Y=1)=\left[1+\exp\left(-\beta_0-\beta_1X_1+\cdots-\beta_mX_m\right)\right]^{-1} \tag{1}$$

where $\beta_0,\ldots,\beta_m$ are suitable coefficients, to be estimated from the available clinical data, that quantify the effect size of each predictor.

Recently, there is an increasing interest in finding ways for integrating NTCP models with genetic information. In fact, several studies have now highlighted that radiosensitivity is a heritable human trait, implying that genetic variants might play a remarkable role in predicting late toxicity. However, the integration of genetic scores in NTCP models is still in the process of being. In fact, the first attempts in this direction have encountered severe limitations, typically yielding genetic scores that add no predictive power with respect to classical clinical/dosimetric models. As we mentioned before, the reason might lie in the complex interactions among genetic loci, a puzzling mechanism that is often very hard to capture. As we shall discuss later on, one of the purposes of this Thesis is to give a contribution towards this direction, leveraging on Machine Learning tools.

Optimizing the treatment efficiency: TCP models

At the moment, radiation oncologists rely on well-established physical models to calculate radiation doses and schedule treatments. The most famous one is arguably the so-called *linear-quadratic model*, which exploits the relationship between oxygen partial pressure and resistance to radiation to estimate the probability of tumor eradication. These models are commonly referred to as Tumor Control Probability (TCP) models.

The idea goes as follows. Previous studies in the literature have highlighted that the concentration of oxygen in biological tissues significantly affects the outcome of radiotherapy. This is because oxygen molecules can magnify the strength of high energy beams: as a consequence, radiotherapy becomes more aggressive in highly supplied regions and less effective were oxygen is missing (the so-called *hypoxia* regime). In mathematical terms, this is modeled as

$$S_f(\mathrm{O}_2,D)=\exp\left(-\alpha(\mathrm{O}_2)D-\beta(\mathrm{O}_2)D^2\right), \tag{2}$$

where O_2 is the oxygen partial pressure, defined over a suitable region of the body $\Omega\subset\mathbb{R}^3$, D is the radiation dose and S_f is the survival fraction. More precisely, $S_f(\mathrm{O}_2(\mathbf{x}),D)$ denotes the fraction of cells at point $\mathbf{x}\in\Omega$ that are still alive after the treatment. Here, $\alpha,\beta:\mathbb{R}\to\mathbb{R}$ are suitable functions that have already been characterized in the literature. The survival fraction S_f is then used to estimate the probability of complete tumor eradication. The latter is commonly modeled as

$$\mathrm{TCP}(D)=\exp\left(-\int_\Omega N(\mathbf{x})S_f(\mathrm{O}_2(\mathbf{x}),D)d\mathbf{x}\right), \tag{3}$$

where $N(\mathbf{x})$ is the number of clonogenic cells per unit volume at the point $\mathbf{x}\in\Omega$. The power of this model, which has been extensively studied and tested, is that it provides an estimate of a successful treatment given the radiation dose.

However, this assumes that the distribution of the oxygen partial pressure is actually known, which, in practice, is never the case. To account for this, researchers rely instead on accurate numerical models that simulate the transfer of oxygen in biological tissues. Still, a proper description of the tissue perfusion near tumors is extremely difficult because of the small scales involved (capillaries, which are the main source of oxygen have a diameter of 8 to 10 micron), and because of the peculiar structure of tumoral regions, which usually feature irregular vessels and hypoxic zones. In particular, even though we may have accurate numerical simulations at our hand, we also need ways for assessing their uncertainty and reliability. As a matter of fact, this is the second purpose of this Thesis: to develop Deep Learning techniques that can enable such evaluations by making them computationally feasible.

Machine Learning: a twofold approach envisioning precision medicine

Machine Learning is a remarkably broad field that concerns the analysis of large datasets and their employment in different tasks, such as predictive modeling, pattern recognition and data representation. It is sometimes considered as a subfield of Artificial Intelligence, even though some researchers disagree and only acknowledge the existence of an overlap between the two subjects [138]. Instead, it is highly recognized that Machine Learning is intimately related to Statistics, as both fields exploit sample data to draw conclusions. Indeed, the two share several methodologies, and their difference is sometimes more of a philosophical matter rather than a practical one, see e.g. [90].

Nonetheless, Machine Learning also comes with unique and remarkable tools such as neural networks, which constitute the foundations of Deep Learning. The latter is an extremely flourishing field that counts many successful applications, from speech recognition to computer vision, and that is recently being enriched with a preeminent mathematical theory, see e.g. the comprehensive overview in [53]. Our interest in Machine and Deep Learning is thus motivated by their uttermost flexibility and by their capacity to handle, and exploit, complex high dimensional data. In particular, we believe that these tools can significantly aid the study of NTCP and TCP models, fostering the development of a unified framework for precision medicine and personalized treatment planning. To further explain our vision, we detail below how, throughout the Thesis, we plan to apply Machine Learning to the case of radiotherapy treatments.

Machine Learning and Statistics: Boosting NTCP models with novel genetic biomarkers

In the context of NTCP modeling, we propose the use of Machine Learning to build informative genetic risk scores that maintain their predictive power when integrated in clinical/dosimetric models. In short, the idea goes as follows. Given a set of genetic variants $S_1, \dots, S_q$, we employ a Deep Learning algorithm based on outlier detection techniques to extract a smaller subset

$$\{S_1, \dots, S_q\} \longrightarrow \{S_{j_1}, \dots, S_{j_p}\},$$

with $p \ll q$, that is informative with respect to the occurrence of late toxicity, while also accounting for nonlinear interactions. Then, we exploit Data Mining techniques to identify suitable predictive patterns and build a high-order interaction-aware polygenic risk score,

$$hi\text{PRS} = g(S_{j_1}, \dots, S_{j_p}),$$

where g is a suitable nonlinear function. The idea is then to include the latter in NTCP models, such as (1), by introducing an additional predictor $X_{m+1} := hi\text{PRS}$ that synthesizes the genetic information. This is ultimately the goal of Chapters 1, 2 and 3.

Our methodological and scientific contributions in this context are well synthesized by the following publications:

[135] 📖 Massi MC, Gasperoni F, Ieva F, Paganoni AM, Zunino P, Manzoni A, **Franco NR**, et al. (2020). *A deep learning approach validates genetic risk factors for late toxicity after prostate cancer radiotherapy in a REQUITE multi-national cohort*, Frontiers in oncology.

This first work is about the use of deep learning for the selection of genetic variants in a clinical case study concerning prostate cancer patients treated with external beam radiotherapy. In the Thesis, this aspect is addressed in Chapter 2.

[133] 📖 Massi MC, **Franco NR**, Manzoni A, Paganoni AM, Park H et al. (2023). *Learning High-Order Interactions for Polygenic Risk Prediction*, PLOS ONE.

This second work is a natural prosecution of the previous and concerns the problem of building predictive genetic biomarkers that include interaction effects. It is a purely methodological contribution that defines the foundations of Chapter 3.

[68] 📖 **Franco NR**, Massi MC, Ieva F, Manzoni A, Paganoni AM, Zunino P et al. (2021). *Development of a method for generating SNP interaction-aware polygenic risk scores for radiotherapy toxicity*, Radiotherapy and Oncology.

This third work is a clinical application of the methodology addressed in the previous contribution and it is also described in Chapter 3.

[165] 📖 Rancati T, Massi MC, **Franco NR** et al. (2021). *Prediction of toxicity after prostate cancer RT: the value of a SNP-interaction polygenic risk score*, Radiotherapy and Oncology.

This fourth work is an extension of the previous paper in which the proposed genetic biomarker is integrated into clinical/dosimetric NTCP models. This aspect is also addressed in Chapter 3.

Machine Learning and physical models: Studying TCP models through numerical simulations

Regarding TCP models, instead, we propose the use of Deep Learning in replacement to expensive numerical simulations, by adopting the perspective of reduced order modeling techniques. In particular, we consider the case in which a suitable oxygen transfer model is available (either as an open or a black-box tool). The latter defines a natural map from the model parameters $\boldsymbol{\mu}$, which may describe both physical and geometrical quantities (e.g. the permeability of the blood vessels and the topology of the vascular network), to the oxygen partial pressure O_2, namely

$$\boldsymbol{\mu} \longrightarrow \mathrm{O}_2^{\boldsymbol{\mu}}.$$

The idea is then to build a nonintrusive model Φ based on deep neural networks such that

$$\Phi(\boldsymbol{\mu}) \approx \mathrm{O}_2^{\boldsymbol{\mu}}, \tag{4}$$

which, thanks to Equations (2) and (3) would then enable a comprehensive and robust analysis of TCP models. In order to construct such Φ, we exploit the numerical solver as a generator of trusted samples, ultimately turning our problem into one of Statistical Learning. Throughout Chapters 4, 5, 6 and 7, we develop suitable strategies that go towards this direction, while also providing mathematical insights on their practical implementation, design and performance. In doing so, we do not limit ourselves to the specific case of radiotherapy applications, but rather take a much broader perspective.

Our research along this direction is well represented by the publications and preprints below.

[65] 📖 **Franco NR**, Manzoni A, Zunino P (2022). *A deep learning approach to reduced order modeling of parametrized partial differential equations*, Mathematics of Computation.

In this first work, which constitutes the core of Chapter 5, we develop a Deep Learning approach for the efficient approximation of the parameter-to-solution map of parametrized PDEs, keeping Equation (4) in mind.

[66] 📖 **Franco NR**, Fresca S, Manzoni A, Zunino P (2023). *Approximation bounds for convolutional neural networks in operator learning*, Neural Networks.

This second work has instead a purely mathematical flavor. Here, we assess the ability of convolutional neural networks in learning nonlinear operators, providing novel insights on their application to parametrized PDEs (and thus of interest for our purposes). This work is addressed in Chapter 6.

[67] 📖 **Franco NR**, Manzoni A, Zunino P (2022). *Learning operators with mesh-informed neural networks*, arXiv preprint.

In this last work, we develop a novel class of neural network architectures that can process complex high dimensional data defined over generic spatial domains, thus extending the applicability of the aforementioned approaches. There, we also discuss a first application to oxygen transfer models. The latter is anticipated in Chapter 4, while the methodology is described in full detail in Chapter 7.

Outline of the Thesis

The Thesis is organized as follows. In Part 1, we focus on the development of genetic biomarkers and their integration in NTCP models. More precisely, in Chapter 1, we discuss the relationship between DNA mutations and sensitivity to radiation, introducing all the required biological concepts and providing the description of an actual clinical cohort.

Then, in Chapter 2, we address the problem of identifying useful genetic variants, highlighting the corresponding challenges and presenting a solution based on Deep Learning algorithms, specifically deep autoencoders, originally developed by Massi et al. [135]. The latter approach, which shares interesting connections with Chapter 5 and is ultimately propedeutical to Chapter 3, is then applied to a clinical case study concerning radiotherapy toxicity in prostate cancer patients.

We then devote Chapter 3 to one of our main contributions to the field: the development of a novel interaction-aware polygenic risk score, obtained by leveraging on the data mining literature. We assess the capabilities of the proposed approach through an extensive set of simulation studies and report the results obtained in the clinical application addressed in the previous Chapter, with additional insights on integrated NTCP models.

In the second Part of the Thesis, instead, we move to physics based approaches and TCP models. Specifically, in Chapter 4, we introduce the radiobiological model and its relationship to precision medicine. There, we highlight the main challenges and limitations encountered by the state-of-the-art, discussing on the potential benefits provided by Deep Learning. In particular, we showcase our results in an idealized setting, leaving the detailed description and derivation of the methodologies to the remaining Chapters.

Specifically, in Chapter 5, we present our Deep Learning approach to model order reduction, providing both theoretical and practical insights. There, even though in a different way with respect to Chapter 2, the autoencoder architecture ends up being the main character on stage. In particular, the primary novelty of Chapter 5 resides in our theoretical results, where we are able to explicitly characterize the so-called *latent dimension*.

In the proposed approach, aside from autoencoders, another fundamental ingredient are te so-called convolutional neural networks. For this reason, we devote Chapter 6 to a deeper and rigorous study of these architectures, ultimately deriving novel approximation bounds that are completely original with respect to the vivid and rapidly evolving field of constructive approximation applied to Deep Learning. Then, in Chapter 7, we generalize our approach to arbitrarily complicated geometries, introducing a novel class of sparse neural network architectures termed Mesh-Informed Neural Networks. There, after a suitable validation against several benchmark examples, we return to the original question about TCP models anticipated in Chapter 4.

Finally, in the last Chapter, we wrap everything up and draw the corresponding conclusions, with an in depth discussion about the two Parts and their potential in providing a unified framework for precision medicine.

Last but not least, at the beginning of the dissertation, we have included a preliminary Chapter with the fundamental concepts of Deep Learning used throughout the Thesis. In doing so, we have adopted an intrinsically mathematical perspective that, we hope, will make the reader further appreciate our work.

Mathematical foundations of Deep Learning

In this preliminary Chapter, we provide a mathematical overview of the fundamental ideas at the core of Deep Learning, one of the most flourishing areas of Machine Learning. In particular, we introduce concepts such as neural network, layer and activation function. Then, we discuss the training of neural network models and provide general insights about their capabilities, both in terms of generalization and expressivity. All these notions, as well as the corresponding mathematical results, will serve as guidelines for the rest of the Thesis.

Deep Learning is arguably one of the most successful areas of Machine Learning, and it consists in the study and development of models based on artificial neural networks. The origins of Deep Learning can be traced back to the early '60s, when Frank Rosenblatt first proposed a probabilistic model called *the perceptron* [170], which he later generalized to the *multi-layer perceptron* [169]. Now, we refer to these models as to shallow and deep (artificial) neural networks, respectively. However, it is not only the names that have changed. Today we also understand neural networks through a completely new formalism which is mostly based on Linear Algebra, Statistics and Functional Analysis. Our purpose for the next Sections is to present the fundamental notions of Deep Learning through this mathematical language. Throughout the whole Chapter, we shall make use of basic concepts of Probability Theory and Functional Analysis. If needed, the reader can refer to [91, 96] and [60, 203], respectively.

Deep neural networks

Originally inspired by the connections of neurons in the human brain, Deep Neural Networks (DNNs) are computational models that have become extremely popular after their tremendous successes in areas such as language processing [73], autonomous driving [101], computer vision and image processing [188]. Nowadays, DNNs have also found application in several scientific areas, such as Statistics and Numerical Analysis, which has gained them the attention of both applied and theoretical mathematicians.

Mathematically speaking, DNNs are a class of nonlinear approximators that is ultimately based on the composition of affine and nonlinear transformations. Here, we shall focus on DNNs having a so-called *feedforward* architecture. We report below some basic definitions, starting from the fundamental building block of a DNN model, the *layer*.

Definition 0.1. *Let $m,n \geq 1$ and $\rho:\mathbb{R}\to\mathbb{R}$. A layer with activation ρ is a map $L:\mathbb{R}^m\to\mathbb{R}^n$ of the form $L(\mathbf{v})=\rho(\mathbf{W}\mathbf{v}+\mathbf{b})$, for some $\mathbf{W}\in\mathbb{R}^{n\times m}$ and $\mathbf{b}\in\mathbb{R}^n$.*

In the literature, $\mathbf{W}$ and $\mathbf{b}$ are usually referred to as the *weight matrix* and *bias vector*, respectively. We point out that the above definition contains an abuse of notation, as ρ is evaluated over an n-dimensional vector. We understand the latter operation component-wise, that is

$$\rho([x_1,\dots,x_n]) \coloneqq [\rho(x_1),\dots,\rho(x_n)]$$

whenever $\rho : \mathbb{R} \to \mathbb{R}$. From here, DNN models are defined through the composition of multiple layers.

Definition 0.2. *Let $m, n \geq 1$. A neural network of depth $l \geq 0$ is a map $\Phi : \mathbb{R}^m \to \mathbb{R}^n$ obtained via composition of $l+1$ layers, $\Phi = L_{l+1} \circ \dots L_1$.*

The layers of a neural network do not need to share the same activation function and usually the output layer, L_{l+1}, does not have one, see e.g. [112, 176]. Architectures with $l = 1$ are known as shallow networks, while the adjective deep is used when $l \geq 2$. Note that we also allow for the degenerate case $l = 0$, where the DNN actually reduces to the output layer. This is somewhat unusual, but it will help us in making the notation lighter. Finally, we say that Φ has activation function ρ, or equivalently that Φ is a ρ-DNN, if all of its (hidden) layers share that same activation. Common choices for ρ include smooth functions, such as the the hyberbolic tangent,

$$\tanh: \quad x \to \frac{\mathrm{e}^x - \mathrm{e}^{-x}}{\mathrm{e}^x + \mathrm{e}^{-x}},$$

or the sigmoid, $x \to \tanh(x/2)/2 + 1/2$, but also piecewise linear maps, such as the ReLU activation,

$$\text{ReLU}: \quad x \to \max\{x, 0\},$$

or the α-leaky ReLU activation, $x \to \max\{x,0\} + \alpha\min\{x,0\}$.

The classical pipeline for building a DNN unit starts with the design of the model architecture. This corresponds to choosing the depth l and the number of neurons within each layer L_i, that is the output dimension n_{i+1}. In particular, at this stage, the entries of the weights and biases do not have specific values: these are to be learned later on by exploiting the data available, a procedure commonly known as *training phase*. We postpone this discussion, which is of crucial importance, to the next Section.

Remark. In principle, we should distinguish between the model architecture and its realization, as some authors correctly do, see e.g. [112, 131]. In fact, for instance, we cannot retrieve the depth of a DNN if we have no knowledge of its architecture, or at least we cannot do this without ambiguity. To see this, assume that $f : \mathbb{R}^m \to \mathbb{R}^n$ is some DNN. Let now L be the identity layer. Then, $f = L \circ f$ but, strictly speaking, the two have different depths. However, as soon as we grant some context these ambiguities can be avoided. Thus, we prefer not to make this distinction and, instead, keep the notation lighter.

Model training and evaluation

DNN models only become operational once we tune their weights and biases. This procedure typically involves the optimization of a suitable objective function, and it is commonly referred to as training phase. In the most simple scenario, the training of a DNN model is a form of nonlinear regression, which thus consists of a purely data-driven routine. Given a dataset of desired input-output pairs, say $\{(\mathbf{v}_i, \mathbf{u}_i)\}_{i=1}^N \subset \mathbb{R}^m \times \mathbb{R}^n$, and having fixed the architecture of a DNN $\Phi : \mathbb{R}^m \to \mathbb{R}^n$, one optimizes the weights and biases in Φ by minimizing a quantity of the form

$$\mathscr{L}(\Phi) = \frac{1}{N}\sum_{i=1}^N \ell\left(\mathbf{u}_i,\ \Phi(\mathbf{v}_i)\right) + \mathscr{R}(\Phi). \tag{0.5}$$

Here, $\ell : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is some measure of discrepancy, such as the Euclidean metric, while $\mathscr{R}$ is a (nonlinear) functional that introduces some regularization. For instance, the latter may involve the L^2-norm of Φ or, if possible, other energy functionals, such as

$$\mathscr{R}(\Phi) = \int_{\mathbb{R}^p} |\Delta\Phi(\mathbf{v})|^2 d\mathbf{v}.$$

As a matter of fact, in some cases, as in the so-called Physics-Informed Neural Networks [164], the regularization term can be the residual of some partial differential equation that Φ is required to satisfy. Then, one may even consider discarding the data-driven term in (0.5).

In the literature, $\{(\mathbf{v}_i, \mathbf{u}_i)\}_{i=1}^N$ is referred to as *training set*, while the term *loss function* is used for $\mathscr{L}$. In general, because of the way in which Φ depends on its weights and biases, the training of a DNN model is a very difficult problem of nonconvex optimization. Furthermore, as DNN architectures often have more than thousands of degrees of freedom, (0.5) ultimately translates into an extremely high-dimensional problem with many local minima. For all these reasons, the optimization of the loss function is usually carried out using batching strategies and stochastic gradient descent algorithms [174]. Among these, the most common optimizers are arguably *Adam*, an adaptive first-order method that exploits momentum [102], and L-BFGS (Limited-memory BFGS), a quasi-Netwon method that is based on estimates of the inverse Hessian matrix [124].

After the training phase, it is fundamental to check whether the model obtained is able to generalize over unseen data. To this end, it is a common practice to assess the DNN performance by relying on an external *test set*, $\{(\mathbf{v}_i^{\text{test}}, \mathbf{u}_i^{\text{test}})\}_{i=1}^{N_{\text{test}}}$. If such a collection of data is available, then the quality of the DNN model can be measured in terms of quantites such as

$$E(\Phi) = \frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} e\left(\mathbf{u}_i^{\text{test}},\ \Phi(\mathbf{v}_i^{\text{test}})\right), \tag{0.6}$$

where $e : \mathbb{R}^n \times \mathbb{R}^n \to [0, +\infty)$ is a suitable error function. If $E(\Phi)$ is large, then Φ is said to not generalize well, and that is irrespectively of the performance registered during the training phase. In particular, we talk about *overfitting* whenever the DNN performes well over the training set but poorly over the test set.

Of note, if we endow the input-output space with a probability distribution $\mathbb{P}$, and if the test samples are drawn independently from $\mathbb{P}$, then we can interpret (0.6) as a Monte Carlo estimate of

$$\mathscr{E}(\Phi) = \mathbb{E}\left[e(U, \Phi(V))\right], \tag{0.7}$$

where $\mathbb{E}$ denotes the expectation operator, while $(V, U) \sim \mathbb{P}$ are random variables describing the input-output pairing. In this sense, there is a strong connection between DNNs and the emerging field of Statistical Learning. Still, as we shall discuss below, we can also understand a lot about neural networks if we adopt the perspective of Numerical Analysis, specifically Approximation theory.

Error decomposition and upper-bounds

Let $\mathbb{P}$ be a probability distribution over the input-output space, $\mathbb{R}^m \times \mathbb{R}^n$, and let $(V, U) \sim \mathbb{P}$. We shall assume that the output variable is square-integrable in the Bochner sense, that is $\mathbb{E}\|U\|^2 < +\infty$, and that the input variable is almost surely bounded, $\|V\| \le R$ for some $R > 0$. For the sake of simplicity, we shall also restrict to the case in which errors and losses are measured using the Euclidean norm $\|\cdot\|$. Then, (0.5) and (0.7) become

$$\mathscr{L}(\Phi) = \frac{1}{N}\sum_{i=1}^N \|\mathbf{u}_i - \Phi(\mathbf{v}_i)\|, \qquad \mathscr{E}(\Phi) = \mathbb{E}\|U - \Phi(V)\|.$$

The difference of the two quantifies how well the training set can represent the underlying phenomenon, and it is commonly referred to as *generalization gap*,

$$\mathscr{G}_{\text{gap}}(\Phi) := |\mathscr{E}(\Phi) - \mathscr{L}(\Phi)|\,.$$

Let us assume to have fixed, once and for all, a suitable neural network architecture. Then, let $\mathscr{H}$ be the collection of all possible DNN models that can be obtained with such an architecture (that is, by only changing the entries in the weights and biases). In order to analyze the role played by all the ingredients, it is useful to define the following maps:

- the conditional expectation of U given $V = \mathbf{v}$,

  $$\Phi_{\text{ideal}}(\mathbf{v}) := \mathbb{E}\left[U \mid V = \mathbf{v}\right].$$

  This is our ideal objective as, among all square-integrable and V-measurable maps, it is the one that minimizes $\mathscr{E}(\Phi)$. In general, it is not a DNN and it does not belong to our hypothesis space $\mathscr{H}$;

- our best candidate (assuming it exists),

  $$\Phi_{\text{best}}(\mathbf{v}) := \underset{\tilde{\Phi}\in\mathscr{H}}{\operatorname{argmin}}\ \sup_{\|\mathbf{v}\|\le R} \|\Phi_{\text{ideal}}(\mathbf{v}) - \tilde{\Phi}(\mathbf{v})\|$$

  that is, the DNN model that better resembles Φ_{ideal}. Note that, here, we adopt a worst-case scenario approach to measure errors. While not necessary, this will simplify our reasoning later on;

- finally, the solution to our empirical optimization problem,

  $$\Phi_{\text{optimal}}(\mathbf{v}) := \underset{\tilde{\Phi}\in\mathscr{H}}{\operatorname{argmin}}\ \mathscr{L}(\tilde{\Phi})$$

  that is, the DNN sought during the training phase. Once again, we assume the above to exist.

We can exploit these definitions to bound $\mathscr{E}(\Phi)$ in terms of several quantities that describe different aspects of the problem. Given $\Phi \in \mathscr{H}$, we have,

$$\begin{aligned}\mathscr{E}(\Phi) &\le \mathscr{L}(\Phi) + \mathscr{G}_{\text{gap}}(\Phi) = \\ &= \mathscr{L}(\Phi_{\text{optimal}}) + \mathscr{L}(\Phi) - \mathscr{L}(\Phi_{\text{optimal}}) + \mathscr{G}_{\text{gap}}(\Phi) \le \\ &\le \mathscr{L}(\Phi_{\text{best}}) + \mathscr{L}(\Phi) - \mathscr{L}(\Phi_{\text{optimal}}) + \mathscr{G}_{\text{gap}}(\Phi).\end{aligned} \tag{0.8}$$

Now, as a direct consequence of the triangular inequality,

$$\begin{aligned}\mathscr{L}(\Phi_{\text{best}}) &\le \mathscr{L}(\Phi_{\text{ideal}}) + \frac{1}{N}\sum_{i=1}^{N} \|\Phi_{\text{ideal}}(\mathbf{v}_i) - \Phi_{\text{best}}(\mathbf{v}_i)\| \le \\ &\le \mathscr{E}(\Phi_{\text{ideal}}) + \mathscr{G}_{\text{gap}}(\Phi_{\text{ideal}}) + \sup_{\|\mathbf{v}\|\le R} \|\Phi_{\text{ideal}}(\mathbf{v}) - \Phi_{\text{best}}(\mathbf{v})\|.\end{aligned} \tag{0.9}$$

Finally, by putting together (0.8) and (0.9), we get

$$\mathscr{E}(\Phi) \le \mathscr{E}(\Phi_{\text{ideal}}) + \mathscr{S}_{\text{error}} + \mathscr{A}_{\text{error}} + \mathscr{O}_{\text{error}} \tag{0.10}$$

where we have defined, respectively, the *sample error*

$$\mathscr{S}_{\text{error}} := \mathscr{G}_{\text{gap}}(\Phi_{\text{ideal}}) + \mathscr{G}_{\text{gap}}(\Phi)$$

the *optimization error*

$$\mathscr{O}_{\text{error}} := |\mathscr{L}(\Phi_{\text{optimal}}) - \mathscr{L}(\Phi)|.$$

and the *approximation error*

$$\mathscr{A}_{\text{error}} := \inf_{\tilde{\Phi}\in\mathscr{H}} \sup_{\|\mathbf{v}\|\leq R} \|\Phi_{\text{ideal}}(\mathbf{v}) - \Phi(\mathbf{v})\|.$$

By definition, $\mathscr{E}(\Phi_{\text{ideal}})$ measures the extent to which the input variable can explain the output variable, so there is nothing that we can do about it. The sample error $\mathscr{S}_{\text{error}}$, instead, is a purely statistical quantity that depends on the available training data. Conversely, the optimization error $\mathscr{O}_{\text{error}}$ has more of a computational flavor, as it is all about our ability of properly minimizing the loss function. Finally, the approximation error $\mathscr{A}_{\text{error}}$ is where the choice of the network architecture comes into play, and thus constitutes a theoretical question for numerical analysts. In light of these considerations, it is worth to comment (0.10) through the eyes of both Statistics and Numerical Analysis, specifically Approximation theory.

The perspective of Statistics

The statistical core of (0.10) lies in the sample error, $\mathscr{S}_{\text{error}}$. The latter is composed of two similar quantities. The first one, is the generalization error of Φ_{ideal}, and it is arguably the easier quantity to address. To see this, we define the random variable $X := \|U - \Phi_{\text{ideal}}(V)\|$. If the observations in the training set are drawn independently, then the values $\{x_i\}_i := \{\|\mathbf{u}_i - \Phi_{\text{ideal}}(\mathbf{v}_i)\|\}_i$ become i.i.d. realizations of X. Hence, if we take the expectation over all training sets $\mathcal{S}_N$ of size N, by the law of large numbers,

$$\mathbb{E}_{\mathcal{S}_N}|\mathscr{G}_{\text{gap}}(\Phi_{\text{ideal}})| \leq \sqrt{\frac{\text{Var}\|U - \Phi_{\text{ideal}}(V)\|}{N}}. \tag{0.11}$$

Note that, due to the presence of $\mathbb{E}_{\mathcal{S}_N}$, the above is not a uniform upper-bound, rather it is a probabilistic one. If we repeatedly sample from $\mathbb{P}$ a collection of N training points, then *on average* we will get a value of $\mathscr{G}_{\text{gap}}(\Phi_{\text{ideal}})$ that is bounded as in (0.11).

The situation becomes more subtle if we move to the second contribute, $\mathscr{G}_{\text{gap}}(\Phi)$. In this case, in fact, the errors $\|\mathbf{u}_i - \Phi(\mathbf{v}_i)\|$ are most likely correlated one another, as Φ was built on the basis of the whole training set. This makes the previous argument invalid, forcing us to resort to something else. One way to overcome this difficulty is by exploiting the aforementioned test set, $T_N := \{(\mathbf{v}_i^{\text{test}}, \mathbf{u}_i^{\text{test}})\}_{i=1}^N$, which we here assume to have size N for simplicity. Then, we may define the *test gap*,

$$\mathscr{G}_{\text{test}} := \mathbb{E}_{T_N}\mathbb{E}_{S_N}|\mathscr{L}(\Phi) - E(\Phi)|$$

where we recall that $E(\Phi)$ is defined as in (0.6). With this setup one can prove that

$$\mathbb{E}_{S_N}|\mathscr{G}_{\text{gap}}(\Phi)| \leq \mathscr{G}_{\text{test}} + C\sqrt{\frac{1}{N}}$$

for some constant $C > 0$, as shown in [128]. Consequently, we may bound the average sample error as

$$\mathbb{E}_{S_N}[\mathscr{S}_{\text{error}}] \leq \mathscr{G}_{\text{test}} + C'\sqrt{\frac{1}{N}}.$$

This shows that, if the performances over the training and the test sets are comparable, then the expected sample error decays at a rate of $N^{-1/2}$ as a function of the training size.

The perspective of Approximation theory

We conclude this Section with a discussion on the approximation error. As the majority of the literature focuses on the scalar case, $n = 1$, we shall do the same.

The first question is whether the approximation error can be made arbitrarily small. Under suitable assumptions, the answer is positive. At this regard, we report below two historical results by Cybenko and Pinkus, which show that DNNs are dense in the space of continuous maps over compact domains. For the interested reader we refer to [50] and [159], respectively.

Theorem 0.1. (Cybenko, 1989). *Let $f : \Omega \subset \mathbb{R}^m \to \mathbb{R}$ be a continuous map defined over a compact domain. Let ρ be the sigmoidal activation. Then, for every $\varepsilon > 0$, there exists a ρ-DNN Φ such that* $\sup_{\mathbf{v}\in\Omega} |f(\mathbf{v}) - \Phi(\mathbf{v})| < \varepsilon$.

Theorem 0.2. (Pinkus, 1999). *Let $f : \Omega \subset \mathbb{R}^m \to \mathbb{R}$ be a continuous map defined over a compact domain. Let $\rho : \mathbb{R} \to \mathbb{R}$ be a continuous map that is not a polynomial. Then, for every $\varepsilon > 0$, there exists a ρ-DNN Φ such that* $\sup_{\mathbf{v}\in\Omega} |f(\mathbf{v}) - \Phi(\mathbf{v})| < \varepsilon$.

The above have many generalizations and alternative formulations, but they all fall under the common name of *universal approximation theorem.* Until recently, these were the main results available in the literature. In particular, there was no quantitative analysis of the approximation error: we knew that these DNNs existed but we had no clue about their architecture. The first breakthrough happened in 2017, with the innovative contribution of Dmitry Yarotski. We report below his main result, which can be found in [202].

Theorem 0.3. (Yarotski, 2017). *Let $f : [0,1]^m \to \mathbb{R}$ be $(s-1)$-times differentiable, and assume that all its derivatives of order $s-1$ are Lipschitz continuous. Then, for every $0 < \varepsilon < 1/2$, there exists a ReLU network Φ with at most*

i) $c \log(1/\varepsilon)$ *layers,*

ii) $c\varepsilon^{-m/s} \log(1/\varepsilon)$ *active weights (i.e. nonzero weight entries),*

such that

$$\sup_{\mathbf{v}\in[0,1]^m} |f(\mathbf{v}) - \Phi(\mathbf{v})| < \varepsilon.$$

Here, $c = c(m,s) > 0$ is a universal constant independent on f and ε.

Further generalizations also exist: see e.g. [78] for the case of integral and energy norms, or [79] for more general activation functions. Nevertheless, the key point is that we now have an intepretation for the design of network architectures. According to Theorem 0.3, the depth of a neural network scales logarithmically with the approximation error; conversely, the number of active weights depends on the wished accuracy and on the smoothness of the underlying ground truth.

The result by Yarotski is extremely powerful but also leaves some open questions. If the input dimension m increases dramatically, how should we proceed to avoid the curse of dimensionality in (ii)? Similarly, what happens if we consider vector-valued maps with very high-dimensional outputs? We postpone this discussion, which will be of crucial importance for us, to Chapter 6. There, we will provide a first answer to these questions and we will present innovative results that aim at expanding the state-of-the-art.

Part 1:

Machine Learning and genomics in precision medicine

1 | Radiotherapy and genomics

In this first Chapter we introduce the concept of radiosensitivity, that is the individual susceptibility to radiation, discussing its relation to genetic variants. To this end, we also provide a brief description of Single Nucleotide Polimorphisms and epistasis, two fundamental notions in genomics that will play an important role in the next Chapters. Finally, as an example, we describe the data concerning an actual clinical cohort.

In oncological healthcare, radiotherapy is a well-established approach that is commonly employed for the treatment of cancer patients. Fundamentally, it exploits ionizing radiation in the form of X-rays or neutron beams in order to kill tumor cells by breaking their DNA filaments. This can happen either directly, by depositing high levels of energy on DNA strands, or indirectly, by effect of free radicals resulting from the radiolysis of water [160]. In particular, radiotherapy is a destructive process, and, depending on the circumstances, it can produce short to long term complications. Short term by-products include, for instance, an increased permeability of the tissue near the irradiated region, which in turn may hinder the correct functioning of subsequent pharmacological treatments, such as chemotherapy. Long term side effects, however, are arguably the most worrying, as they can significantly impact patients lives for years. These aftereffects, which are commonly referred to as *late (radiotherapy) toxicity*, can remain silent for up to 5 years, and they can be very different depending on the situation. For instance, in the long term, prostate cancer patients may experience haematuria [99], i.e. presence of blood in the urine, while breast cancer patients may incurr in skin induration (fibrosis) [178].

Despite the massive improvements of oncological research and radiotherapy techniques, late toxicity remains a hot topic, as it still affects a relevant number of patients undergoing radiotherapy (up to 50% in some cases [98]). The current understanding of this phenomenon is that some patients, due to environmental and genetic factors, happen to be more sensitive to radiation than others. This individual susceptibility to radiation has been termed *radiosensitivity*. While the latter can be influenced by external factors, such as past surgeries [114], we have now evidence that radiosensitivity is a heritable human trait [199]. Inspired by these findings, several works have been conducted along this direction, e.g. [61, 99, 100], highlighting a deep connection between the manifestation of long term complications and the presence of certain genetic variants.

The purpose of these investigations is to go towards the implementation of personalized treatment routines that can prevent late toxicity. For instance, on the basis of their DNA, each patient could be assigned a (genetic) risk score that quantifies his/her susceptibility to ionizing radiation, before actually undergoing radiotherapy. If the resulting score is considered too high, then we may accommodate this fact by proposing lower doses of radiation to be distributed across a larger number of days. Conversely, if the risk score turns out to be sufficiently small, we may opt for a different strategy known as *hypofractionation*, where more doses are delivered in fewer days, thus allowing the patient to complete his/her treatment cycle faster. In general, the approach may be refined further through the integration of the genetic risk score into more sophisticated Normal Tissue Complication Probability (NTCP) models, where dosimetric and clinical variables are also included.

In radiation oncology, NTCP models are commonly employed to estimate the probability of late toxicity on the basis of previous knowledge about the patient. In the majority of the approaches, they are built by exploiting well-established techniques such as logistic regression. Thus, an example of an integrated NTCP model can be

$$\mathrm{logit}\mathbb{P}(Y = 1) = \gamma + \beta_0 \mathrm{PRS} + \beta_1 C_1 + \cdots + \beta_m C_m. \tag{1.1}$$

Here, logit is the inverse of the sigmoid activation,

$$\mathrm{logit}(q) := \log\left(\frac{q}{1-q}\right),$$

while Y is the binary variable representing the presence of long term complications. Thus, on the left-hand side of (1.1) we have a function of the probability of late toxicity, $\mathbb{P}(Y = 1)$, while on the right-hand side we have the affine predictor written in terms of the model covariates. The latter include a genetic term, the polygenic risk score PRS, and other variables of clinical/dosimetric nature, C_i, such as age, presence of comorbidities (e.g. diabetes) or radiation dose. The coefficients $\gamma, \beta_i \in \mathbb{R}$ are instead the model parameters, which are estimated from data.

Because of their simplicity and interpretability, these kind of NTCP models are usually highly appreciated by clinicians. The only practical challenge posed by this approach lies in the definition for the genetic term in (1.1). In fact, there are at least two fundamental bottlenecks that hinder the straightforward implementation of polygenic risk scores. One lies in the identification of useful genetic variants, as the human genome contains millions of them. The other resides in the complex interaction that genetic loci can entail with one another, a phenomenon known as *epistasis*. In Chapters 2 and 3 we will discuss on how to overcome these difficulties using Machine Learning. Before that, we devote the next two Sections to a more in depth discussion about genetic variants and their interaction. We also take the chance, in Section 1.3, to provide a detailed description about a clinical cohort consisting of prostate cancer patients.This latter population will serve as a case study later on, in Chapters 2 and 3 .

1.1 Single Nucleotide Polimorphisms (SNPs)

Single Nucleotide Polimorphisms, SNPs for short, are a way for encoding genetic mutations that occur at a specific position in the genome and that involve a single nucleotide. In general, the human genome consists of a polymer known as deoxyribonucleic acid (DNA), which comprises two specular polynucleotide chains. The latter are essentially a sequence of nucleotides, each composed by one of four nucleobasis, namely cytosine (C), guanine (G), adenine (A) or thymine (T). The majority of these sequences is the same across all human beings, as that is what ultimately defines our species in the biological sense. However, there are some locations at which different individuals may present different nucleotides. For example, at a given genetic loci, most individuals may present a G nucleotide, but in a minority of individuals, that same position is occupied by an A. In this case we say that, at that specific location in the genome, there is a SNP. The two variants, G or A in this example, are said to be the major (or dominant) allele and the minor (or recessive) allele, respectively [139]. Of note, as the mutation may concern any of the two polynucleotide chains, each person has three possible configurations for each SNP:

- *Dominant homozygosis* (G-G), that is, when both DNA chains present the major allele at the given position.
- *Recessive homozygosis* (A-A), as above but with the minor allele appearing in both chains.
- *Heterozygosis* (G-A or A-G), i.e. when the two DNA helix disagree, as they each present one of the two variants.

Since the three situations only differ by the number of minor alleles, a common approach is to represent SNPs configurations with numeric labels: 0 for dominant homozygosis, 1 for heterozygosis and 2 for recessive homozygosis.

At this point, it is worth to make a practical remark. Despite what we just said, it is actually very common to spot noninteger values in SNPs datasets. This is due to the so-called *imputation*, an established statistical technique that is employed for inferring unobserved genotypes. In fact, due to the very large number of SNPs in the human genome, $\sim 10^6$, it can be very expensive to obtain a complete DNA sequencing of a single individual. Thus, a common strategy is to measure only a limited portion of the genome, and then infer the unobserved genotypes by exploiting either the correlation among genetic traits or the existence of haplotypes (that is, group of alleles that are inherited together from a single parent [49]). While this clearly helps reducing the costs of DNA sequencing, it comes with the drawback of assigning continuous values to imputed SNPs. Consequently, for those applications where integer entries are imperative, imputed values have to be converted somehow, e.g. by rounding to the closest integer.

In general, however, SNPs values do not provide an exhaustive overview of the genetic information, as they only signal the presence of mutations without specifying their expression rates. Not only, it is a matter of fact that SNPs are mostly located in noncoding regions of the DNA, impyling that they mostly affect the patient indirectly [173, 194]. Still, despite all these considerations, SNPs have turned out to be a powerful tool for predicting the insurgence of certain diseases, such as Alzheimer [59], breast and ovarian cancer [132], diabetes [194] and many others, see e.g. [94]. In particular, as we anticipated, it has been shown that SNPs play an important role in determining the susceptibility of patients to radiotherapy [199].

1.2 The epistatic effect

Epistasis, literally "resting upon", is a phenomenon in which a certain phenotype is determined by the interaction of multiple genetic loci. The term was first coined by Bateson [14], in 1909, in an attempt to describe a form of interaction in which a gene was able to mask the effects of another gene. A common example is that of the shepard's purse, a ruderal plant that can either produce triangular or oval seed capsules. As we now understand it, the shape of the capsules is determined by two SNPs, say S_1 and S_2. When crossing doubly heterozygous individuals, that is plants with $\{S_1 = 1,\ S_2 = 1\}$, researchers have found 15:1 Mendelian ratios, meaning that only 1 out of 16 new born plants would present oval capsules. The biological explanation is that the two genes can mask each other whenever exhibiting a dominant allele; it is only when $\{S_1 = S_2 = 2\}$ that the plant will produce oval seeds (recessive-by-recessive interaction).

Over the years, scientists have found evidence of gene-gene interactions in many different biological contexts, including human healthcare, to the point that epistasis is now considered ubiquitous in determining the susceptibility to common human diseases [140]. Epistatic effects are also found in relation to radiosensitivity, a mechanism that was first observed in simple organisms, such as plants [22] or yeast [106], and later in human beings [11].

In light of this, it is clear that any predictive model of radiosensitivity should take epistasis into consideration. This calls for novel methodologies and approaches that can handle the intrinsic complexity of genes interactions. To this end, several researchers (including ourselves) are now finding a valuable help in the literature of Machine Learning, see e.g. [10, 95, 135].

1.3 Clinical case study for the next Chapters

In this Section we present a dataset of primary importance for Chapters 2 and 3, which consists of a population of male patients diagnosed with prostate cancer and later treated with external beam radiotherapy (no brachytherapy). The cohort counts 1'681 prostate cancer patients recruited across the hospitals of seven European countries (Belgium, France, Germany, Italy, the Netherlands, Spain, UK) plus the United States. Prior to radiotherapy, between April 2014 and October 2016, the patients enrolled as a part of the REQUITE project, where they agreed to participate in a clinical follow-up of at least 2 years, with the purpose of monitoring the occurrence of long term complications. Before the start of the radiotherapy treatment, the patients also donated two blood samples from which genomic and transcriptomic data were collected. In particular, the genotype data was generated under the format of SNPs encoding, with imputation being applied to most genetic loci. During the treatment, patients where prescribed with either conventionally fractionated or hypofractionated radiotherapy depending on the local standard-of-care regimens.

Toxicity endpoints were defined either by health professionals (CTCAE scoring) or through the use of individual surveys (Patient Reported Outcome). For our studies, we have been focusing on the following toxicities.

- *Rectal bleeding* (CTCAE scoring),
  Presence of blood in the rectum associated to anorectal injuries.
- *Urinary frequency variation* (CTCAE scoring),
  Significant change in the urinary frequency with severe limitation of every day activities.
- *Haematuria* (CTCAE scoring),
  Anomalous presence of blood in the urine (either visible by microscope or by eye).
- *Nocturia* (Patient Reported Outcome),
  Form of incontinence that leads patients to urinate more than 2-3 times per night.
- *Decreased stream flow* (Patient Reported Outcome),
  Hesitant or dripping stream while urinating.

Patients were scored with a degree of severity for each endpoint separately (grade 0 = no toxicity, grade 1 = mild toxicity, grade 2 and 3 = severe toxicity), before the treatment and once a year during the follow-up. This was to ensure a proper definition of *late* toxicity and avoid confounding factors. More precisely, the long term endpoints were defined as follows. Let Z denote any of the toxicities above. For $k \in \mathbb{N}$, let $Z_k \in \{0, 1, 2, 3\}$ be the degree of severity k years after the treatment, where Z_0 denotes the value at baseline (i.e. before radiotherapy). Given a minimum grade value $z \in \{1, 2, 3\}$, we define the late toxicity endpoint at year k as

$$Y_k := \begin{cases} 1 & Z_k \geq z \text{ and } Z_k > Z_0 \\ 0 & \text{otherwise.} \end{cases} \tag{1.2}$$

In other words: for a given endpoint, and a given follow-up time, we say that the patient experienced late toxicity if the reported symptoms were above a given threshold ($Z_k \geq z$) and the situation was actually worse than the one before treatment ($Z_k > Z_0$). We report in Table 1.1, the minimum grades that we adopted for our studies in Chapters 2 and 3. There, we focused on the occurrence of late toxicity within the first two years of follow-up, i.e. on the outcome

$$Y := \max\{Y_1,\, Y_2\}. \tag{1.3}$$

Finally, to further avoid confounding factors, we excluded those patients with an intrinsically known higher risk of toxicity. In particular, we did not include patients who had systemic lupus erythematosus, rheumatoid arthritis and other collagen vascular diseases. Additionally, for the analysis of the rectal bleeding endpoint, we also excluded those patients with hemorrhoids before radiotherapy. Conversely, for the urinary endpoints, we neglected those that underwent transurethral resection of the bladder or were using anti-muscarinic drugs.

Concerning the genomic data, after an in depth review of the existing literature, we decided to restrict our attention to a subset of 43 SNPs previously identified as associated to radiotoxicity and prostate cancer. We report them in Table 1.2.

For additional details about the cohort and the REQUITE project we refer to [179].

Endpoint	**Min. grade**	**Patients**	**With toxicity**
Rectal bleeding	1	1366	160 (11.7%)
Urinary frequency variation	2	1334	56 (4.2%)
Haematuria	1	1343	74 (5.5%)
Nocturia	2	1250	223 (17.8%)
Decreased stream flow	1	1234	211 (17.1%)

Table 1.1: Definition of the long term endpoints for 1-2 years follow up, cf. Equation (1.3). Min. grade = minimum degree of severity, cf. Equation (1.2). The total number of patients can vary from endpoint to endpoint due to missing values or additional exclusion criteria.

SNP	**Alleles**	**AIG**
rs147596965	G - T	0.06
rs2842169	T - C	0.03
rs11219068	G - A	0.13
rs708498	A - G	0.01
rs4906759	C - T	0.01
rs4775602	A - C	0.01
rs79604958	G - C	0.03
rs12591436	G - T	0.15
rs8075565	C - T	0.11
rs62091368	G - A	0.12
rs673783	T - G	0.18
rs8098701	C - T	0.01
rs141799618	C - G	0.03
rs76273496	T - C	0.01
rs7366282	A - C	0.01
rs11122573	C - T	0.00
rs6003982	C - G	0.25
rs10497203	A - C	0.00
rs7582141	G - T	0.00
rs6432512	C - T	0.00
rs264651	A - G	0.01
rs264588	C - A	0.01

SNP	**Alleles**	**AIG**
rs264631	C - G	0.01
rs77530448	A - G	0.04
rs1045485	G - C	0.00
rs10209697	G - A	0.01
rs144596911	G - A	0.01
rs71610881	G - A	0.02
rs342442	C - T	0.03
rs6535028	T - C	0.00
rs10519410	A - G	0.00
rs7720298	C - G	0.03
rs141342719	C - T	0.01
rs17599026	C - T	0.03
rs17055178	A - G	0.01
rs7356945	C - T	0.06
rs4997823	G - A	0.10
rs845552	A - G	0.01
rs1799983	T - G	0.10
rs7829759	A - C	0.02
rs17362923	C - G	0.00
rs10101158	T - A	0.04
rs10969913	A - G	0.01

Table 1.2: SNPs considered for our studies concerning radiosensitivity in prostate cancer patients. Dominant alleles are reported first, recessive alleles last. AIG = Average Imputation Gap, that is, the average difference (in absolute value) between the reported SNP value and its closest integer; values close to 0.5 indicate a high level of imputation.

2 | Feature selection using Deep Learning

We devote this second Chapter to the presentation of a Deep Learning algorithm, originally developed by Massi et al. in [134], that we employ for the identification of genetic variants associated to radiotoxicity. The whole idea is grounded on the use of an autoencoder architecture: an extremely powerful tool that, in a completely different context, will also play an important role in Chapter 5.

The identification of useful genetic loci is a fundamental prerequisite for the development of predictive genomic models. A first answer to this problem is provided by Genome-Wide Association Studies (GWAS), a data-driven approach where the entire genome is scanned for statistically significant associations with the outcome of interest. In GWAS, researchers assess the significance of each SNP individually by using classical methods of statistical inference, such as the χ^2 test for independence, see [44]. Only those SNPs that obtain a sufficiently small p-value in the χ^2 test are selected, while the others get discarded. Depending on the application, the p-value threshold may vary from 10^{-5} to 10^{-8}. Researchers also employ suitable strategies to account for multiple hypothesis testing, such as Bonferroni and Sidak corrections, or permutation procedures [44]. Nevertheless, GWAS only provide a first insight on the selection of useful genetic loci. In fact, they often identify a large number associations, leaving researchers with hundreds or thousands of SNPs to work with. Not to mention that the methods employed in GWAS do not account for epistatic effects. Therefore, other approaches are needed to further refine this selection. Some of these are based on greedy strategies, such as Maximum-Relevance-Minimum-Redundancy algorithms [122], while others exploit the intrinsic selection performed by certain classifiers, such as LASSO logistic regression models [7]. In the following Sections, after a brief introduction about autoencoders, we describe a novel approach based on deep neural networks that has the benefit of accounting for epistatic effects. The latter was first proposed by Massi et al. in [134, 135], to which I partially contributed at beginning of my PhD.

2.1 Autoencoders in a nutshell

An autoencoder is a specific type of neural network architecture that is commonly employed for data compression (cf. Figure 2.1). We report a rigorous definition below.

Definition 2.1. *Let $q, n \in \mathbb{N}$ with $q > n$. Let $\Psi' : \mathbb{R}^q \to \mathbb{R}^n$ and $\Psi : \mathbb{R}^n \to \mathbb{R}^q$ be two neural networks, such that all the hidden layers of both Ψ and Ψ' have at least n neurons. Then, the map*

$$\Xi := \Psi \circ \Psi'$$

is said to be an autoencoder of latent dimension n, while the networks Ψ' and Ψ are respectively known under the names of encoder and decoder.

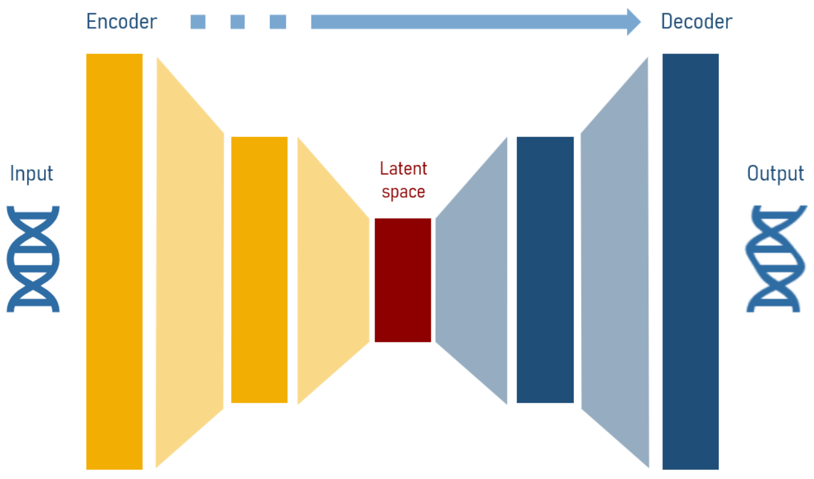


Figure 2.1: Visual representation of an autoencoder architecture.

Given a training set $\{\mathbf{x}_i\}_{i=1,\dots,N} \subset \mathbb{R}^q$ the autoencoder $\Xi = \Psi \circ \psi'$ is usually trained according to the loss function below,

$$\mathscr{L}(\Xi) = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{x}_i - \Xi(\mathbf{x}_i)\|^2 \tag{2.1}$$

where $\|\cdot\|$ is the Euclidean norm. In this way, the autoencoder is pushed to learn a latent representation of the input variable. In fact, after the training, the encoder Ψ' naturally yields a synthetic representation of the input $\mathbf{z} = \Psi'(\mathbf{x}) \in \mathbb{R}^n$, where only n latent features are used instead of the initial q components. The fact that such representation is meaningful is automatically granted by the fact that the autoencoder has been trained to minimize the loss of information, i.e. such that $\mathbf{x} \approx \Psi(\mathbf{z})$. In this sense, an autoencoder can be seen as a nonlinear alternative to linear compression methods, such as the well-known Principal Component Analysis (PCA). Indeed, when the concept first appeared in the literature, several authors spoke about autoencoders as a form of nonlinear PCA performed through neural networks, see e.g. [110].

In some cases it is useful to provide the latent space with additional structure. To this end, several alternatives to classical autoencoders have been proposed, such as variational, denoising and deep sparse autoencoders. For our purposes, we are interested in the latter class of models. A deep sparse autoencoder is a particular type of autoencoder in which the latent representations are given by sparse vectors. In practice, this is achieved by modifying the loss function in (2.1) to

$$\mathscr{L}(\Psi, \Psi') = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{x}_i - (\Psi \circ \Psi')(\mathbf{x}_i)\|^2 + \frac{\lambda}{N}\sum_{i=1}^{N} |\Psi'(\mathbf{x}_i)| \tag{2.2}$$

where $\lambda > 0$ is some penalization parameter, while $|\cdot|$ is the ℓ^1-norm. Once again, this is similar in spirit to sparse PCA. The idea in this case is that, instead of considering extremely small values of the latent dimension n, we may allow for larger latent spaces but recover a similar compression of information by requiring the additional sparsity constraint. If we are more interested about the autoencoder learning a suitable representation than having a small latent dimension, this can be a valid alternative. Furthermore, it has the practical advantage of leading to architectures that are easier to identify, in terms of tuning of the hyperparameters.

Aside from dimensionality reduction, autoencoders can also be used for other purposes, such as anomaly and outlier detection, see e.g. [17, 39]. In that case, the idea is to train

Ξ by only considering a subsample of conventional observations (to be defined in a suitable sense), in such a way that the autoencoder can develop a latent representation of *normality* (in the sense of ordinariness). Then, whenever a new input datum $\mathbf{x} \in \mathbb{R}^q$ is found with a high reconstruction error, $\|\mathbf{x} - \Xi(\mathbf{x})\|$, it is marked as an outlier. Up to some adjustments, this is the key idea underlying the Deep Sparse AutoEncoder Ensamble algorithm, which we shall describe in the next Section. While the approach is completely general, for the sake of simplicity, we shall present it limitedly to the case of SNPs identification.

2.2 The Deep Sparse AutoEncoder Ensamble algorithm (DSAEE)

We now return to our original problem of selecting genetic variants. Mathematically speaking, the identification of relevant SNPs is a problem of feature selection. To set some notation, let $S_1, \dots, S_q$ be the SNPs under investigation, and let Y be the binary outcome, such as, for instance, the presence/absence of late toxicity after radiotherapy. We assume to have at our disposal a corresponding dataset $\{s_1^{(i)}, \dots, s_q^{(i)}, y^{(i)}\}_{i=1}^N$ consisting of independent and identically distributed observations. In general, if the analysis is conducted on the basis of previous GWAS, $q \propto 10^2, 10^3$, otherwise the order of magnitude is $q \propto 10^6$.

At this stage, our main goal is to exploit the available data in order to identify a restricted pool $\{S_{j_1}, \dots, S_{j_p}\} \subset \{S_j\}_{j=1}^q$ of SNPs that are predictive for Y. As we mentioned previously, this is nothing but a problem of feature selection in a binary classification setting. Still, it is a rather challenging task because of the following.

i) The large number of features q, which sometimes is even higher than the number of observations N.

ii) The presence of epistasis, which may cover up the effects of certain mutations if high-order interactions are not considered.

iii) The class imbalance, i.e. the disproportion between the two groups of patients, respectively $\mathcal{Z} := \{j \mid y^{(j)} = 0\}$ and $\mathcal{O} := \{j \mid y^{(j)} = 1\}$. This scenario is typically encountered when dealing with rare outcomes or diseases, in which case one has $|\mathcal{O}| \gg |\mathcal{Z}|$.

Despite the vast literature on feature selection, addressing all the above issues in a computationally feasible way remains a challenging task. Our proposal is to accommodate these problems by employing the Deep Learning approach developed by Massi et al. in [134].

As we anticipated, the whole idea revolves around the use of autoencoders. To start, we fix an ensamble dimension $B \in \mathbb{N}$, $B > 1$. For each $b \in \{1, \dots, B\}$, we select randomly a subset $\mathcal{Z}_b \subset \mathcal{Z}$ of patients without toxicity, where $|\mathcal{Z}_b| = |\mathcal{O}|$. This selection is used to define a test set of equally balanced observations, and a training set that only consists of patients without toxicity,

$$\mathbf{X}^b_{\text{test}} := \{\mathbf{x}_i\}_{i \in \mathcal{Z}_b \cup \mathcal{O}}, \qquad \mathbf{X}^b_{\text{train}} := \{\mathbf{x}_i\}_{i \in \mathcal{Z} \setminus \mathcal{Z}_b},$$

where $\mathbf{x}_i := [s_1^{(i)}, \dots, s_q^{(i)}] \in \mathbb{R}^q$ is the vector of SNPs values. Then, in analogy to anomaly detection algorithms, for each $b \in \{1, \dots, B\}$, we train an autoencoder $\Xi_b : \mathbb{R}^q \to \mathbb{R}^q$ such that $\mathbf{x} \approx \Xi_b(\mathbf{x})$ for all $\mathbf{x} \in \mathbf{X}^b_{\text{train}}$. To this end, the authors in [134] propose the use of deep sparse autoencoders, as the introduction of the sparsity penalization appeared to improve their results.

After the training, each autoencoder is tested against its own test set, $\mathbf{X}^b_{\text{test}}$, where it is expected to commit higher errors. To quantify this difference for each of the q SNPs, we compute the following error gaps

$$\Delta_b^{(j)} := \frac{1}{|\mathcal{O}|} \sum_{i \in \mathcal{O}} |\mathbf{x}_i^{(j)} - \Xi_b^{(j)}(\mathbf{x}_i)| - \frac{1}{|\mathcal{Z}_b|} \sum_{i \in \mathcal{Z}_b} |\mathbf{x}_i^{(j)} - \Xi_b^{(j)}(\mathbf{x}_i)|$$

where $j \in \{1,\dots,q\}$, $\mathbf{x}_i^{(j)} = s_i^{(j)}$ is the value of the jth SNP as observed in patient i, while $\Xi_b^{(j)}(\mathbf{x}_i)$ is the jth component of $\Xi_b(\mathbf{x}_i) \in \mathbb{R}^q$. The results are then averaged across all ensambles, yielding a unique gap for each SNP S_j,

$$\Delta^{(j)} := \frac{1}{B}\sum_{b=1}^{B}\Delta_b^{(j)}.$$

Now, the idea is that the SNPs with a higher $\Delta^{(j)}$ are the most useful for stratifying patients. In fact, those SNPs are the ones that contribute the most in increasing the reconstruction error over the minority class. In particular, if we consider the occurrence of late toxicity as an anomaly, those SNPs are the ones that facilitate the identification of outliers, i.e. patients with toxicity. Of note, treating late toxicity as a rare event is often a reasonable assumption, especially if we restrict to severe complications, which are usually experienced by 1 out of 10 patients.

From an operational point of view, we can think of $\Delta^{(j)}$ as to a measure of importance, which allows us to rank the q SNPs in the following natural way,

$$S_{j_1},\dots,S_{j_q}$$

with $\Delta^{(j_k)} \geq \Delta^{(j_{k+1})}$ for all $k \in \{1,\dots,q-1\}$. In particular, to draw a restricted pool of $p \ll q$ SNPs that are predictive for Y, we may simply select the first p in the sorted list, namely $\{S_{j_1},\dots,S_{j_p}\}$.

The main strenghts of the DSAEE approach are: (i) its ability in handling the class imbalance, which comes from the exploitation of anomaly detection routines; (ii) the fact that it can account for possible interaction effects, thanks to the nonlinear capabilities of autoencoders, (iii) its scalability, as autoencoders can notoriously handle inputs in high dimensional spaces. However, it has some limitations as well. For instance, it is blind to feature correlation: in the extreme case of two identical SNPs that are both predictive for Y, it will likely select both of them. For the interested reader, we leave [134] as a reference, where the authors analyze the DSAEE algorithm in full detail, providing also suitable comparisons with the state-of-the-art.

2.3 Clinical application

We now report the results obtained by applying the DSAEE approach to the clinical cohort presented in Chapter 1, Section 1.3. We shall not comment the performance of the algorithm or provide comparisons with other approaches: as I did not contribute directly to the development of this methodology, we only consider this application as a preliminary step for the actual construction of the polygenic risk score, which is carried out in Chapter 3, Section 3.5.

We recall that the aforementioned cohort concerns a population of prostate cancer patients, in which five toxicity endpoints were monitored. We applied the DSAEE algorithm to all of the five outcomes available, considering a 1-2 years follow-up time frame for their definition (cf. Equation (1.3) in Section 1.3).

To foster the comparability of the results, we chose a single autoencoder architecture for all the experiments, which we have reported in Table 2.1. In particular, we employed deep sparse autoencoders with a latent dimension of $n = 20$ and a sigmoidal activation at the most inner layer to favor sparse representations. During training, the penalization term was set to $\lambda = 10^{-5}$, see Equation (2.2). The ensamble dimension was set to $B = 50$, and the autoencoders were trained for 400 epochs using the Adam optimizer with default settings (learning rate equal to 10^{-3}). After running the ensamble algorithm, for each endpoint separately, we picked the top 13 SNPs in the DSAEE ranking (equivalently, the top 30% out of the initial 43). We report in Figure 2.2 the SNPs selected and their rankings for each endpoint. Of note, it often happened that SNPs previously reported as associated

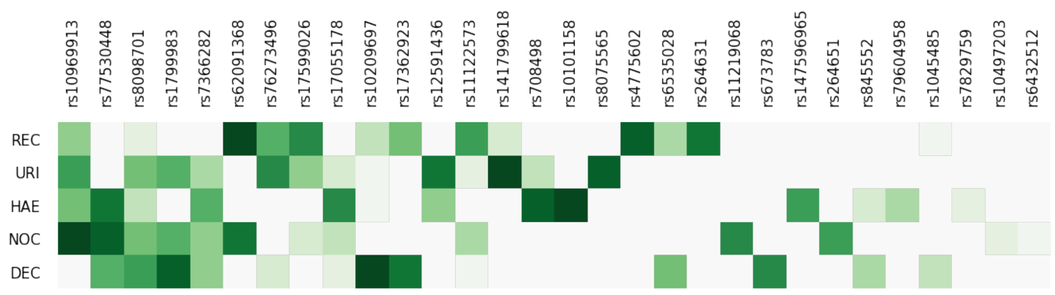


Figure 2.2: Visual representation of the SNPs selected by the DSAEE algorithm for the prostate cancer cohort, Section 2.3. Darker colors correspond to higher rankings. For each endpoint, only the rankings of the top 13 SNPs are reported. REC = Rectal bleeding, URI = Urinary frequency variation, HAE = Haematuria, NOC = Nocturia, DEC = Decreased stream flow.

	Layer	**Input**	**Output**	**Activation**	**dofs**
Encoder	Dense	43	40	Hyperbolic tangent	1760
	Dense	40	30	Hyperbolic tangent	1230
	Dense	30	20	Sigmoid	620
Decoder	Dense	20	30	Hyperbolic tangent	630
	Dense	30	40	Hyperbolic tangent	1240
	Dense	40	43	None	1763

Table 2.1: Autoencoder architecture. dofs = degrees of freedom, i.e., the total number of entries in the weight matrix and in the bias vector of a given layer.

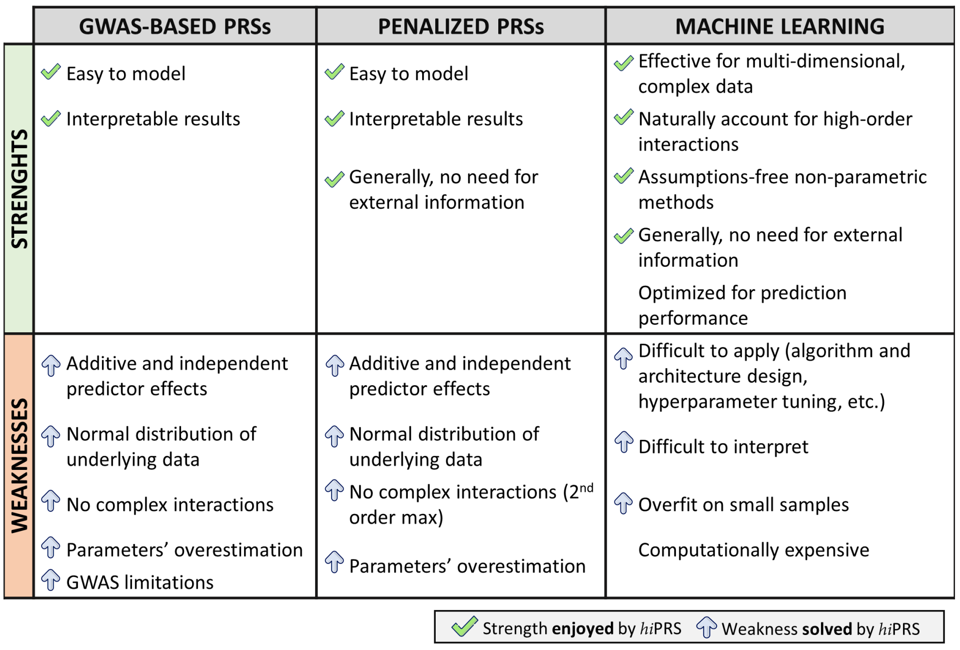
to a given outcome were selected for a different endpoint. For instance, SNP $rs10969913$ appeared highly relevant for the nocturia endpoint, even though it was originally discovered as associated to decreased urinary stream. In general, we can appreciate from Figure 2.2 that several SNPs were simultaneously ranked as highly important for different toxicities. In particular, coherently to what we would expect, urinary endpoints tend to have more SNPs in common compared to rectal bleeding. Overall, 13 of the initial 43 SNPs were discarded as they were not selected for any of the five endpoints. For an in depth discussion about all these findings we refer to the clinical paper [135], where we have published all our results in full detail.

3 | Interaction-aware Polygenic Risk Scores

*In this third Chapter, we present a novel approach to polygenic risk scoring that is able to handle class imbalance while also accounting for high order interactions and preserving model interpretability. I developed this methodology, that we now call hi*PRS*, during my PhD, with the collaboration of Dr. Massi M.C. The approach was applied successfully in the context of biomarker discovery for radiosensitivity in prostate cancer patients, see Franco et al. "Development of a method for generating SNP interaction-aware polygenic risk scores for radiotherapy toxicity", Radiotherapy & Oncology [68], and later generalized in a subsequent work, see Massi et al., "Learning High-Order Interactions for Polygenic Risk Prediction", PLOS One [133]. Here, we shall proceed in reverse: first, we present the idea and assess its properties on simulated data; then, we move to the clinical applications, and discuss how the proposed approach can be integrated in NTCPs to foster their predictive performance.*

In computational biology, the development of predictive genetic biomarkers is a notoriously challenging task that brings together complex high dimensional data and statistical inference. Still, it is also an incredibly active area of research because of its massive implications in healthcare and precision medicine. Here, we shall focus on a specific class of genetic biomarkers that are commonly known under the name of Polygenic Risk Scores (PRSs). The latter are risk indicators that are derived from SNPs values, and whose purpose is to estimate the effect of multiple genetic variants on a given phenotype [192]. However, as we shall discuss throughout the Chapter, classical approaches to polygenic risk scoring have several limitations: for instance, they rely on the assumption that SNPs effects over the phenotype are purely additive, which is often not true [148]. All these drawbacks may hinder the development of useful polygenic risk scores and, in the case of radiotherapy treatments, preclude the improvement of NTCP models. In light of this, we propose a novel algorithm, termed *hi*PRS, for the construction of interaction-aware biomarkers. The latter, coupled with a suitable feature selection strategy, such as the DSAEE, results in a comprehensive approach to polygenic risk scoring that combines together predictive power and clinical interpretability.

We have structured the rest of the Chapter as follows. First, in Section 3.1, we provide some additional context and briefly synthesize the state-of-the-art. Then, in Section 3.2, we detail the methodology underlying the *hi*PRS approach, whose capabilities are assessed in Section 3.3. There, within a controlled environment, we perform several simulation studies, analyzing the behavior of *hi*PRS under different clinically relevant conditions (e.g., varying sample size or noise in the data) and providing a suitable comparison with different benchmark algorithms. Finally, in Section 3.5, we apply the methodology to the clinical case studies first introduced in Chapter 1 and partially analyzed in Section 2.3.



	GWAS-BASED PRSs	PENALIZED PRSs	MACHINE LEARNING
STRENGHTS	✓ Easy to model ✓ Interpretable results	✓ Easy to model ✓ Interpretable results ✓ Generally, no need for external information	✓ Effective for multi-dimensional, complex data ✓ Naturally account for high-order interactions ✓ Assumptions-free non-parametric methods ✓ Generally, no need for external information Optimized for prediction performance
WEAKNESSES	⇧ Additive and independent predictor effects ⇧ Normal distribution of underlying data ⇧ No complex interactions ⇧ Parameters' overestimation ⇧ GWAS limitations	⇧ Additive and independent predictor effects ⇧ Normal distribution of underlying data ⇧ No complex interactions (2nd order max) ⇧ Parameters' overestimation	⇧ Difficult to apply (algorithm and architecture design, hyperparameter tuning, etc.) ⇧ Difficult to interpret ⇧ Overfit on small samples Computationally expensive

✓ Strength **enjoyed** by *hi*PRS ⇧ Weakness **solved** by *hi*PRS

Figure 3.1: Strengths and weaknesses of three main PRS categories (cf. 3.1). Green tick = strength that is also enjoyed by *hi*PRS. Blue arrow = weakness of which *hi*PRS does not suffer.

3.1 Literature review

PRSs are one of the most traditional approaches to model genetic risk. They exploit a fixed model approach to sum the contribution of multiple risk alleles over a specific complex disease [38]. Calculating PRSs is a common practice because of their simplicity, computational efficiency, and straightforward interpretability. Indeed, while polygenic scores are used to predict phenotypes, there are other interests beyond forecasting. For instance, model interpretability is often key for research purposes such as the discovery or validation of SNPs associations. The scheme in Figure 3.1 gives an overview on the strenghts and weaknesses of the most common PRS approaches in the literature.

Standard weighted PRS estimation relies on the summary statistics of Genome-Wide Association Study (GWAS), obtained on one or more discovery cohorts, modeling the independent effect of individual SNPs on the outcome. These PRSs (cf. Figure 3.1, first column, GWAS-based PRSs) exploit SNP-specific odds ratios or effect sizes to weight the contribution of the risk alleles on the disease risk or outcome [37, 43]. The set of SNPs to be included in this estimation may of course affect the predictive power of the PRS significantly. Some approaches include all SNPs, with the risk of incorporating useless or redundant information, while others retain a subset of SNPs based on predefined criteria (e.g., those passing an arbitrary p-value threshold in the GWAS results [37]).

Despite their wide adoption and appreciated simplicity and interpretability, the performance and reliability of traditional PRSs have been largely discussed and several methodological concerns have been raised (see, e.g., [92] and references therein) as the approach presents some evident limitations.

In particular, (i) the GWAS studies exploited for weights estimation are oftentimes underpowered by multiple independent testing and their effect sizes overestimated because of winners' curse and biases [181, 19, 185, 144]. Additionally, (ii) GWAS effect weights are traditionally computed considering the effect of single SNPs on the phenotype. By doing so, they do not account for the complex and nonlinear interactions between alleles in the genotype and their role in determining the phenotype, or the epistasis effect [1, 38, 120, 121, 148, 151]. Essentially, as we discussed back in Chapter 1, epistasis translates into the departure from independence (or additivity, from a statistical point of view) of the effects of multiple loci in the way that they combine to cause disease [48]. In other words, interaction effects exist between loci, and their presence was reported to make major contributions to phenotypes [85, 120, 121, 141, 151, 183, 195]. However, including SNP-SNP interactions in GWAS and risk scoring models is computationally challenging due to the high dimensions involved: in fact, the number of possible interactions grows exponentially with the number of SNPs considered. Additionally, this also increases dramatically the amount of independent tests to be carried out, thus strongly affecting their reliability. The authors in [119] attempted the inclusion of SNP-SNP interactions by filtering those that where relevant accordingly to some imposed biological criteria, but only managed to consider up to second-order interactions. Other methodological concerns of GWAS-based approaches are the fact that (iii) GWAS weights ignore the mediating role of clinical covariates when estimating SNPs effect on complex diseases [92], and that (iv) these models incorporate strict assumptions, e.g., they include additive and independent predictor effects, and assume that observations are uncorrelated [1, 38, 197]. These assumptions do not necessarily hold true when modeling complex polygenic diseases. For instance, linkage disequilibrium, i.e., the non-random association of alleles at two or more loci in a population [38], statistically translates into strong correlation between predictors. To account for linkage disequilibrium, some methods were developed to optimize SNPs *reweighting* (LD pruning and p-value thresholding, LDpred [193], and others).

Other well-recognized solutions to polygenic risk scoring discard GWAS weights and directly take genotype data as input, including various forms of penalization to restrict the pool of predictors (cf. Figure 3.1, Penalized PRSs). These include genomic BLUP [81], shrinkage methods (e.g. LASSO) or generalized linear mixed models. However, all these methods share with the most traditional PRSs the assumption of additive effect of single SNPs on the outcome, and may incur in the curse of dimensionality when trying to include all potential interaction terms, with the risk of overestimating effect sizes and obtaining unreliable models. This is particularly true if modeling high-order interactions. Nevertheless, these complex interactions were found useful in describing genotype-phenotype relationships for complex traits and common diseases both in humans and model organisms [46, 68, 77, 167, 190], highlighting the need for novel approaches able to account for high-order interactions.

In this respect, great attention has been recently devoted to polygenic risk prediction via Machine Learning algorithms, (Figure 3.1, right-most column). These algorithms employ multivariate, non-parametric methods that robustly recognize patterns from non-normally distributed and strongly correlated data [151, 183, 197]. Moreover, these methods are naturally capable of modeling highly interactive complex data structures, making them powerful tools for complex disease prediction [151, 183, 197]. However, most Machine Learning algorithms, such as random forests, support vector machines or neural networks, demonstrate great predictive power but lack in model interpretability, leaving researchers with no information on the role and structure of the complex interactions influencing the phenotype prediction. Moreover, these algorithms require a lot of training samples to avoid overfitting, but the available cohorts in real world research settings are oftentimes quite small.

In this Chapter, we aim at addressing polygenic risk prediction by proposing a novel PRS approach, called High-order Interactions-aware Polygenic Risk Score (*hi*PRS), whose most remarkable feature is the capability of robustly and reliably incorporating high-order interactions in modeling polygenic risk, while constructing a simple and interpretable model.

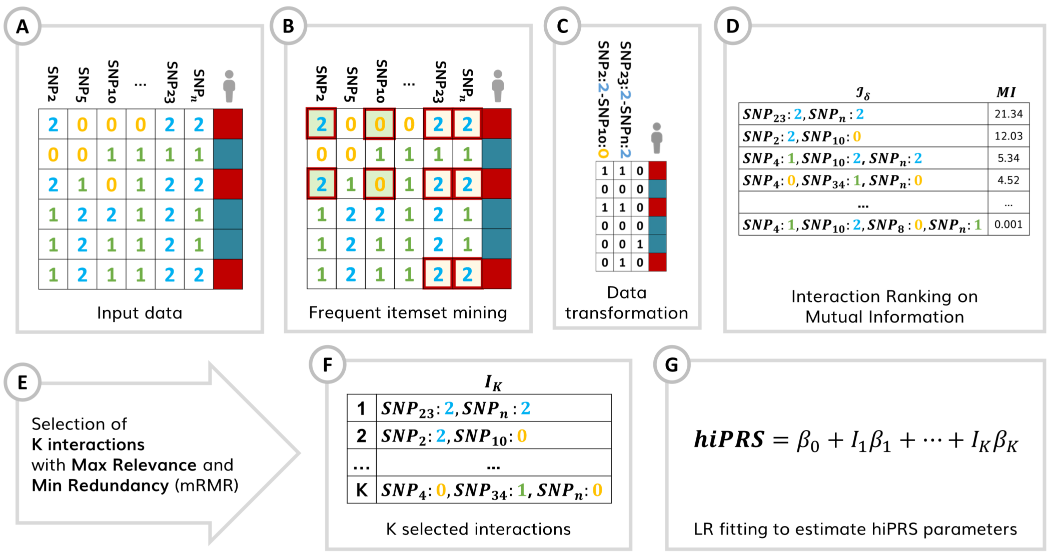

A
Input data
B
Frequent itemset mining
C
Data transformation
D
Interaction Ranking on Mutual Information
$\mathcal{I}_\delta$
MI
SNP23: 2, SNPn: 2 — 21.34
SNP2: 2, SNP10: 0 — 12.03
SNP4: 1, SNP10: 2, SNPn: 2 — 5.34
SNP4: 0, SNP34: 1, SNPn: 0 — 4.52
SNP4: 1, SNP10: 2, SNP8: 0, SNPn: 1 — 0.001
E
Selection of K interactions with Max Relevance and Min Redundancy (mRMR)
F
I_K
1 SNP23: 2, SNPn: 2
2 SNP2: 2, SNP10: 0
K SNP4: 0, SNP34: 1, SNPn: 0
K selected interactions
G
$hiPRS = \beta_0 + I_1\beta_1 + \dots + I_K\beta_K$
LR fitting to estimate hiPRS parameters

Figure 3.2: Workflow of the *hi*PRS algorithm. Starting from SNPs data (A), the algorithm exploits data mining routines to identify candidate interactions that appear with a sufficient frequency in the positive class (B). Then, a corresponding dataset of observations is computed, reporting the value of each candidate interaction across all patients (C). Interactions are then ranked on the basis of their association with the outcome, quantified by means of relevance measures such as mutual information (D). Then, a maximum relevance - minimum redundancy algorithm is applied to filter out the list of candidates, leaving the user with a smaller pool of K interaction terms (E-F). Finally, the latter are combined through a weighted sum into the final polygenic risk score, where the weights are estimated by fitting a logistic regression model.

3.2 The *hi*PRS algorithm

In order to describe our approach, we first introduce some notation. Let $S_1, \dots, S_p$ be p SNPs and let Y be a random variable denoting a binary outcome. We consider each SNP to be a categorical variable taking values in $\{0, 1, 2\}$, where labels read as in Chapter 1. In particular, we denote major allele homozygosis by 0, heterozygosis by 1, and minor allele homozygosis by 2. To model SNP-alleles interactions, we use products of indicator functions. For instance, $I := \mathbf{1}_{\{0\}}(S_1)\mathbf{1}_{\{1\}}(S_2)$ encodes the interaction between major allele homozigosis in S_1 and heterozygosis in S_2. Here, $\mathbf{1}_A$ denotes the indicator function of the set $A \subseteq \mathbb{R}$, which equals 1 for all $x \in A$ and 0 otherwise. Then, we may then define the collection of all SNP-allele interactions as

$$\mathcal{I} := \left\{ \prod_{j \in J} \mathbf{1}_{\{l_j\}}(S_j) \text{ such that } J \subset \{1, \dots, p\} \text{ and } l_j \in \{0, 1, 2\} \right\}.$$

We note that, with little abuse of notation, the set $\mathcal{I}$ also contains the dummy-variables associated to the SNP-alleles, as those are obtained when J is a singleton.

We are given a dataset $\{s_1^{(j)}, \dots, s_p^{(j)}, y^{(j)}\}_{j=1}^N$ consisting of N i.i.d. realizations of the SNPs and the outcome Y. Starting from these, we aim to construct a PRS of the form

$$hi\text{PRS} = \beta_0 + \beta_1 I_1 + \dots + \beta_K I_K \tag{3.1}$$

where $\{I_k\}_{k=1}^K \subset \mathcal{I}$. To this end, we propose a novel scoring method, *hi*PRS, where K is user-specified and a data-driven algorithm returns the list of interactions with the corresponding weights. We detail the whole idea, which we have depicted in Figure 3.2, below.

For any $I \in \mathcal{I}$, let $\{i^{(j)}\}_{j=1}^N$ be the corresponding observations in the dataset. We define the collection of all cases and its complement as

$$O := \{j \mid y^{(j)} = 1\}, \qquad Z := \{j \mid y^{(j)} = 0\}.$$

We make the assumption that $|O| \ll |Z|$, which is the typical scenario of a rare outcome. We refer to O and Z respectively as the minority and majority class. As a first step, we scan the data relative to the minority class, and we search for those interactions $\mathcal{I}_{\delta,l_{\max}}$ that appear with an empirical frequency above a given threshold $\delta > 0$, and have length at most $l_{\max}$. More precisely, we define

$$\mathcal{I}_{\delta,l_{\max}} := \left\{ I \in \mathcal{I} \text{ such that } \frac{1}{|O|}\sum_{j\in O} i^{(j)} > \delta \text{ and } \text{Length}(I) \le l_{\max} \right\}, \tag{3.2}$$

where the Length(I) is the number of SNPs involved in the definition of I. In principle, computing $\mathcal{I}_{\delta,l_{\max}}$ can be very demanding since $|\mathcal{I}| = 4^p$. However, this drawback is mitigated by two key factors. First of all, we note that each interaction uniquely corresponds to a *pattern of alleles*. For instance, let

$$I = \mathbf{1}_{\{1\}}(S_2)\mathbf{1}_{\{0\}}(S_3)\mathbf{1}_{\{1\}}(S_5) = \mathbf{1}_{\{1\}\times\{0\}\times\{1\}}(S_2, S_3, S_5).$$

Then $i^{(j)} = 1$ if and only if the pattern $\{S_2 = 1, S_3 = 0, S_5 = 1\}$ is observed in the jth patient. This duality between interactions and patterns allows us to reframe (3.2) in the context of frequent itemsets mining, where we can rely on a multitude of algorithms such as *Apriori* and *FP-Growth*. Additionally, the computational cost is alleviated by the fact that we limit our search to the minority class O.

The next step is to extract a suitable sublist $\{I_k\}_{k=1}^K \subset \mathcal{I}_{\delta,l_{\max}}$ to be used in (3.1). While this problem can be framed in the context of feature selection, finding an optimal solution can be very hard due to the large number of candidates. To overcome this drawback, we introduce a filtering technique based on the so-called Minimum Redundancy – Maximum Relevance approaches, mRMR for short. The idea goes as follows. First, we introduce a relevance measure based on the (empirical) mutual information, that is

$$\mathbb{I}(I,Y) := \sum_{\substack{j=1,\dots,N \\ a=0,1 \\ b=0,1}} \frac{1}{N}|\{i^{(j)} = a, \ y^{(j)} = b\}| \log\left(\frac{|\{i^{(j)} = a, \ y^{(j)} = b\}|/N}{|\{i^{(j)} = a\}|/N \cdot |\{y^{(j)} = b\}|/N} \right). \tag{3.3}$$

By definition, $\mathbb{I}(I,Y) \ge 0$ and larger values are obtained when I and Y are informative with respect to each other. Parallel to this, we introduce a redudancy measure $\mathbb{S} : \mathcal{I} \times \mathcal{I} \to \mathbb{N}$ that quantifies the similarity between two given interactions. More precisely, we define $\mathbb{S}(I,T)$ to be the number of common alleles within the two patterns. We then construct the set $\{I_k\}_{k=1}^K$ along the lines of mRMR methods, that is through the greedy optimization of the ratio between relevance and redundancy. We report a detailed pseudocode in Algorithm 1.

We remark that K is an hyperparameter chosen by the user. In particular, one may as well optimize its value according to some grid-search algorithm of choice. Indeed, the most computationally expensive parts in the *hi*PRS pipeline are the candidates search and the computation of relevance/redundancy measures. Once these steps have been carried out, multiple values of K can be tested and the user may choose the one considered to be optimal.

Once the set $\mathcal{I}_{\text{picked}} = \{I_k\}_{k=1}^K$ has been identified, we compute the weights $\beta_0, \dots, \beta_K$ by fitting the Logistic Regression model below

$$\text{logit}\mathbb{P}(Y = 1) = \beta_0 + \beta_1 I_1 + \dots + \beta_K I_K. \tag{3.4}$$

Finally, we define the *hi*PRS according to (3.1), meaning that the score is actually the (affine) linear predictor in (3.4).

Algorithm 3.1: Extraction of the *hi*PRS allele-patterns.

```
Input  : $K$, $\mathcal{I}_{\delta,l_{\max}}$ and $\{(i_1^{(j)},\dots,i_{|\mathcal{I}_{\delta,l_{\max}}|}^{(j)},y^{(j)})\}_{j=1}^N$.
Output: $\{I_k\}_{k=1}^K \subset \mathcal{I}_{\delta,l_{\max}}$.

/* Select most relevant interaction */
$I_1 \leftarrow \mathrm{argmax}_{I\in\mathcal{I}_{\delta,l_{\max}}} \mathbb{I}(I,Y)$
$\mathcal{I}_{\text{picked}} \leftarrow \{I_1\}$
$\mathcal{I}_{\text{left}} \leftarrow \mathcal{I}_{\delta,l_{\max}} \setminus \mathcal{I}_{\text{picked}}$

/* Add patterns iteratively */
while $|\mathcal{I}_{\text{picked}}| < K$ do
    for $I \in \mathcal{I}_{\text{left}}$ do
        $V_I \leftarrow \mathbb{I}(I,Y)$ // Relevance
        $W_I \leftarrow \frac{1}{|\mathcal{I}_{\text{picked}}|}\sum_{T\in\mathcal{I}_{\text{picked}}} \mathbb{S}(I,T)$ // Average redundancy
    end
    if $\min_{I\in\mathcal{I}_{\text{left}}} W_I = 0$ then
        /* If no redundancy, select most relevant */
        $I^* \leftarrow \mathrm{argmax}_{I\in\mathcal{I}_{\text{left}}:\, W_I=0} V_I$
    else
        /* Otherwise, pick best compromise */
        $I^* \leftarrow \mathrm{argmax}_{I\in\mathcal{I}_{\text{left}}} (V_I/W_I)$
    end
    $\mathcal{I}_{\text{picked}} \leftarrow \mathcal{I}_{\text{picked}} \cup \{I^*\}$
    $\mathcal{I}_{\text{left}} \leftarrow \mathcal{I}_{\delta,l_{\max}} \setminus \mathcal{I}_{\text{picked}}$
end
return $\mathcal{I}_{\text{picked}}$
```

Remark. It is interesting to note that the representation formula considered in Equation (3.4) is completely general. By that, we mean that any model (including the best one) can be written in such form. To see this, we note that any predictive model would take the form

$$\mathbb{P}(Y=1) = f(S_1,\dots,S_p)$$

for some $f : \{0,1,2\}^p \to [0,1]$. Equivalently, $\mathrm{logit}\mathbb{P}(Y=1) = g(S_1,\dots,S_p)$, where $g := \mathrm{logit}\circ f$. Depending on the approach, the actual definition of *the optimal model* may change, but it will nevertheless be given by some $g_* : \{0,1,2\}^p \to \mathbb{R}$. Now let $A := \{0,1,2\}^p$. Since A is finite, it is straightforward to see that

$$g_*(S_1,\dots,S_p) = \sum_{\mathbf{a}\in A} g_*(\mathbf{a})\mathbf{1}_{\{\mathbf{a}\}}(S_1,\dots,S_p).$$

In particular, if we let

$$K := |A| = 3^p, \quad \beta_k := g_*(\mathbf{a}_k), \quad I_k := \prod_{j=1}^p \mathbf{1}_{\{a_k^{(j)}\}}(S_j),$$

where $A = \{\mathbf{a}_k\}_{k=1}^K$ and $\mathbf{a}_k = [a_k^{(1)},\dots,a_k^{(p)}]$, then we exactly recover Equation (3.4). Of course, in practice, our idea is to resort to much smaller values of K by exploiting the use of general interaction terms of any order.

Remark. Despite the great efficiency of data mining algorithms, such as *FP-Growth*, the computational cost of *hi*PRS may still grow exponentially with the number of SNPs, p. In particular, even though we may alleviate this cost by posing tighter constraints on the hyperparameters δ and $l_{\max}$, and even if the minority class is fairly small, the algorithm cannot scale to the dimensions addressed in GWAS, that is $p \propto 10^6$. In light of this fact, it is fundamental to either support *hi*PRS with previous knowledge coming from the literature, or to resort, in a preliminary stage, to some reliable and scalable feature selection algorithm, such as the DSAEE approach in Chapter 2.

Remark. The computation of the empirical mutual information, $\mathbb{I}(I, Y)$, in Equation (3.3), pretends the preliminary computation of the supports over both the minority and majority class. In particular, let $L := |\mathcal{I}_{\delta,l_{\max}}|$. To evaluate (3.3), we are required to compute the data matrix $\mathbf{C} = (c_{j,k})_{j,k} \in \mathbb{R}^{N\times L}$, where $c_{j,k}$ is the observed value of the kth candidate interaction in the jth patient. In fact, since we restricted our preliminary search to the minority class, it is not straightforward to fill all the entries in $\mathbf{C}$. Also, as the number of candidate patterns is typically very large, evaluating each interaction term by its formula can be unnecessarily expensive. Parallel computing may surely help, but we can also exploit the particular structure of patterns to grant faster computations. The idea goes as follows. Let $\{Z_i\}_{i=1}^{3p}$ be the dummy variables associated to the p SNPs, that is

$$Z_i = \mathbf{1}_{\{\text{mod}(i,3)\}}(S_{\text{ceil}(i/3)}),$$

where $\text{mod}(a,b)$ is the modulus remainder of the integer division of a and b, while $\text{ceil}(a) := \min\{c \in \mathbb{N} \mid c \geq a\}$. The corresponding data matrix is then $\mathbf{Z} := (z_{j,i})_{j,i} \in \mathbb{R}^{N\times 3p}$. Let us define the *incompatibility matrix* $\mathbf{M} := (m_{i,k})_{i,k} \in \mathbb{R}^{3p\times L}$ as

$$m_{i,k} := \begin{cases} 1 & Z_i \cdot I_k \equiv 0 \\ 0 & \text{otherwise} \end{cases},$$

where $\mathcal{I}_{\delta,l_{\max}} = \{I_k\}_{k=1}^{L}$. If we now consider $\mathbf{Z}, \mathbf{M}$ and $\mathbf{C}$ as logical matrices, that is, matrices having entries in the boolean domain $\{0,1\}$, then it is straightforward to prove that

$$\mathbf{C} = \neg\left(\mathbf{Z}\cdot\mathbf{M}\right), \tag{3.5}$$

where the matrix product is intended in the boolean sense, whereas $\neg$ is the common notation for the logical negation operator, here applied entrywise. To see that (3.5) holds, consider the jth observation for the kth interaction, $c_{j,k}$. By definition of logic sum and product, the corresponding term on the right-hand side of (3.5) is

$$\neg\left(\sum_{i=1}^{3p} z_{j,i}m_{i,k}\right) = \neg\left(\exists i \in \{1,\dots,3p\} \mid z_{j,i} \wedge m_{i,k}\right) =$$
$$= \neg\left(\exists i \in \{1,\dots,3p\} \mid z_{j,i} \,\wedge\, Z_i\cdot I_k \equiv 0\right) = \neg\left(\neg i_{j,k}\right) = c_{j,k}.$$

Equivalently, the above states that a pattern is not observed in a patient if and only if the latter reports the presence of an incompatible dummie variable. With this approach, the computational time for constructing the data matrix $\mathbf{C}$ is $\mathcal{O}(pLN^{\alpha})$, with $\alpha \leq 0.3728596$, cf. [4, 104]. In particular, if $p < N^{1-\alpha}$, this procedure can be substantially better with respect to the naive term by term computation, which would require $\mathcal{O}(NL)$ operations.

3.3 Simulation study

In this Section, we assess the predictive power of *hi*PRS and its capability to capture interactions in the generative mechanism of the phenotype. To do so, we run a large set of experiments within a simulated environtment, whose generative mechanism was designed to present complex nonlinear dependencies between a binary outcome Y and high-order SNP-SNP interactions. In particular, we test *hi*PRS against several benchmark scoring algorithms and we verify its robustness with respect to sample size, class imbalance and noise.

To this end, we shall first describe the mathematical model underlying the simulated data and the benchmark algorithms. Then, we report all the different experiments and their results.

3.3.1 Generative model

The synthetic data was built as follows. We considered a pool of $p = 15$ independent SNPs, S_1,...,S_p, and a binary outcome variable Y. We assume each of the SNPs to follow a uniform distribution over $\{0,1,2\}$. In order to impose a nonlinear dependence between the two, we considered a model of the following form

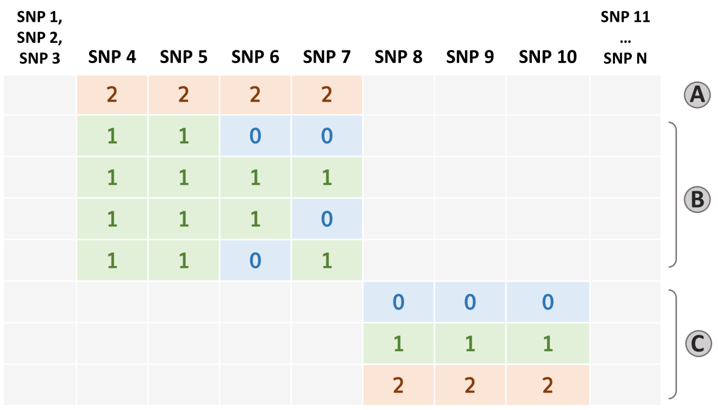
$$Y := (1-F)\tilde{Y} + F(1-\tilde{Y}),$$

where $\tilde{Y}$ is univocally determined by the fifteen SNPs via the relation below (see also Figure 3.3),

$$\begin{aligned}\tilde{Y} = 1 \iff \{S_4 = 2,\ S_5 = 2,\ S_6 = 2,\ S_7 = 2\} \vee \\ \{S_4 = 1,\ S_5 = 1,\ S_6 \neq 2,\ S_7 \neq 2) \vee \{S_8 = S_9 = S_{10}\},\end{aligned}$$

whereas $F \sim B(\varepsilon)$ is a binary random variable, independent on the SNPs, that we use to model either noise in the data or unexplained variability. Note in fact that if $F = 0$, then $\tilde{Y}$ and Y coincide, otherwise the labels are flipped. It is worth noting the following facts.

1. Let $\tilde{p}_Y := \mathbb{P}(\tilde{Y} = 1)$ and $p_Y := \mathbb{P}(Y = 1)$. Upto some basic calculations, one can show that
$$\tilde{p}_Y = 121/729 \approx 16.6\% \quad \text{and} \quad p_Y = \tilde{p}_Y(1-\varepsilon) + (1-\tilde{p}_Y)\varepsilon. \tag{3.6}$$
To see this, define the events $A := \{S_4 = 2,\ S_5 = 2,\ S_6 = 2,\ S_7 = 2\}$, $B := \{S_4 = 1,\ S_5 = 1,\ S_6 \neq 2,\ S_7 \neq 2\}$, $C := \{S_8 = S_9 = S_{10}\}$. Since we assumed the SNPs to be independent and uniformly distributed, it follows that
$$\mathbb{P}(A) = (1/3)^4,\ \ \mathbb{P}(B) = (1/3)^2 \cdot (2/3)^2,\ \ \mathbb{P}(C) = 3 \cdot (1/3)^3.$$
Also, since A and B are disjoint, $\mathbb{P}(A \cup B) = 1/81 + 4/81 = 5/81$. Finally, since $A \cup B$ and C are independent, we have
$$\begin{aligned}\tilde{p}_Y = \mathbb{P}(A \cup B \cup C) = \mathbb{P}(A \cup B) + \mathbb{P}(C) - \mathbb{P}((A \cup B) \cap C) = \\ = \mathbb{P}(A \cup B) + \mathbb{P}(C) - \mathbb{P}(A \cup B)\mathbb{P}(C) = 5/81 + 1/9 - (5/81)(1/9) = 121/729.\end{aligned}$$
thus proving the first statement in (3.6). Conversely, the second one is obvious as the independence of $\tilde{Y}$ and F implies
$$p_Y = \mathbb{E}[Y] = \mathbb{E}[(1-F)\tilde{Y}] + \mathbb{E}[F(1-\tilde{Y})] = \mathbb{E}[1-F]\mathbb{E}[Y] + \mathbb{E}[F]\mathbb{E}[1-\tilde{Y}].$$
In particular, we note that for $\varepsilon \leq 0.2$ we have $p_Y \leq 30\%$, resulting in a class imbalance where cases are the minority.



SNP 1, SNP 2, SNP 3	SNP 4	SNP 5	SNP 6	SNP 7	SNP 8	SNP 9	SNP 10	SNP 11 ... SNP N	
	2	2	2	2					A
	1	1	0	0					B
	1	1	1	1					
	1	1	1	0					
	1	1	0	1					
					0	0	0		C
					1	1	1		
					2	2	2		

Figure 3.3: Graphical representation of the three rules describing the positive class. Cases are obtained when $\mathbf{A} \vee \mathbf{B} \vee \mathbf{C}$. More details on the generative model are provided in Section 3.3.1.

2. We can explicitely quantify the amount of variability in Y that cannot be explained by the SNPs only. Indeed,

$$1 - \frac{\mathrm{Var}\left(\mathbb{E}[Y|S_1,\dots,S_p]\right)}{\mathrm{Var}(Y)} = 1 - \frac{(1-2\varepsilon)^2\tilde{p}_Y(1-\tilde{p}_Y)}{p_Y(1-p_Y)}, \tag{3.7}$$

since $\mathbb{E}[\tilde{Y}|S_1,\dots,S_p] = \tilde{Y}$ and thus $\mathbb{E}[Y|S_1,\dots,S_p] = (1-2\varepsilon)\tilde{Y} + \varepsilon$, by exploiting independence and classical properties of conditional expectations. We refer to (3.7) as to *missing heritability*. In our final experiment, we use the latter to provide better insights on the *hi*PRS performance for different values of ε.

3.3.2 Benchmark algorithms

We report below a short description and the benchmark algorithms together with their implementation details.

Penalized PRSs with additive effects. From this class of methods we chose three of *hi*PRS competitors, namely Lasso [191], Ridge [87] and Elastic-Net [208]. These algorithms are traditional penalized LR models that only account for the additive main effect of the predictors on the target variable, while imposing an ℓ_2 bound of the form $||\beta||_2^2 \leq s$ (Ridge) or an ℓ_1 bound $||\beta|| \leq s$ (Lasso), or the combination of the two (Elastic-Net) on the coefficients. Note that ℓ_1 penalizations also perform feature selection by shrinking coefficients to zero, making these approaches popular to model polygenic risk from large genotype data. Fundamentally, the penalization terms enter the loss function during the training phase, and their contribution is weighted by some hyperparameter, e.g. λ for Lasso and Ridge, λ_1 and λ_2 for Elastic-Net. To run these algorithms in our experiments, we relied on the implementation available in the Python library scikit-learn, with default values for λ, λ_1 and λ_2. All the code was written in Python 3.7.

Glinternet. Glinternet is a method for learning pairwise interactions in a LR satisfying strong hierarchy: whenever an interaction is estimated to be nonzero, both its associated main effects are also included in the model. The idea of the algorithm is based on a variant of Lasso, namely group-Lasso [204], that sets groups of predictors to zero. Glinternet sets up main effects and first-order (i.e., pairwise) interactions via groups of variables and then intuitively selects those that have a strong overall contribution from all their levels toward explaining the response. For more formal definitions we refer the reader to [123].

The amount of regularization is controlled by λ, with larger values corresponding to stronger shrinkage and less interactions included. The freely distributed R implementation of glinternet allows the user to define the number of pairwise interactions (n_int) to find and the size of a grid of λ values of decreasing strength to fit the model with. This grid is built automatically by splitting equally from λ_{max}, that is data-derived as the value for which all coefficients are zero, to the minimum value, computed as $\lambda_{min} = \lambda_{max}/0.01$. The algorithm will fit this path of values and stop when n_int is reached. For this experiment we set $n\_int = [3, 8]$, defining a grid of 100 λs. All the other hyperparameters were left to default settings.

DNN-Badre. DNN-Badre [10] constructs a Deep Feed Forward Neural Network (DNN) to predict a binary outcome. To implement their approach, we built a DNN architecture with the same specifics provided by the authors, i.e. number of layers, neurons and activation functions. We trained the models via stochastic gradient descent, with mini-batches of batch size 10 and for a total of 200 epochs. The optimization was carried out using the Adam optimizer, with a learning rate of 10^{-3}. All the code was written in Python 3.7, mostly using the Pytorch library.

SVM-Behravan. SVM-Behravan is a multi-step Machine Learning based algorithm recently presented as a state-of-the-art approach to polygenic risk scoring [16, 15]. We will provide here an intuitive description of the algorithm with the needed details to understand our settings for the present work, while for more technical specifications we refer the reader to the seminal work in [16]. The algorithm presented in Behravan et al. (2018) is composed of a so called *first module*, where an XGBoost is used to evaluate the importance of SNPs on a risk prediction task by providing an initial list of candidate predictive SNPs. The authors use the average of feature importances (a.k.a. “gain”) provided by the gradient tree boosting method, as the contribution of each SNP to the risk. Then, in the *second module*, the candidate SNPs are used for an adaptive iterative search to capture the optimal ways of combining candidate SNPs to achieve high risk prediction accuracy on validation data. In particular, top M and bottom M SNPs from the candidate list are ranked separately based on accuracy, then top and bottom N SNPs are switched between the two lists. The process is repeated with M (i.e., *window size*) increased of W at each iteration, until the two sublists overlap and the optimal ranking is achieved. Finally, an SVM is trained to distinguish cases (positive samples) and controls (negative samples) using the S top-ranked SNPs in the optimal ranking as feature vectors and a linear kernel.

In the original paper the performance of the algorithm is averaged across 5-fold CV, meaning that the pipeline from first module to SVM is repeated 5 times on different folds of the training and test set. As the author included this step to overcome the problem of small samples to train high-performance risk prediction models, we followed their instructions and for each of the 30 simulated dataset we performed the 5-fold CV. However, to keep the running time of this computationally very expensive method reasonable and for a fair comparison with all the other methods for which we did not perform extensive optimal hyperparameter search, we avoided the very long optimization of the XGBoost model by setting manually the optimal hyperparameters described in the paper. In particular, we set $M = 2$, $W = 1$ and $N = 1$.

3.3.3 Performance metrics

In imbalanced settings, the Accuracy of a classifier (i.e., the fraction of correctly classified observations) can be a misleading metric to assess a model performance. Therefore, to evaluate *hi*PRS and the benchmark algorithms in predicting the outcome, we exploited two metrics that combined provide a full picture of the real performance of these methods, especially in predicting the most interesting class (i.e. the positive class), which is oftentimes the underrepresented one in real data. Therefore, for our experiments we chose the Area Under the receiving operating characteristic Curve (AUC) and the Average Precision (AP).

The two are computed as follows. For any fixed discrimination threshold, let TP and FP be the true and false positives, respectively. Similarly, let TN and FN be the true and false negatives, respectively. First, we derive the following quantities,

$$TPR = \frac{TP}{TP + FP}, \qquad FPR = \frac{FP}{FP + FN}, \qquad Recall = \frac{TP}{TP + FN}.$$

that is, the True Positive Rate (TPR), the False Positive Rate (FPR) and the Recall. TPR is the fraction of true positives out of the positives, also known as *sensitivity* or *precision*. FPR, or *specificity*, is the fraction of false positives out of the negatives. Finally, the Recall quantifies the ability of the classifier to recognize positive samples. By plotting TPR against FPR for various discrimination threshold levels, we obtain the ROC curve. Conversely, the pair $(Recall, TPR)$ yields the precision-recall curve. AUC summarizes the performance of a binary classifier by computing the area under the ROC curve, while AP considers the area under the precision-recall curve. In practice, the two areas are estimated using quadrature rules. In our implementation, we employ the trapezoidal rule for AUC, while we use the rectangular rule for AP.

3.3.4 Results

We now describe the results obtained for a large set of experiments. We recall that the simulated data was designed to present complex nonlinear dependencies between a binary outcome Y and $p = 15$ SNPs. In particular, the positive class in the generated data, namely $\{Y = 1\}$, was defined in terms of the rules reported in Figure 3.3, which are ultimately high-order interactions. Additionally, each of the observed outcome values was mislabeled with probability ε (hereby called *random noise*), in an attempt to make the relationship between the outcome and the predictors less deterministic.

*hi*PRS outperforms benchmark algorithms in prediction performance

To judge the performance of *hi*PRS, we run a first experiment on simulated data and we compared the results with those of a comprehensive set of benchmark algorithms coming both from the traditional and more recent literature on polygenic risk prediction. Note that, to ensure a fair comparison, we only included methods that did not rely on any type of external information (e.g. summary statistics) besides individual-level genotype. More precisely, we recall that we picked methods from the following two classes: PRSs taking raw SNP values as input (Penalized PRSs) and algorithms from the Machine Learning literature. In the first group we included Lasso [191], Ridge [87], Elastic-Net [208] and glinternet [123]. The first three are penalized LR-based PRS methods with additive main effects and no interactions, while the last one is a penalized LR with group-Lasso regularization that includes second-order interactions. Conversely, as Machine Learning methods, we included two scoring algorithms accounting for high-order interactions: the approach proposed by Behravan et al. [16], that exploits XGboost for interaction selection and SVM for classification, and the one by Badre et al. [10], that relies on a DNN model. Specific technical details on each of these approaches, together with their implementation and chosen hyper-parameters, can be found in Section 3.3.2. All of this was considered within an experimental setting of mild complexity (i.e. reasonable sample size and low noise), consisting of 30 independent simulations. For each of these we generated 1000 observations for training and 500 observations for testing. The mislabeling *noise* probability was set to $\varepsilon = 0.01$. The glinternet algorithm allows the user to define the number of interaction terms to include in the model, which we set to 3 and 8. Note that this method includes main effects of all considered interactions and estimates a parameter for each categorical level and their products. Therefore, 8 interaction terms actually correspond to more than $8 \times 3^2 = 72$ regressors in the LR model (in particular, in our experiments we obtained 93 of them). In light of this, we chose 8 as maximum number of interactions, as that reflected the amount of generating rules in the simulated data (cf. Figure 3.3), while the value of 3 is meant to test the performance of a simpler model. Differently from glinternet, in *hi*PRS the number

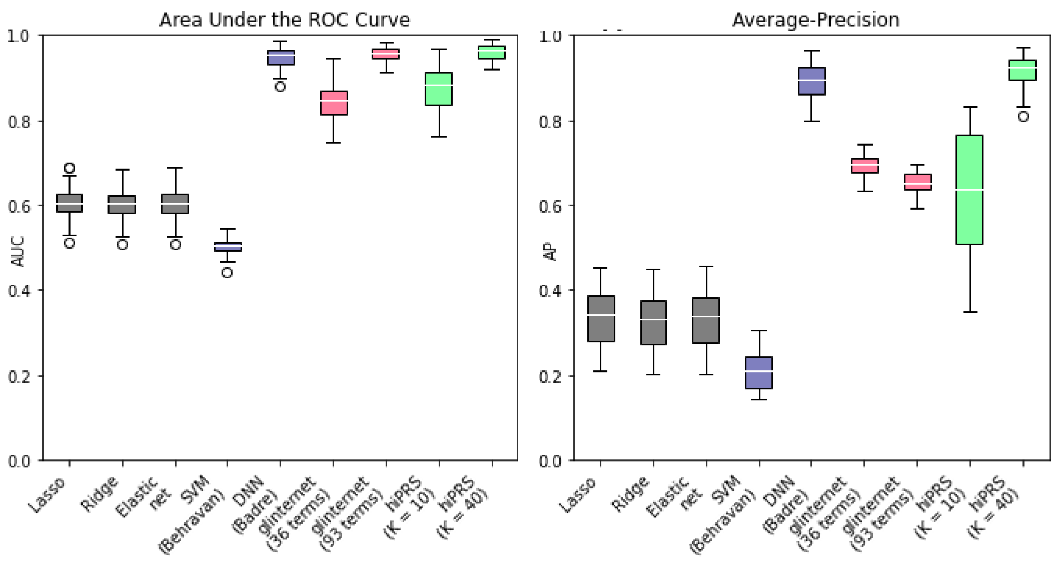
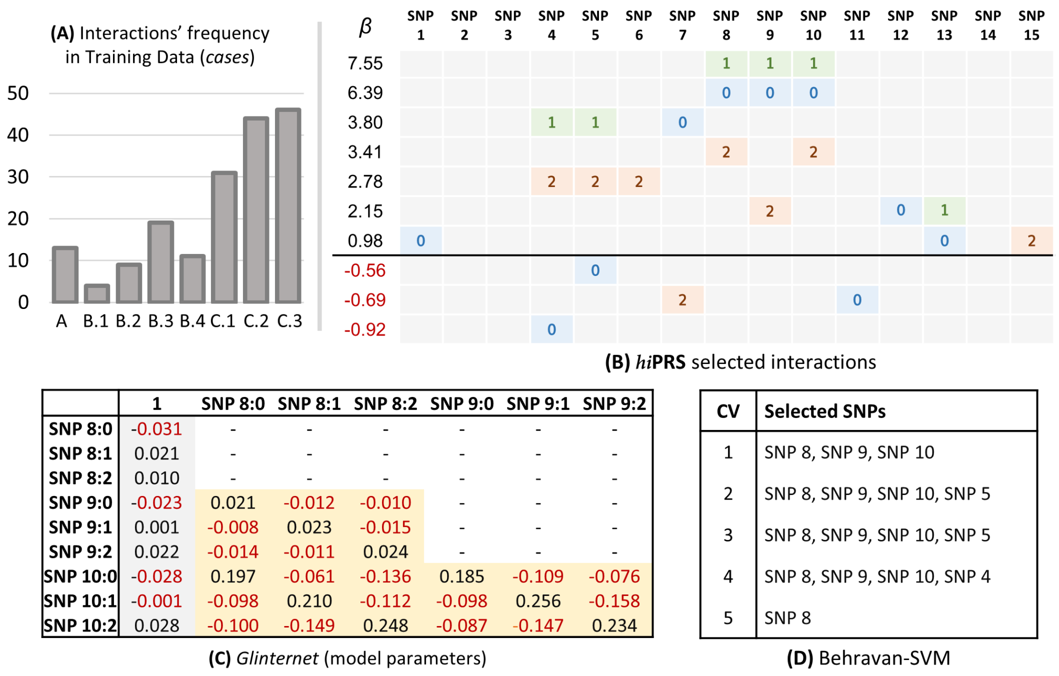


Figure 3.4: Results on risk prediction against benchmark PRSs and Machine Learning approaches, Simulation study, Section 3.3. AUC (left) and AP (right) performance distributions of 30 independent trials. In grey the three traditional penalized PRSs approaches with additive effects only; in violet the two ML algorithms (SVM-Behravan and DNN-Badre); in pink glinternet algorithm for two model dimensions (3 interactions, i.e. 36 terms, and 8 interactions, i.e. 93 terms); in green *hi*PRS for $K = 10$ and $K = 40$.

of interaction terms K actually corresponds to the final dimension of the model. Here, we set $K = 10, 40$, resulting in two models of different complexity. In principle, both of them have enough degrees of freedom to discover the generative model, nevertheless, they both allow for handy readability and interpretability of the fitted model. We did not impose any limit on the order of the interactions, whereas we set the frequency threshold δ to 0.05.

In Figure 3.4, we report the boxplots of the performance metrics in the 30 independent trials. The three Penalized PRSs behave similarly, with an average AUC around 0.6 and an Average Precision (AP) slightly above 0.3. Their performance reflects the maximum predictive power achievable by additive PRSs in the presence of epistatis. SVM-Behravan has the worst performance among the classifiers on both metrics, probably due to the fact that despite the SNP selection step accounts for interactions, it is followed by a linear SVM classifier that considers additive effects only. The DNN is unsurprisingly extremely performing, but its black box nature does not allow us to identify the role of the predictors in scoring observations. Moreover, whilst still high, it has a slightly lower and more variable AP. This might be due to the tendency to overfit the small sample of positive observations during training, possibly caused by the very large number of parameters typical of these models. Glinternet best AUC performance is achieved when including 8 interactions. However, the same model has a significant drop in AP, falling behind its simpler version with 36 terms: this goes to show that, despite the inclusion of more predictors, the identified interactions are not necessarily the most useful to identify the class of interest when the data is imbalanced. Moreover, the 93 parameters associated to the 8 interactions could easily overfit the underrepresented positive class. The best performer overall is *hi*PRS modeling 40 terms, showcasing the ability of our approach to identify predictive interactions that generalize well irrespectively of the under-representation of the positive class. The results obtained for $K = 10$ are also remarkable, especially if we consider that they correspond to the model having the least number of parameters.



β	SNP 1	SNP 2	SNP 3	SNP 4	SNP 5	SNP 6	SNP 7	SNP 8	SNP 9	SNP 10	SNP 11	SNP 12	SNP 13	SNP 14	SNP 15
7.55								1	1	1					
6.39								0	0	0					
3.80				1	1		0								
3.41								2		2					
2.78				2	2	2									
2.15									2			0	1		
0.98	0												0		2
-0.56					0										
-0.69							2				0				
-0.92				0											

(B) *hi***PRS** selected interactions

	1	SNP 8:0	SNP 8:1	SNP 8:2	SNP 9:0	SNP 9:1	SNP 9:2
SNP 8:0	-0.031	-	-	-	-	-	-
SNP 8:1	0.021	-	-	-	-	-	-
SNP 8:2	0.010	-	-	-	-	-	-
SNP 9:0	-0.023	0.021	-0.012	-0.010	-	-	-
SNP 9:1	0.001	-0.008	0.023	-0.015	-	-	-
SNP 9:2	0.022	-0.014	-0.011	0.024	-	-	-
SNP 10:0	-0.028	0.197	-0.061	-0.136	0.185	-0.109	-0.076
SNP 10:1	-0.001	-0.098	0.210	-0.112	-0.098	0.256	-0.158
SNP 10:2	0.028	-0.100	-0.149	0.248	-0.087	-0.147	0.234

(C) *Glinternet* (model parameters)

CV	Selected SNPs
1	SNP 8, SNP 9, SNP 10
2	SNP 8, SNP 9, SNP 10, SNP 5
3	SNP 8, SNP 9, SNP 10, SNP 5
4	SNP 8, SNP 9, SNP 10, SNP 4
5	SNP 8

(D) Behravan-SVM

Figure 3.5: Interpretability analysis of *hi*PRS, Simulation study, Section 3.3. (A) Absolute frequency of the generative rules in the training data, limited to the positive class. (B) Interactions selected by *hi*PRS with $K = 10$ and corresponding β coefficients. (C) Coefficients of the glinternet model with 3 interaction terms: main effects are in gray, interactions in yellow. (D) Lists of SNPs selected by SVM-Behravan during its five internal cross validations, cf. Benchmark Algorithms in the Materials and Methods Section. Note: reported results are limited to one simulation among the 30 randomly generated datasets.

*hi*PRS captures and explains interaction-based generative mechanisms better than benchmarks

One of the added values of *hi*PRS is the capability of capturing dependencies among predictors, by selecting the most relevant to determine the phenotype and presenting them within a simple and interpretable model. In many research settings, a slight loss in prediction quality may be acceptable if it leads to a more meaningful interpretation of the predictors [36, 89]. To test the ability of *hi*PRS in capturing them, we focused on one of the previously mentioned datasets, and checked the 10 interactions selected by the simplest model, $K = 10$. We report the selected interactions in Figure 3.5. We mention that, to ensure a fair comparison, we picked the dataset where the worst performer, SVM-Behravan, achieved one of its highest AUC. This dataset contained 182 cases in the training set, i.e. 18.2% of the total. The bar chart in Figure 3.5 reports the absolute frequency of the generating rules available in the positive class. *hi*PRS captures most of rule C, assigning very high positive betas to the first, second and fourth interaction. Combined together, the fourth and fifth selected interactions partially recover rule A, while the third one almost fully captures rule B.3. Note that, among positive samples in the training data, B.3 is the most frequent version of rule B (see Figure 3.5, left panel). It is also interesting to note that *hi*PRS finds two protective terms, namely $\{SNP_4 = 0\}$ and $\{SNP_5 = 0\}$, that both nullify rules A and B.

Let us now briefly discuss the results obtained by the competitors. In panel (c) of Figure 3.5 we report the model parameters for glinternet when modeling 3 interactions. Here, we see that the good performance of glinternet is granted by the inclusion of three interaction terms in the model, which, despite being of second order only, are subsets of the true generating rule C. Moreover, the estimated effect sizes are correctly positive for couples of identical allele frequencies only (i.e., the diagonal of the matrices of coefficients associated

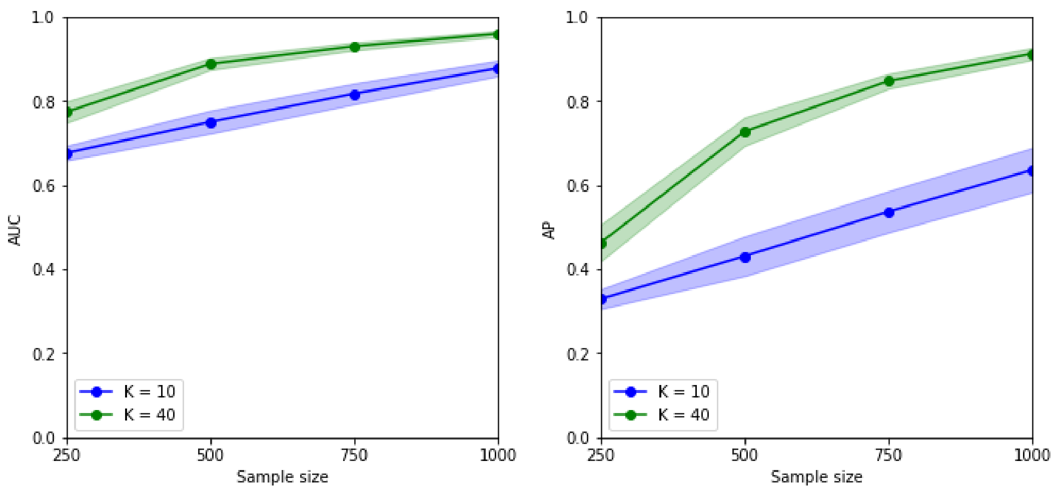


Figure 3.6: Sensitivity analysis with respect to sample size, Simulation study, Section 3.3. Average performance of *hi*PRS in terms of AUC (left and AP (right).

to each interaction). However, due to extremely large number of parameters, it is very hard to inspect the model and drive suitable conclusions. Furthermore, main effect sizes are not truly relevant in determining the phenotype and the generative mechanism is only captured partially, as rule C is the only one that is actually identified. Within the same Figure, but in panel (d), we list the SNPs selected by SVM-Behravan during its 5-fold cross-validation. Notably, SVM-Behravan is able to recover some of the SNPs associated with the generating rules, however, we are left with no information on the structure of the interactions.

*hi*PRS can deal with extremely small sample size

We tested *hi*PRS performance for small sample sizes. We believe this to be a relevant setting, as in most real research scenarios clinicians have to deal with individual-level data of significantly small cohorts. To this end, starting from the sample size that we considered in our previous experiments, we repeated our analysis on a sequence of four decreasing sample sizes, namely $n = 1000, 750, 500, 250$. That is, for each n, we run 30 independent simulations, and we registered *hi*PRS performance in terms of AUC and AP. To ensure comparable results, the noise level was fixed to $\varepsilon = 0.01$ for all experiments, and the model was always evaluated on a test set of 500 instances. Once again, we tested *hi*PRS with $K \in \{10, 40\}$, while $\delta = 0.05$ as before. Results are reported in Figure 3.6. Despite the unavoidable decrease in performance, we note that even for very small samples, 250 observations, *hi*PRS is able to provide insightful results, with AUC levels of 0.8 and ~ 0.7 respectively. AP is lower, ~ 0.5 for $K = 40$, but still significantly better than traditional penalized PRSs when trained on 1000 observations. Nonetheless, note that a training sample of 250 observations in total corresponds to less then 75 cases to learn from (cf. Simulated Data in Materials and Methods), which is an extremely challenging setting. For 500 training samples (i.e. ~ 150 cases at most) *hi*PRS with $K = 40$ reaches almost 0.9 AUC and an AP of ~ 0.7. These results testify in favour of *hi*PRS generalization potential, which is likely induced by the mRMR-based interaction selection algorithm. Indeed, the latter is optimized to select the most predictive features, while favouring the introduction of *diverse* information in the model. Minimizing redundancy in the selection makes *hi*PRS less prone to overfitting, irrespectively of the sample size or the number of cases.

*hi*PRS is robust to class imbalance

To test *hi*PRS against extreme class imbalance, we had to modify slightly our procedure in the generation of the simulated data. This is because although our generative mechanism

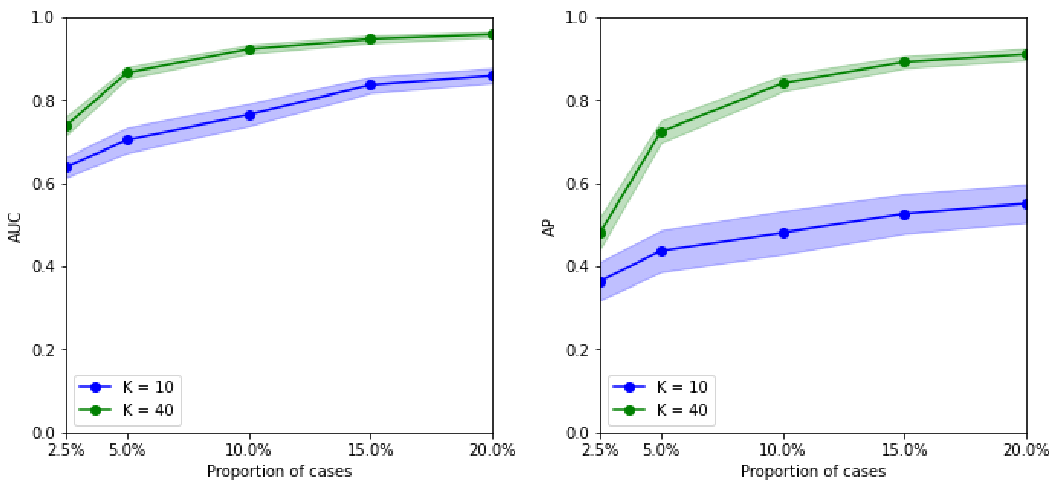


Figure 3.7: Sensitivity analysis with respect to class imbalance, Simulation study, Section 3.3. Average performance of *hi*PRS in terms of AUC (left and AP (right).

is based upon random generation of allele categories and random noise (ε), it also relies on a deterministic rule-based definition of the positive class. Therefore, the positive class will always appear in the data with an approximately fixed frequency depending on ε. We overcame this drawback via undersampling. More precisely, let $0 \leq q \leq 1$ be any wished proportion. To obtain a training sample of 1000 instances and $1000q$ cases, we generate a larger dataset where observations are added iteratively until there are at least $1000q$ cases and $1000(1-q)$ controls, then we discard all exceeding observations. We adopted this procedure for a varying proportion of cases, namely q = 2.5%, 5%, 10%, 15% and 20%. For each of those values, we generated 30 independent training sets, and measured *hi*PRS performance over as many independent test sets of 500 instances. We mention that, while we used subsampling to generate the training data, we stuck to our usual approach for the test data. Indeed, the complexity lies in training the models on imbalanced data; furthermore, the distribution of classes in the test set has no impact on the metrics that we use for evaluation, i.e. AUC and AP.

We report in Figure 3.7 the results for this simulation study. Notably, even for the smallest proportion of observations in the positive class (2.5%), *hi*PRS succeeds in learning a sufficiently generalizable set of interactions, with AUC values above 0.6 when $K = 10$, and above 0.75 when $K = 40$. Moreover, the larger model goes beyond 0.7 on both AUC and AP for a proportion of cases of 5%, meaning only 50 observations to learn from.

*hi*PRS is robust to variability (missing heritability)

Missing heritability is the proportion of variance in the phenotype that cannot be explained by genotype information [130]. This variability can be induced by several factors [130]: one is the need to account for SNP-SNP interactions when modeling phenotypic traits [129], but it can be nonetheless induced by factors that cannot be modeled with the data at hand. In our simulation setting, we reproduce the effect of missing heritability via the *noise* parameter, as the two can be easily linked together. The explicit way by which the missing heritability depends on ε is detailed in Equation (3.7). To test *hi*PRS robustness to variability in the training data, we generate 30 training and test sets, respectively with 1000 and 500 observations, for each of the following noise levels, $\varepsilon = 0.01$, 0.02, 0.05, 0.1 and 0.2. Results are in Figure 3.8. Note that, up to a noise level of 10% (i.e. ~ 100 mislabeled observations in the training set, and ~ 50 in the test set), both models maintain an average AUC near 0.8. This is a remarkable fact if we consider that a noise of $\varepsilon = 0.1$ corresponds to a missing heritability of 51%, i.e. a setting where the SNPs themselves can only explain at most 49% of the target variability.

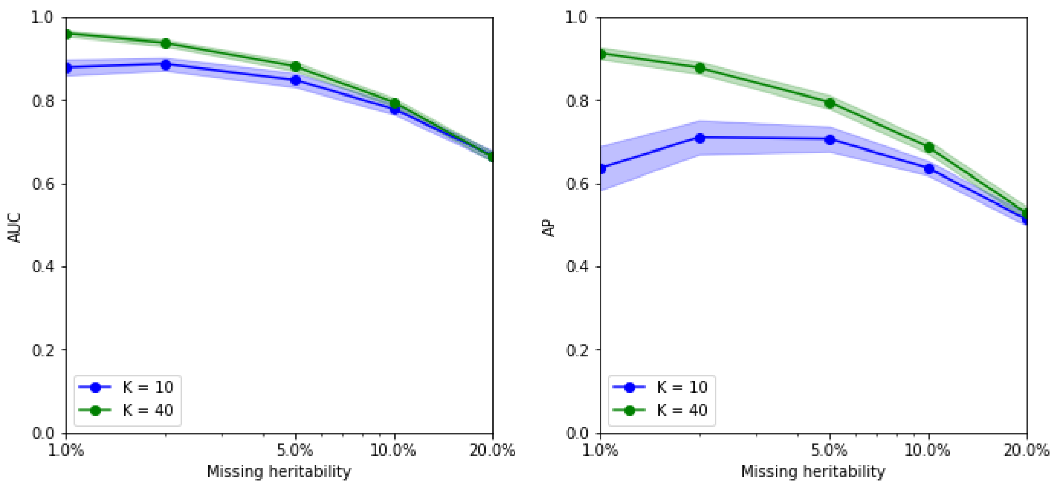


Figure 3.8: Sensitivity analysis with respect to noise (equivalently, missing heritability, cf Equation 3.7), Simulation study, Section 3.3. Average performance of *hi*PRS in terms of AUC (left and AP (right).

3.4 Favoring interpretability: Splitted *hi*PRS

Sometimes, it is more important to favor clinical interpretability with respect to predictive performance. To this end, we propose an alternative implementation of *hi*PRS that can better accommodate this necessity. In Section 3.2, we ranked the candidate interactions $\mathcal{I}_{\delta,l_{\max}}$ through a criteria based on mutual information. In general, the latter is only a measure of association and it does not distinguish between positive or negative effects. To account for this, we may replace mutual information scores with odds-ratios. More precisely, for each candidate interaction $I \in \mathcal{I}_{\delta,l_{\max}}$ we compute its corresponding odds-ratio with respect to Y, namely,

$$\mathrm{OR}(I,Y) := \frac{\left(\sum_{j:\, i_j=1} y_j\right) / \left(\sum_{j:\, i_j=1}(1-y_j)\right)}{\left(\sum_{j:\, i_j=0} y_j\right) / \left(\sum_{j:\, i_j=0}(1-y_j)\right)}.$$

Then, we use the latter to split the candidates list into two sublists,

$$\mathcal{R}_{\delta,l_{\max}} := \{I \in \mathcal{I}_{\delta,l_{\max}} \mid \mathrm{OR}(I,Y) > 1\}, \quad \mathcal{P}_{\delta,l_{\max}} := \{I \in \mathcal{I}_{\delta,l_{\max}} \mid \mathrm{OR}(I,Y) < 1\},$$

respectively consisting of *risk* and *protective* interactions. Given two integers K, J, we then apply our mRMR selection algorithm to the two sublists separately, thus extracting two groups, $\mathcal{R}_K \subset \mathcal{R}_{\delta,l_{\max}}$ and $\mathcal{P}_J \subset \mathcal{P}_{\delta,l_{\max}}$, each, respectively, of dimension K and J. At this stage, the mRMR routine can be implemented exactly as before or using alternative relevance measures. For instance, instead of the mutual information, one may exploit again the odds-ratio to quantify the relevance of each interaction (e,g, by replacing $\mathbb{I}(I,Y)$ with $|\log \mathrm{OR}(I,Y)|$, which works both for the risk and protective case). We take advantage of the two groups to define a risk score R and protection score P,

$$R := \frac{1}{K}\sum_{I \in \mathcal{R}_K} I, \qquad P := \frac{1}{J}\sum_{T \in \mathcal{P}_J} T.$$

The intuition is that the risk score, R, counts the number of risk interactions observed within a given patient, and similarly for the protection score. In this way, the information relevant for inferring on Y gets squished into a single pair of highly interpretable variables. Note however that, here, differently from Equation (3.1), all terms are weighted equally. In this case, the final model is

$$\mathrm{logit}\mathbb{P}\,(Y=1) = \gamma + \alpha R + \beta P, \tag{3.8}$$

leading to the polygenic risk score below,

$$\text{Splitted } hi\text{PRS} = \gamma + \alpha R + \beta P. \tag{3.9}$$

We call this approach *Splitted hi*PRS. As it is highly similar to the original one, we shall not investigate its properties through another simulation study. In general, though, because this model weights multiple interactions with the same coefficient, it is expected to perform slightly worse with respect to *hi*PRS.

3.5 Clinical application

We now apply the proposed methodologies to the clinical cohort presented in Chapter 1, Section 1.3. The analysis include the construction of the polygenic risk score and its integration in NTCP models.

3.5.1 PRS construction and evaluation

We recall that the clinical case study featuring prostate cancer patients concerned the development of a genetic risk score for five different toxicities, namely urinary frequency variation, haematuria, nocturia, decreased stream flow and rectal bleeding (cf. Chapter 1, Section 1.3), reported 1-2 years after radiotherapy. The original dataset contained information about thousands of genetic variants: looking at the literature we identified 43 SNPs to start with, which we later reduced to 13 (per endpoint) in Chapter 2 thanks to the DSAEE algorithm. We have reported them in Table 3.1 for better readability.

In agreement with the clinicians involved, we decided to apply our splitted version of *hi*PRS to further favor clinical interpretability. The procedure for the score generation was carried out separately for each of the five endpoints. Regarding the interaction search, we set a frequency threshold of $\delta = 10\%$ and no upper-bound on the interactions length, $l_{\max} = +\infty$. As the splitted version of *hi*PRS uses odds-ratios to partition the candidates list, we also employed them to quantify the relevance of each interaction alone. That is, as discussed in Section 3.4, we replaced the mutual information with the absolute value of the log odds-ratio. As in principle we had no reason to diversify between risk and protection, we also set $K = J$. The number of interaction terms, K, was then selected via grid search, optimizing the AUC performance over a small subset of possible values, $K \in \{1, \dots, 15\}$. Of note, this latter routine is not particularly expensive as the majority of the computational cost lies in the candidates search and in the computation of the contingency tables, which can be done once for all.

We have reported in Table 3.2 the details about the final model describing the *hi*PRS implementation for each endpoint, cf. Equations (3.8) and (3.9). Further details about the risk and protection patterns can be found in Figures 3.9-3.13, where we have provided a visual representation of the interactions selected by our algorithm. In general, for all endpoints, the risk and protection scores show a statistically significant association with the outcome (the 95% confidence bands never contain the origin). Furthermore, the two are always assigned coefficients with the expected sign, namely $\alpha > 0$, corresponding to an increased risk, and $\beta < 0$, resulting in a protective effect. To better quantify the predictive capability of the proposed genetic score, we have also computed the performance of a benchmark PRS, obtained by directly including all the SNPs in Table 3.1 but without interaction effects. In particular, the latter considers a logistic regression model of the form

$$\text{logit}\mathbb{P}(Y_j = 1) = \beta_0 + \sum_{i=1}^{p} \sum_{k=0}^{2} \beta_{i,j,k} \mathbf{1}_{\{k\}}(S_{i,j}), \tag{3.10}$$

with $p = 13$ and $S_{i,j}$ being the ith SNP selected for the outcome Y_j.

URI	HAE	NOC	DEC	REC
rs141799618	rs10101158	rs10969913	rs10209697	rs62091368
rs8075565	rs708498	rs77530448	rs1799983	rs4775602
rs12591436	rs77530448	rs62091368	rs17362923	rs264631
rs76273496	rs17055178	rs11219068	rs673783	rs17599026
rs10969913	rs147596965	rs264651	rs8098701	rs11122573
rs1799983	rs7366282	rs1799983	rs77530448	rs76273496
rs8098701	rs10969913	rs8098701	rs6535028	rs17362923
rs17599026	rs12591436	rs7366282	rs7366282	rs10969913
rs7366282	rs79604958	rs11122573	rs845552	rs6535028
rs708498	rs8098701	rs17055178	rs1045485	rs10209697
rs17055178	rs845552	rs17599026	rs76273496	rs141799618
rs11122573	rs7829759	rs10497203	rs17055178	rs8098701
rs10209697	rs10209697	rs6432512	rs11122573	rs1045485

Table 3.1: SNPs employed for the polygenic risk scores in the prostate cancer cohort. Toxicity endpoints read as follows: URI = Urinary frequency variation, HAE = Haematuria, NOC = Nocturia, DEC = Decreased stream flow, REC = Rectal bleeding.

	URI	HAE	NOC	DEC	REC
K	15	13	8	15	12
α	13.25 ± 3.86	9.63 ± 3.43	3.22 ± 1.57	7.04 ± 1.94	3.73 ± 1.84
β	-5.37 ± 2.62	-4.60 ± 2.53	-3.82 ± 1.57	-4.51 ± 1.66	-2.48 ± 1.66
γ	-3.27	-3.13	-1.32	-1.63	-2.16
Sensitivity*	67.9%	71.6%	77.6%	64.9%	75.6%
Specificity*	77.9%	60.2%	38.6%	65.6%	45.5%
OR*	7.456	3.818	2.171	3.529	2.593
Cutoff	5.1%	4.5%	17.6%	18.8%	10.3%
AUC	0.78	0.71	0.61	0.68	0.63
Benchmark AUC	0.65	0.63	0.60	0.61	0.61

Table 3.2: Performance of the logistic regression models, fitted using the risk and protection scores as predictors (cf. Equation 3.8). Toxicity endpoints read as in Table 3.1. K = number of risk (protective) interactions, α = risk score coefficient, β = protection score coefficient, γ = intercept, OR = odds-ratio, AUC = Area Under the ROC Curve. The α and β coefficients are reported with their 95% confidence intervals. (*) These rows refer the metrics obtained by thresholding the predicted probabilities in the logistic model according to the cutoff that maximizes the Youden index (third-last row). Benchmark AUC = performance of the additive model, obtained using the SNPs in Table 3.1 but without interaction terms, cf. Equation 3.10.

	URI	HAE	NOC	DEC	REC
Median (toxicity)	0.611	0.740	-0.149	0.168	0.208
Median (no toxicity)	-0.357	0.033	-0.224	-0.133	0.001
Wilcoxon	5.76e-13	9.73e-10	3.48e-07	2.35e-16	8.73e-08
Kolmogorov-Smirnov	3.39e-10	1.41e-06	1.44e-04	1.41e-14	6.52e-06

Table 3.3: Comparison of the PRS distribution in patients with toxicity vs patients without. Wilcoxon = p-value of the Wilcoxon test, used to compare the medians in the two populations. Kolmogorov-Smirnov = p-value of the two-samples Kolmogorov-Smirnov test, used to compare the distribution of the biomarker across the two populations. Endpoints read as in Table 3.1

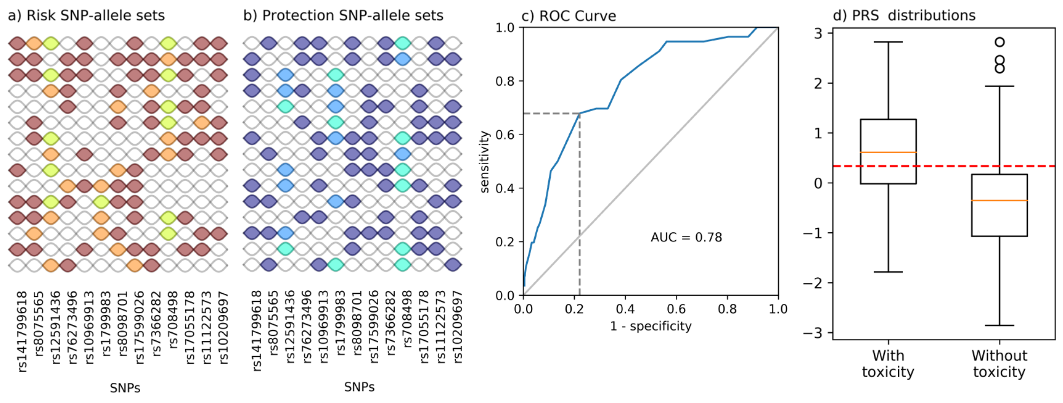


Figure 3.9: Splitted *hi*PRS for the **urinary frequency variation** endpoint. Panel a) Selected risk interactions, encoded as 🟩 = 2, 🟧 = 1, 🟥 = 0. Panel b) Selected protective interactions, encoded as 🟩 = 2, 🟦 = 1, 🟪 = 0. Panel c) Receiver Operating Characteristic (ROC) curve associated to the genetic score. Dashed lines indicate the point maximizing the Youden index (that is, the point with the smallest ℓ_1 distance from the top-left corner). Panel d) Score distribution in patients with toxicity vs patients without. Red dashed line = score value associated to the cutoff with maximum Youden index.

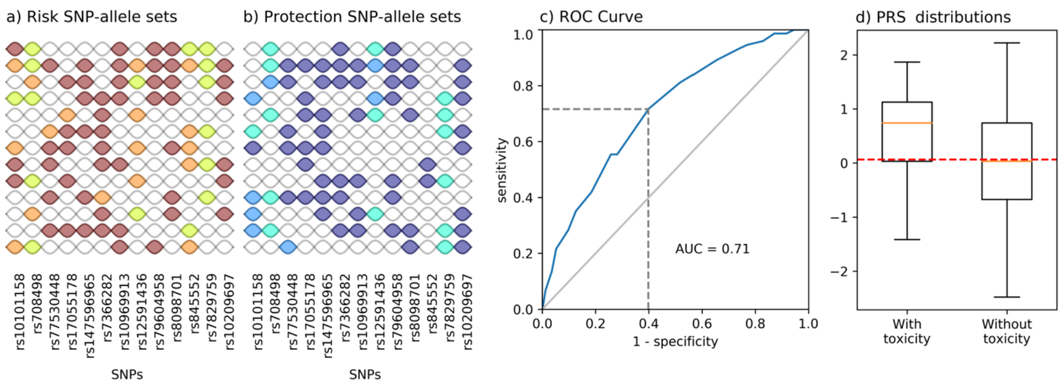


Figure 3.10: Splitted *hi*PRS for the **haematuria** endpoint. Panels read as in Figure 3.9.

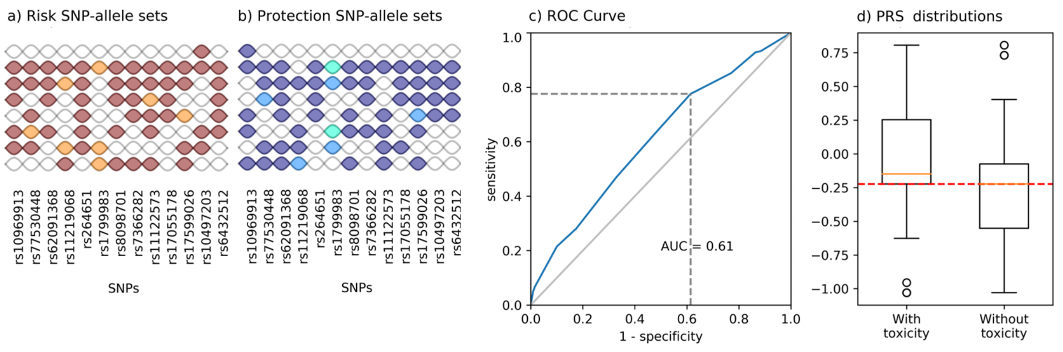


Figure 3.11: Splitted *hi*PRS for the **nocturia** endpoint. Panels read as in Figure 3.9.

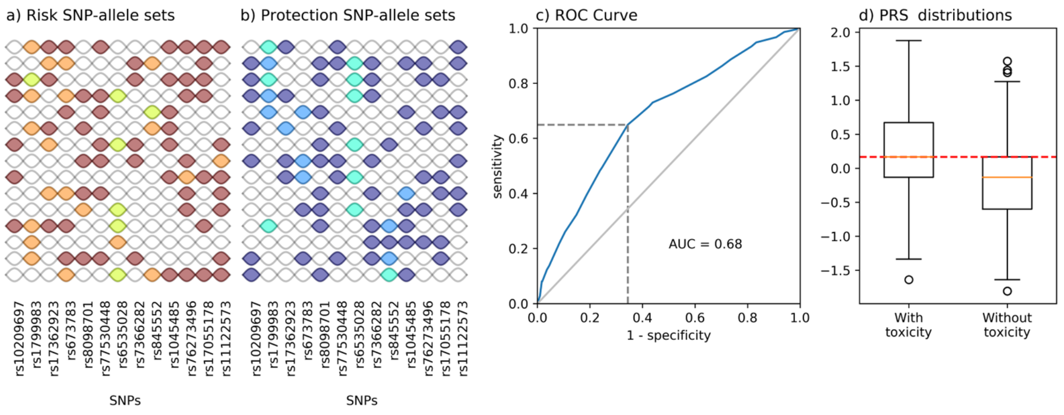


Figure 3.12: Splitted *hi*PRS for the **decreased stream** endpoint. Panels read as in Figure 3.9.

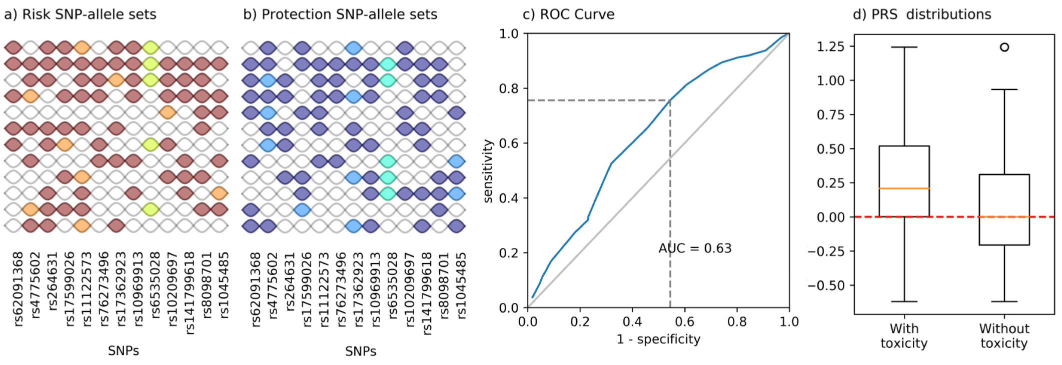


Figure 3.13: Splitted *hi*PRS for the **rectal bleeding** endpoint. Panels read as in Figure 3.9.

It is remarkable to see that, despite fitting way more coefficients with respect to our *hi*PRS implementation, the benchmark model performed consistently worse. In particular, the urinary toxicities concerning frequency variation, blood presence and decreased stream flow, appear to be those were epistatic effects are more relevant. Of note, here, the benchmark model features $d = 40$ learnable parameters. In general, fitting too many coefficients can result in biased estimates if the data at hand are not sufficiently numerous (for instance, some authors consider $d \geq n/5$ as a rule of thumb, cf. [189]). Clearly, such bias becomes even more problematic when the model is used for prediction over new patients.

To further investigate the ability of the proposed genetic score in stratifying patients with/without toxicity, we have also compared the score distribution within the two populations, see Table 3.3. In general, the two classes of patients report statistically significant differences in terms of median values and overall score distribution, with p-values always below 10^{-3}. In light of these promising results, we also tested whether classical NTCP models could benefit from the introduction of the proposed polygenic risk score. We discuss this latter analysis below.

3.5.2 PRS integration in NTCP models

Continuing our analysis on the prostate cancer cohort, we now extend the use of our PRS to general NTCP models. As we discussed back in Chapter 1, the latter are a standard tool for the estimation of late toxicity. Clinicians use NTCP models to predict the probability of late toxicity on the basis of individual characteristics (e.g. age, previous surgeries) and treatment details (e.g. radiation dose). Our purpose is to see whether the inclusion of our interaction-aware PRS can improve the performance of classical NTCP models.

		URI	HAE	NOC	DEC	REC
Clinical/dosimetric NTPC	(AUC)	0.64	0.66	0.66	0.56	0.57
Genetic NTPC	(AUC)	0.78	0.71	0.61	0.68	0.63
Combined NTPC	(AUC)	0.83	0.75	0.70	0.68	0.63
p-value		< 0.0001	0.001	0.01	< 0.0001	0.014

Table 3.4: Comparison of the three NTCP models in terms of AUC performance (for each endpoint separately). The reported p-value refers to the statistical test assessing whether the introduction of the PRS actually increased the AUC of the original clinico-dosimetric model.

To this end, for each of the five endpoints, we fit a corresponding logistic regression model of the form

$$\text{logit}\mathbb{P}(Y_j = 1) = \gamma_0 + \sum_{i=1}^{q_j} \alpha_{i,j} X_{i,j} + \beta_j \text{PRS}_j, \tag{3.11}$$

where, Y_j is the jth toxicity outcome, PRS_j is the polygenic risk score constructed at the previous step (thus with the Splitted *hi*PRS approach), while $X_{i,j}$ are the clinical/dosimetric covariates. In general, the latter are different for each toxicity endpoint, as we defined them according to the existing literature: see Table 3.5 at the end of this Section for a detailed description. To assess the added value of the genetic score, we fit (3.11) in three different ways: i) by forcing $\beta_j \equiv 0$ (clinical/dosimetric model), ii) by letting all $\alpha_{i,j} \equiv 0$ (genetic model), iii) without imposing any constraint (integrated NTCP model). Internal validation was performed using bootstrapping (10000 resamples).

Results are in Table 3.4 and Figure 3.14. As shown from the p-values in Table 3.4, the contribution of the PRS is always reflected in a statistically significant increase in AUC performance, especially for urinary toxicites. Additionally, for nearly all endpoints except the nocturia one, classical clinical/dosimetric models tend to perform worse with respect to purely genetic NTCPs. Of note, the most relevant predictor in the nocturia models appears to be the baseline covariate: as this endpoint was defined using patients questionnaires (cf. Chapter 1, Section 1.3), this might indicate an intrinsically poor tracking of the outcome. Nonetheless, for the other toxicities, the aforementioned increase in AUC can also be appreciated in Figure 3.14, with the genetic approach nearly always dominating the clinical NTCP model.

These promising results show that, if genetic information is included in a suitable way, it is actually possible to improve clinical/dosimetric NTCP models, paving the way to the harmonious combination of the two. Finally, we note how these integrated models can suggest ways for optimizing the treatment planning of radiotherapy. For instance, Figure 3.15 shows how the radiation dose impacts on the probability of late toxicity for different values of the PRS. More precisely, the curves are plotted according to Equation (3.11), by fixing the PRS values and letting the dose change: there we see how changes in the radiation dose can be extremely relevant for patients with a higher genetic risk, while they have little effect on more resistant individuals (at least on the range of Gray values considered). We mention that Figure 3.15 does not report the risk curves for the nocturia outcome: in fact, our previous observations already raised doubts about the actual validity of the results obtained for that specific toxicity endpoint.

In practical applications, clinicians could then exploit these models to optimize radiation doses, for instance by choosing values that keep the probability of late toxicity under a suitable threshold while preserving the treatment efficiency (if possible).

For further details about this case study, the interested reader can refer to our recent publications, see [68, 165].

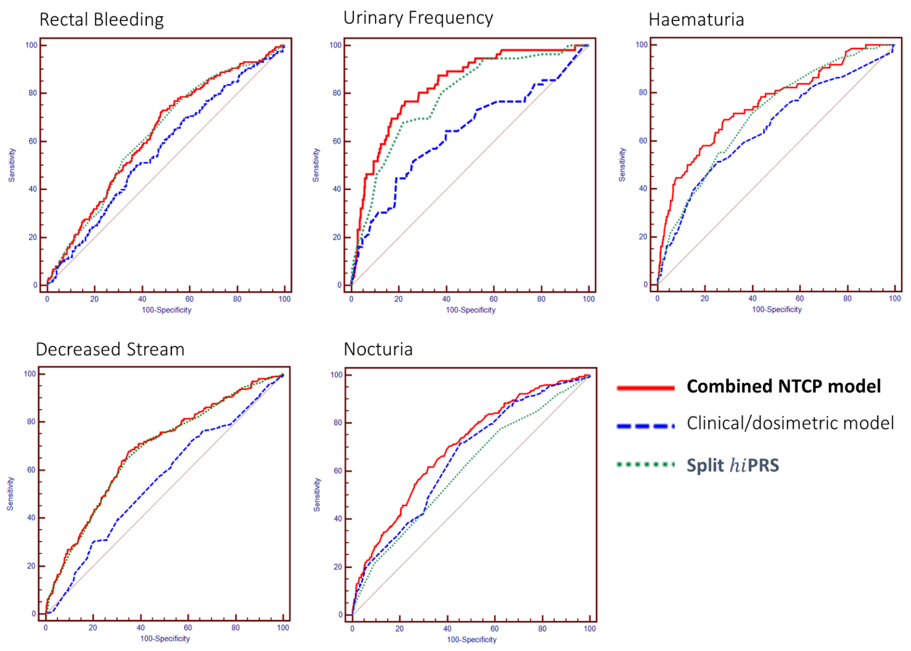


Figure 3.14: Comparison of different NTCP models for five late toxicity endpoints in prostate cancer patients treated with radiotherapy. Clinical/dosimetric models covariates are as in Table 3.5. Genetic models refer to our Splitted *hi*PRS approach. Combined model = Clinical/dosimetric model with an additional covariate given by the proposed genetic score (Splitted *hi*PRS approach).

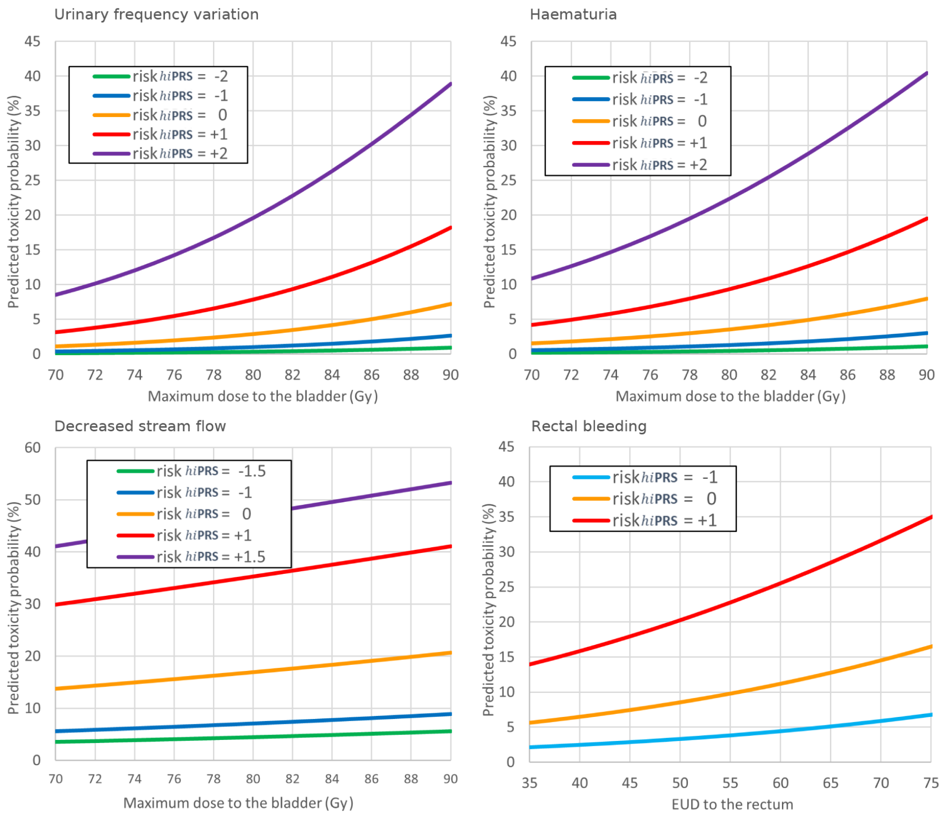


Figure 3.15: Radiation dose optimization based on individual genetic characteristics. The curves reported are slices of the logistic regression models obtained for different values of the polygenic risk score and varying radiation dose (other clinical variables are fixed to default values).

	URI	HAE	NOC	DEC	REC
Rectum equivalent uniform dose					✓
Bladder maximum dose	✓	✓	✓	✓	
Baseline symptoms	✓		✓		
Diabetes	✓		✓		
Smoking habit				✓	
Prostatectomy	✓	✓			
TURP			✓		

Table 3.5: Clinical/dosimetric covariates for the NTCP models of the five toxicity endpoints (URI = Urinary frequency variation, HAE = Haematuria, NOC = Nocturia, DEC = Decreased stream flow, REC = Rectal bleeding). Rectum equivalent uniform dose = daily radiation dose to the rectum (in Grays), converted in equivalent dose according to the linear-quadratic model, see e.g. [31]. Bladder maximum dose = maximum dose received by the bladder during each treatment (in Grays), converted in equivalent dose according to the linear-quadratic model. Baseline symptoms = binary variable indicating whether a mild form of the given toxicity was already observed before treatment. Diabetes = binary variable (yes vs no). Smoking habit = binary variable (equals "no" if and only if the patient never smoked or quitted smoking before being diagnosed with cancer). Prostatectomy, TURP (TransUrethral Resection of the Prostate) = binary variables encoding whether the patient underwent the specified surgical operation. Checkmarks indicate whether a given clinical/dosimetric covariate was included or not in the corresponding NTCP model.

3.6 Conclusions

In this Chapter we presented *hi*PRS, a novel approach to polygenic risk scoring that captures and models the effect on the phenotype of single SNPs and SNP-SNP interactions of potentially very high order. The algorithm takes individual-level genotype data as an input, overcoming potential biases of GWAS information, and providing a predictive yet easily interpretable tool. *hi*PRS allows the user to define the size of the model and the maximum order of the interactions to search for, which allows for reliable parameter estimations, especially for small samples, and convenient inspection by domain practitioners.

We have tested *hi*PRS against similar benchmark methods that rely on individual-level data, demonstrating its superior performance with respect to traditional PRSs and more complex Machine Learning based methods. In the presence of epistatic effects, *hi*PRS outperforms additive models, such as Lasso and Ridge, and yields comparable results to other state-of-the-art methods accounting for interactions, such as penalized PRSs (glinternet) and models based on artificial neural networks (DNN-Badre). Still, the interpretability of *hi*PRS also grants some additional insights, allowing users to gather information about the true generating mechanism of the phenotype. This is not to say that *hi*PRS always discovers biologically meaningful interactions, but its results can be considered hypothesis-generating, thus inspiring new experiments to evaluate epistasis at the biological level [48]. To evaluate the capability of *hi*PRS in tackling the complexities of real research scenarios, we also assessed the approach through an extensive set of simulations, showcasing its ability to deal with noise, strong class imbalance and small sample sizes.

Finally, we applied the algorithm to a clinical case study concerning radiotherapy late toxicity in prostate cancer patients. There, we showed how *hi*PRS can extract useful genetic information for boosting the performance of clinical/dosimetric NTCP models, with encouraging results for three out of five toxicity endpoints. This confirms the fact that radiosensitivity is a complex trait, influenced by both environmental and genetic factors, and that it cannot be explained by the segregation of single genes.

In the future, these integrated models may provide clinicians with suitable guidelines for the personalized treatment planning of radiotherapy, suggesting changes in the prescription dose or the use of specific aid devices like rectum spacers [200].

We conclude with two final remarks. The first one, is about the computational cost. Indeed, one major limitation of *hi*PRS lies in the scalability of the algorithm, especially when the number of SNPs grows dramatically. In fact, the search of high-order interactions is intrinsically expensive, and the computational cost of the mRMR selection routine grows quadratically with the number of candidate interactions. For this reason, our algorithm is better suited for those contexts where the number of SNPs is limited, as in clinical studies that start from literature-validated SNPs or that perform a preliminary selection of genetic variants (e.g., with the DSAEE approach in Chapter 2). There, the computational cost can be almost negligible: for instance, fitting *hi*PRS always took less than 5 seconds in our experiments.

As a final remark, instead, we point out that the proposed approach can flexibly accommodate any kind of categorical data, even though *hi*PRS was originally designed to process SNPs data. For instance, it might be effortlessly applied to other data describing genetic variants or epigenetic mutations encoded as binary variables, single and multi-level categorical clinical information, or their combination. This makes the *hi*PRS approach particularly interesting to tackle multi-omic studies, or to model the recognized interactions between genotype and environmental factors, or to investigate the mediating role of clinical conditions in determining the genotype effect on complex traits, widening the scope of applicability and relevance of our proposal.

Part 2:

Enabling real-time simulations for precision medicine with Deep Learning

4 | Mechanistic models for radiotherapy

We devote this fourth Chapter to the presentation of physics based models for radiotherapy. These are based on the relationship between the damage to the tissue caused by ionizing radiation and the presence of oxygen, a biological fact that is made mathematically rigorous through Tumor Control Probability (TCP) models. After having presented the fundamental ideas, we highlight some of the practical challenges, such as the elevated computational cost, that prevent a robust analysis of TCP models, including sensitivity analysis and uncertainty quantification. In response to these drawbacks, we then show how Deep Learning can enable the pursuit of these tasks, showcasing our results over a simplified model for oxygen transfer. Nonetheless, we postpone the detailed description (and derivation) of the corresponding Deep Learning algorithms to Chapters 5 and 7.

Tumors are abnormal masses that appear due to an excessive cell proliferation, usually caused by erroneous replications of the DNA. They are also known as *neoplasms* (literally "new formation" in ancient Greek), and they may be either benign or malignant, depending on whether they tend or not to continue their growth and spread all over the body. Tumors of this type are notoriously known as cancers, and they are characterized by an an uncontrolled proliferation that results in a constant invasion and destruction of the neighbouring healthy tissue. In particular, cancers consist of clonogenic cells, that is, cells with a genetic anomaly responsible for their ungoverned replication (*clonality*).

Because of the complicated way in which neoplasms growth, the tumoral microenvironment can be very intricated and difficult to describe. Nonetheless, tumors also share common characteristics, such as their multi-layer structure. Generally speaking, we may simplify the tumoral tissue as composed of three nested layers: the outer one is characterized by a good perfusion in terms of blood and oxygen supplies; middle layers, instead, are hypoxic in nature, meaning that they report extremely low oxygen concentrations, reaching the critical values of 8–10 mmHg in partial pressure [33]; finally, the last layer at the core is characterized by necrotic cells.

The eradication of clonogenic cells that populate malignant neoplasms, is one of the main purposes in oncological healthcare. To this end, one of the most well-established approaches is radiotherapy. As we discussed in the previous Chapters, radiotherapy is a destructive process that aims at killing cancer cells by depositing high levels of energy over their DNA strands, ultimately provoking either cellular apoptosis or non-apoptotic death [12]. Surprisingly, as first argued by JC Mottram [142] in 1936, the proportion of cells that survive a given radiation dose is not only dependent on the dose itself, but also depends on the local oxygen partial pressure, a phenomenon known as the *oxygen effect*. More precisely, those regions that are in hypoxic conditions tend to be more resistant to radiation, while the impact of radiotherapy is magnified in the presence of larger oxygen supplies.

In light of this, it is clear that understanding the perfusion of oxygen in biological tissues is of fundamental importance for the development of optimized radiotherapy treatments. Still, because of the small scales involved and the consequent impracticality in making real-life measurements, the study of this phenomenon has to rely on numerical simulations. As we shall discuss in a moment, accurate biophysiological models for oxygen transfer already exist and may be exploited for such tasks. However, high computational costs and uncertainty in the model parameters can significantly reduce the true potential of these approaches. Our purpose for this Chapter is to provide additional details on this topic and showcase how Deep Learning methods can help us in overcoming these drawbacks.

The Chapter is organized as follows. First, in Section 4.1, we describe a model for microvascular oxygen transfer, highlighting the presence of model parameters and their uncertainty. Then, in Section 4.2, we discuss the radiobiological model, where resistance to radiation is linked to oxygen deficiencies through the so-called linear-quadratic model, ultimately leading to Tumor Control Probability models (TCP) for the optimization of radiotherapy. Finally, in Section 4.3 we anticipate how Deep Learning can enable a robust analysis of TCP models, showcasing our results in a simplified scenario.

4.1 A mesoscale model for oxygen transfer

Cellular respiration is a fundamental biological process in which cells convert nutrients, such as sugar and amino acids, into energy resources stored in the form of adenosine triphosphate (ATP). These processes consist of a sequence of metabolic reactions that are commonly enabled by oxidizing agents such as oxygen molecules. In humans, cells receive oxygen supplies from the capillaries, small blood vessels of roughly 8-10 micrometers in diameter that are spread all over the body and that constitute the terminal part of the cardiocirculatory system. The purpose of oxygen transfer models is to capture this mechanism at the correct scales, providing a suitable description of both the vascular network and the tissue perfusion. For this reason, this phenomenon is typically modelled at the so-called *mesoscale* [177].

In this Section, we aim at briefly describing the mesoscale oxygen transfer model proposed by Possenti et al. in [160], with the intent of later using it as a starting point for radiotherapy applications. The model presented in [160] can be thought as a more accurate and general version of Krogh's model, which was originally developed in 1919 by danish physiologist August Krogh [111]. In order to properly state and comprehend its formulation, it is worth to introduce some preliminary notation.

We denote by $\Omega \subset \mathbb{R}^3$ a bounded domain representing the biological tissue. In Krogh's model, capillaries are represented as concentric cylinders with a suitable diameter. In [160], instead, they are approximated with 1-dimensional piecewise linear manifolds representing their centerline, which significantly simplifies the problem at the computational level while also allowing for more complicated geometries, cf. [51, 117]. We denote by $\Lambda \subset \Omega$ the 1D vascular graph. To preserve some additional information about the capillaries and their permeability, a scalar quantity $R > 0$ describing their radius is also defined. We let $C_t : \Omega \to \mathbb{R}$ and $C_v : \Omega \to \mathbb{R}$ be the oxygen concentration in the interstitial tissue and the vascular network, respectively. With this set up, the authors in [160] model the microvascular oxygen transfer through the coupled 3D-1D model below,

$$\mathcal{O}: \begin{cases} \nabla \cdot (-D_t \nabla C_t + \mathbf{u}_t C_t) + V_{\max} \left(C_t + \alpha_t p_{m_{50}}\right)^{-1} C_t = \phi_{O_2} \delta_\Lambda & \text{in } \Omega \\ \pi R^2 \partial_s \left(-D_v \partial_s C_v + u_v C_v + u_v k_1 \left(C_v^\gamma + k_2\right)^{-1} C_v^\gamma H\right) = -\phi_{O_2} & \text{on } \Lambda \\ \phi_{O_2} = 2\pi R P_{O_2} (C_v - \overline{C}_t) + \frac{1}{2}(1 - \sigma_{O_2}) \left(C_v + \overline{C}_t\right) \phi_v & \text{on } \Lambda \\ C_v = C_{\text{in}} & \text{on } \partial\Lambda_{\text{in}} \\ -D_v \partial_s C_v = 0 & \text{on } \partial\Lambda_{\text{out}} \\ -D_t \nabla C_t \cdot \mathbf{n} = \beta_{O_2} (C_t - c_{0,t}) & \text{on } \partial\Omega \end{cases} . \tag{4.1}$$

Fundamentally, the model consists of a stationary advection-diffusion equation coupled with suitable interface and boundary conditions. Here, $\partial\Lambda_{\text{in}}$ and $\partial\Lambda_{\text{out}}$ are used to describe the flow direction in the blood vessels, characterizing respectively points of inflow and outflow. The overline is used to denote average quantities, specifically, $\overline{C}_t(\mathbf{x})$ is the average value of C_t over the cross-section at the specified point in the blood vessel (thus, a circle of radius R centered at $\mathbf{x} \in \Lambda$). The parameters $D_t, \alpha_t, p_{m_{50}}, \phi_{\text{O}_2}, D_v, k_1, \gamma, k_2, P_{\text{O}_2}, \sigma_{\text{O}_2}, C_{\text{in}}, \beta_{\text{O}_2}, c_{0,t}$ are all given physical quantities: for a detailed description of these, we refer to Table 4.1.

Besides these quantities and the main unknowns, namely C_t and C_v, the oxygen transfer model $\mathcal{O}$ also features other physical terms that need to be specified. These are:

- δ_Λ, the singular measure describing the oxygen resources coming from the vascular graph. The latter is defined as the unique measure for which the following
$$\int_\Omega v(\mathbf{x})\delta_\Lambda(d\mathbf{x}) = \int_\Lambda v(\mathbf{s})d\mathbf{s} \tag{4.2}$$
holds for all $v \in \mathcal{C}(\Omega)$;

- $\mathbf{u}_t$ and u_v, respectively the interstitial and vascular fluid velocities. These are computed by solving the following system of equations, namely
$$\mathcal{F}: \begin{cases} \mathbf{u}_t + \frac{K}{\mu_t}\nabla p_t = 0 & \text{in } \Omega \\ \nabla\cdot\mathbf{u}_t + \phi_l - \phi_v\delta_\Lambda = 0 & \text{in } \Omega \\ 8\mu_v u_v + R^2\partial_s p_v = 0 & \text{on } \Lambda \\ \pi R^2\partial_s u_v + \phi_v = 0 & \text{on } \Lambda \end{cases},$$
in the unknowns $\mathbf{u}_t, u_v$ (fluid velocity) and p_t, p_v (fluid pressure, respectively in the tissue and the blood vessels). Here K, μ_t, μ_v, ϕ_l are given physical quantities (cf. Table 4.1), while
$$\phi_v := 2\pi R L_p\left((p_v - p_t) - \sigma(p_v - \overline{p}_t)\right)$$
is the exchange term between the two domains. As before, $\overline{p}_t$ indicates the cross-section average. The coefficients $L_p, \sigma, \pi_v, \pi_t$ are given (see Table 4.1);

- $H : \Lambda \to \mathbb{R}$, the discharge hematocrit, which is modeled as the solution to the conservation law below,
$$\mathcal{H}: \left\{\pi R^2 u_v\partial_s H - \phi_v H = 0 \quad \text{on } \Lambda.\right.$$
complemented with suitable boundary conditions, see [161].

In practice, once the model parameters have been set, problems $\mathcal{F}$ and $\mathcal{H}$ are solved sequentially to enable the solution of the oxygen transfer model $\mathcal{O}$. For additional details about the numerical discretization and solution of these systems we refer to [161, 160].

For our purposes, model $\mathcal{O}$ is interesting because of two main reasons. The first one, is that it provides an accurate description of microvascular oxygen transfer which has already been validated against physiological data, cf. [161, 160]. The second one, is that is allows for a flexible description of the vascular network and the overall micro-environment: in particular, it can be used to properly model the different layers of a tumoral region, and thus explore extensively the effects of radiotherapy in different scenarios. The link between model $\mathcal{O}$ and TCP models is discussed in the next Section.

Parameter	Description
Λ	Vascular network
R	Vessels radius
$V_{\max}$	Maximum oxygen consumption rate
D_t	Oxygen diffusion coefficient in interstitial tissue
D_t	Oxygen diffusion coefficient in blood vessels
α_t	Interstitium oxygen solubility
$p_{m_{50}}$	Oxygen partial pressure at half consumption rate
γ	Hill constant
k_1	Product of Hüfner's factor and mean corpuscolar hemoglobin concentrations
k_2	Product of plasma oxygen solubility and oxygen partial pressure at half saturation, raised to the γ power
P_{O_2}	Permeability of the vascular wall to oxygen
σ_{O_2}	Reflection coefficient relative to the oxygen molecule
C_{in}	Oxygen inflow
β_{O_2}	Boundary resistance simulating the presence of adjacent tissues
$c_{0,t}$	Far-field concentration
K	Permeability of the interstitial tissue
μ_t	Fluid viscosity in interstitial tissues
μ_v	Fluid viscosity in blood vessels
ϕ_l	Lymphatic drainage

Table 4.1: Physical and geometric parameters of the microvascular oxygen transfer model $\mathcal{O}$.

4.2 Tumor Control Probability (TCP) models

TCP models aim at estimating the probability of tumor eradication given physical and dosimetric factors. Among radiobiologists, the most famous model is arguably the *linear-quadratic* model, which describes cells survival in terms of radiation dose. More precisely, the latter states that the survival fraction S_f depends on the radiation dose D through the empirical law

$$S_f = e^{-\alpha D - \beta D^2}.$$

The parameters α and β are local coefficients that depend on physical properties of the tissue. In particular, they depend on the oxygen partial pressure O_2, which is a function of the spatial domain proportional to the oxygen concentration, according to Henry's law $O_2 \propto C_t$. Therefore, a better description of the survival fraction is given by the equation below

$$S_f(O_2, D) = \exp\left(-\alpha(O_2)D - \beta(O_2)D^2\right), \tag{4.3}$$

where

$$\alpha(O_2) = \frac{(a_1 + a_2\ell)\,O_2 + (a_3 + a_4\ell)\eta}{O_2 + \eta}, \qquad \beta(O_2) = \left(\frac{b_1 O_2 + b_3\eta}{O_2 + \eta}\right)^2.$$

Here, the coefficients a_i, b_i are suitable constants that have already been estimated in the literature. Conversely, ℓ is the so-called linear energy transfer, which depends on the type of radiation: for instance, X-rays are associated with lower values of ℓ compared to neutron beams radiations. Finally, η represents the oxygen partial pressure at which the relative radiosensitivity equals half of its maximum.

Equation (4.3) models the survival fraction as a map $\mathbf{x} \to S_f(\mathrm{O}_2(\mathbf{x}), D)$, that associates to each point $\mathbf{x} \in \Omega$ the local proportion of cells surviving radiation. In the case of tumor control, the ultimate goal is to eradicate all the clonogenic cells present at the points $\mathbf{x} \in \Omega$. To estimate the overall probability of success (in other words, the TCP), a classical approach is to describe the phenomenon in terms of Poisson point processes, leveraging on the assumption of unicellular independence [187]. The idea goes as follows. We let $\mathscr{N}$ be an inhomogeneous Poisson point process defined over Ω, such that for all measurable subsets $A \subseteq \Omega$, $\mathscr{N}(A)$ is the (random) number of clonogenic cells in A surviving a given radiation dose. We prescribe the intensity $\lambda : \Omega \to [0, +\infty)$ of such process as $\lambda(\mathbf{x}) = N(\mathbf{x})S_f(\mathrm{O}_2(\mathbf{x}), D)$, where $N(\mathbf{x})$ is the density of clonogenic at the point $\mathbf{x} \in \Omega$. Then, given a suitable radiation dose D, the global probability of tumor eradication is $\mathrm{TCP}(D) = \mathbb{P}(\mathscr{N} = 0)$, that is

$$\mathrm{TCP}(D) = \exp\left(-\int_{\Omega} N(\mathbf{x})S_f(\mathrm{O}_2(\mathbf{x}), D)d\mathbf{x}\right). \tag{4.4}$$

Equation (4.4) is of high interest as it can be used to optimize radiation doses. In particular, given the characteristics of the tumoral microenvironment, such as the distribution of oxygen partial pressure, we can compute the minimum value of D for which the probability of successful treatment remains above a given threshold.

Nonetheless, there are some limitations. For an accurate computation of the TCP, Equation (4.4) requires a precise description of the scalar field O_2. As we mentioned before, this is only manageable through the numerical simulation of mathematical models such as $\mathcal{O}$, Equation (4.1). However, while these models may be biologically accurate, they are limited by our uncertainty about the actual structure of the tumoral microenvironment.

For instance, if we want to analyze the case of a poorly vascularized tissue, we may solve system (4.1) for a sufficiently sparse vascular network $\Lambda \subset \Omega$. This will yield some solution for the oxygen partial pressure, $\mathrm{O}_2(\Lambda)$ for which we can evaluate the TCP. However, our choice of Λ is completely arbitrary: in fact, even if we prescribe the graph density, there are still uncountably many possible topologies for the vascular network. This uncertainty in the actual structure of Λ is reflected in our approximation for O_2 and, consequently, in the value of the TCP. Similar issues also are also encountered for the other parameters in (4.1): in fact, even though they are scalar values, they might be subject to random fluctuations around the actual physiological (or pathological) values, generating additional sources of uncertainty that propagate to the final TCP computation.

Consequently, a robust investigation of TCP models requires multiple runs of the numerical solver and results in a so-called many-query scenario. In particular, this can be an extremely demanding task because of the non negligible computational cost entailed by each numerical simulation: as soon as we change the parameter values, the PDE model has to be solved once again. Also, there is nearly nothing that we can precompute to save up computational time, as some parameters, such as the topology of the vascular network Λ, can completely change the properties of system.

Within this Thesis, our objective is to overcome this computational bottleneck by developing suitable model reduction strategies based on Deep Learning algorithms. It will be a long journey, distributed across the next three Chapters. However, our efforts will be completely repaid, as we will then have a whole new spectrum of reduction strategies at our disposal.

Even though our methodologies are developed and presented in the remainder of the dissertation, in the next Section we shall provide a brief anticipation on the potential of our approaches, showcasing how Deep Learning can help us in the study of TCP models.

4.3 Deep Learning for radiotherapy optimization

To showcase how we plan to use Deep Learning for the study of TCP models, in this Section we provide a brief anticipation on the results that we obtained limitedly to a simplified model for oxygen transfer. In the Thesis, the latter is addressed at the end of Chapter 7, in Section 7.3. The purpose of this analysis is to quantify how uncertainties in the vascular network can affect TCP values. To this end, we consider the following ideal setting. We denote by $\Omega := \{\mathbf{x} \in \mathbb{R}^2,\ |\mathbf{x}| < 1\}$ the open unit disk, representing an idealized bidimensional tissue. We model the oxygen partial pressure $u : \Omega \to \mathbb{R}$ through the differential model below,

$$\begin{cases} -\alpha\Delta u + u = (1-u)C(\Lambda)\delta_\Lambda & \text{in } \Omega \\ -\alpha\nabla u \cdot \mathbf{n} = \beta u & \text{on } \partial\Omega \end{cases} \tag{4.5}$$

where $\alpha = 0.1$ and $\beta = 0.01$ are a fixed diffusion and resistance coefficient, respectively, $\Lambda \subset \Omega$ is the vascular network, δ_Λ is defined as in (4.2), and

$$C(\Lambda) = \frac{1}{|\Lambda|},$$

where $|\Lambda|$ is the total length of the vascular graph. This normalization is introduced to ensure that all vascular networks provide the same oxygen supply: in this way, the perfusion of the tissue is only affected by the morphology of the graph.

We focus our attention on the following quantities of interest,

- The proportion of the tissue that is exposed to low oxygen supply, modeled as

$$V := \frac{1}{|\Omega|}\left|\{u < \varepsilon\}\right|,$$

  representing the size of the region under hypoxia. Here, $|A|$ is the Lebesgue measure of the set A, while we set the pressure threshold to $\varepsilon = 0.1$.

- The TCP for a given radiation dose, here fixed to be $D = 10$, computed according to Equation (4.4).

We remark that system (4.5) is an extremely simplified model for oxygen transfer, and, in particular, all scalar quantities are dimensionless. As a consequence, we need to reset the values for the parameters appearing in the formulae of the survival fraction and the TCP (namely, a_i, b_i, η, ℓ and N, which we assume to be uniform over Ω, coherently with the simplified dimensionless setting).

Now, our objective is to quantify the effect of Λ over these quantities of interest. In order to explore properly the possible scenarios and model the presence of uncertainties we proceed as follows. We use a macroscale parameter, $\lambda > 0$, to indicate the expected level of vascularization in the interstitial tissue. Then, given λ, we generate the vascular network Λ randomly, exploiting a suitable Poisson point process and Voronoi diagrams (for further details, we refer to Chapter 7, Section 7.3). Figure 4.1 shows the example of two vascular graphs and the corresponding solutions.

This way of modeling uncertainties turns our quantities of interest, V and TCP, into random variables, whose distribution changes together with the macroscale parameter λ. Higher values of λ correspond to denser graphs and thus higher TCPs: however, even if we fix λ, the values of the TCP will still be subject to random fluctuations because of the uncertainty in the vascular graph. The quantification of these fluctuations is what the current literature about TCPs lacks. Some authors conjecture that these effects might have little relevance under certain circumstances [75], while others believe that unraveling the causal relationship between tumor vascular structure and tissue oxygenation will pave the way for new personalized therapeutic approaches [21].

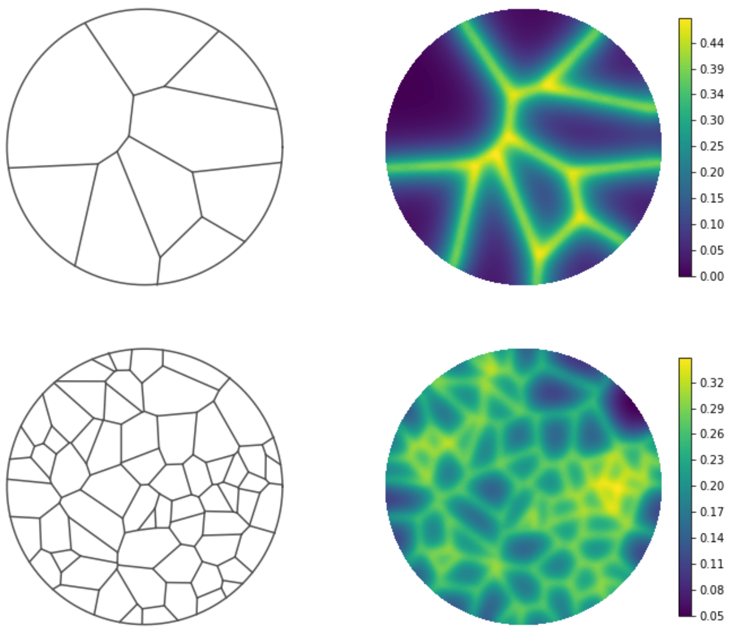


Figure 4.1: Two vascular networks of different density (left) and the corresponding distributions of oxygen (right).

To achieve our objective, many numerical simulations of Equation (4.5) are required, as we need to explore the range of possible solutions for many different configurations of the vascular network. To overcome this computational bottleneck, we replace the original numerical solver with a suitable surrogate given by a neural network architecture Φ. The latter, given the vascular network Λ, provides an approximation of the whole solution field, i.e.

$$\Phi(\Lambda) \approx u_\Lambda$$

where u_Λ is the solution to (4.5). The neural network model is built using the tools developed later in the Thesis, specifically in Chapter 7, and is trained on 4500 trusted samples generated with the original solver. After training, the network was able to approximate new solutions, for unseen network topologies, with an average L^2 error of 3.89%.

We used the neural network model to explore the variability of the oxygen distribution over $100'000$ new configurations, ultimately yielding the results in Figure 4.2. There, we see how the expected volume under hypoxia, $\mathbb{E}[V]$, decays exponentially with the vascularization level λ. The random fluctuations, represented by the inter-quantile and inter-decile bands, become smaller and smaller for increasing values of λ. For the TCP we see a similar behavior: the uncertainties tend to vanish for highly vascularized tissues and the average probability of tumor eradication saturates at 1 exponentially fast. Still, we point that, even in poorly vascularized regions, the TCP can reach values near 90%: it is a rare event, as shown by the inter-decile bands, but it can still occur.

Despite their simplicity, these results show how Deep Learning enables a completely new understanding of these models. Moreover, the proposed approach is completely general and thus provides a remarkable potential. We observe in fact that the neural network model is providing a global approximation of the whole solution field u. In particular, once trained, the same surrogate model can be employed for many different tasks and analysis. This would have not been possible if we only focused on the direct correspondence between the input parameters and the scalar quantities of interest. These promising results are the main motivation for our research in this field, which we shall report in the next Chapters.

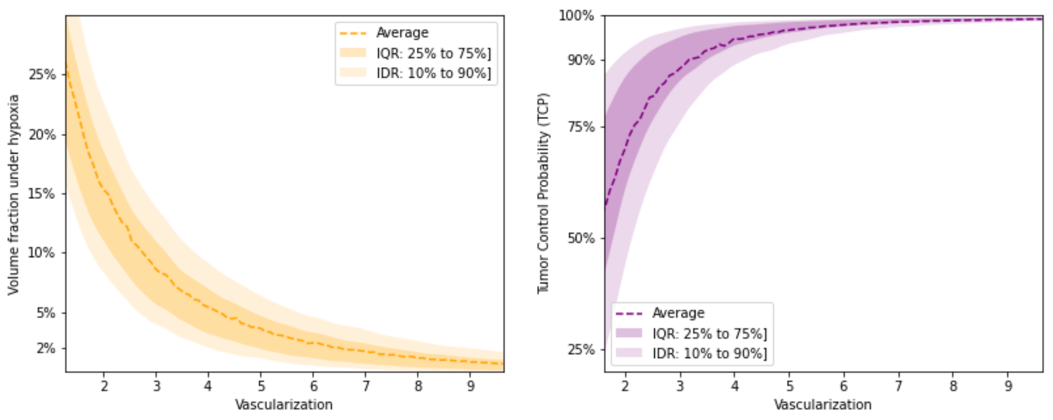


Figure 4.2: Volume fraction under hypoxia (left) and TCP at fixed radiation dose (right) for different levels of tissue vascularization. Dashed lines correspond to average values. IQR = Inter Quantile Range, IDR = Inter Decile Range.

Remark. In this example, and more in general, the key point is to have an efficient approximation of the parameter-to-solution map, $\Lambda \to u_\Lambda$. This problem is at the core of a research field known as Reduced Order Modeling (ROM), which features a very active and diverse literature. In particular, it may be worth to further motivate why we decided to employ Deep Learning algorithms to achieve our goal, considering the uncountable alternatives available in the literature. As we shall discuss in Chapter 5, the main motivation lies in the limitations of classical ROMs, which are notoriously based on linear projection techniques. In short, these approaches can encounter significant difficulties when dealing with space-localized parameters and singularities (see, e.g., Example 5.1 in Chapter 5). These, however, are two fundamental features of our case study: in fact, the dependency of the solution on the blood vessels is purely geometrical, and the vascular graph Λ enters the PDE model through a singular measure with 1D support. While linear techniques can hardly tackle these problems, nonlinear methods based on deep neural networks, instead, have the potential of providing an effective practical solution.

5 | A Deep Learning approach to Reduced Order Modeling of PDEs

In this fifth Chapter, we present a Deep Learning approach for the approximation of the parameter-to-solution map in the context of parametrized PDE models. In the same spirit of other works in the literature, see e.g. [70], our approach is based on the use of autoencoders and ultimately results in the construction of a nonintrusive surrogate model. However, with respect to the existing literature, our presentation is enriched with a corresponding mathematical analysis, providing rigorous results about the choice of the latent dimension and the complexity of the neural networks involved. These findings have been recently reported in Franco et al., "A Deep Learning approach to Reduced Order Modeling of parameter dependent Partial Differential Equations", Mathematics of Computation [65]. At the end of the Chapter, we also present a simplified application to oxygen transfer models, discussing its advantages and limitations.

In many areas of science, such as physics, biology and engineering, phenomena are modeled in terms of Partial Differential Equations (PDEs) that exhibit dependence on one or multiple parameters. As an example, consider the stationary advection-diffusion equation below,

$$\begin{cases} -\nabla\cdot(\boldsymbol{\sigma}_{\boldsymbol{\mu}}\nabla u) + \boldsymbol{b}_{\boldsymbol{\mu}}\cdot\nabla u = f_{\boldsymbol{\mu}} & \text{in } \Omega, \\ u = g_{\boldsymbol{\mu}} & \text{on } \partial\Omega, \end{cases} \tag{5.1}$$

where $\Omega\subset\mathbb{R}^d$ is a bounded domain and $\boldsymbol{\mu}$ a vector parameter taking values in a suitable parameter space $\Theta\subset\mathbb{R}^p$. For each $\boldsymbol{\mu}\in\Theta$, we assume the above to admit a unique solution $u_{\boldsymbol{\mu}}$, to be sought within a given Hilbert space $(V,\|\cdot\|)$. Equation (5.1) can be thought as a prototype problem for more comprehensive clinical models, such as the oxygen transfer model in Section 4.1.

In some cases, one is not interested in computing the PDE solution for a single fixed $\boldsymbol{\mu}\in\Theta$, but rather for an ensemble of parameter values. In general, this corresponds to exploring the so-called *solution manifold* $\mathcal{S} := \{u_{\boldsymbol{\mu}}\}_{\boldsymbol{\mu}\in\Theta}$ [63, 116]. The map $\boldsymbol{\mu}\to u_{\boldsymbol{\mu}}$ is known under many equivalent names such as the *parametric map* [176], the parameter-to-state map [88] or the solution map [150]. Approximating the parametric map in a highly-efficient way is a challenging task that can be encountered in several contexts, from optimal control problems with parametric PDEs constraints [25] to multiscale fluid mechanics [103], or Bayesian inversion and uncertainty quantification [32]. In all these cases, the main drawback is represented by the computational cost entailed by traditional PDE solvers. In fact, despite their accuracy, each query of a numerical scheme such as the Finite Element Method (FEM) implies a computational cost that easily becomes unsustainable in many query applications, where computations are supposed to be carried out within a very short amount of time.

One possibility is then to replace Full Order Models (FOMs) with cheaper surrogate models, namely Reduced Order Models (ROMs). ROMs originate from the need of alleviating the computational burden of FOMs at the price of a negligible compromise in terms of accuracy. During the last decades, several successful model reduction techniques have been developed, such as the Reduced Basis method [162] and others. However, the majority of these ROMs heavily relies on linear projection techniques, thus limiting significantly the spectrum of possible applications. Indeed, ROMs based on linear reduction methods encounter substantial difficulties whenever the solution manifold has a so-called Kolmogorov n-width [105] that decays slowly with n. The Kolmogorov n-width is a quantity that measures the degree of accuracy by which a set can be approximated using linear subspaces of dimension n, namely

$$d_n(\mathcal{S}) := \inf_{\substack{V_n\subset V,\\ \dim(V_n)=n}} \sup_{u\in\mathcal{S}} \inf_{\mathbf{v}\in V_n} \|u-v\|. \tag{5.2}$$

If $d_n(\mathcal{S})$ decays slowly with n, then projection-based ROMs can reach meaningful accuracies only for large values of n, which in turn leads to expensive models. We point out that this phenomenon is far from being uncommon. As a matter of fact, the slow decay on $d_n(\mathcal{S})$ is typical of time-dependent transport-dominated problems, even under fairly simple circumstances [74, 150]. The same is also true for stationary problems that are mostly diffusive, such as the oxygen transfer model in (4.1), as spatially localized parameters, singularties and nonlinearities can easily decelerate the decay of the Kolmogorov n-width. The interested reader can find a simple yet remarkable example of this fact at the end of Section 5.1, Example 5.1.

In order to tackle these drawbacks, we propose a novel approach based on Deep Neural Networks (DNNs) that naturally accounts for possible nonlinearities in the solution manifold. Our construction is mostly inspired by the recent advancements in nonlinear approximation theory, see e.g. [45, 52, 176], and the increasing use of deep learning techniques for parametrized PDEs and operator learning, as in [41, 70, 112, 126]. In particular, we focus on nonintrusive ROMs where the solution map is approximated by a deep neural network Φ. This idea has been recently investigated both theoretically, as in [112, 136, 176], and practically, e.g. [70, 72]. By now, the drawbacks posed by this approach are mainly practical: it is often unclear how the network architecture should be designed and which optimization strategies are better suited for the purpose. Also, we lack the understanding of the possible ways the nonlinearities in the DNN should be exploited in order to make the most out of it. Here, we wish to partially answer these questions and provide a constructive way of designing such Φ.

The key idea is to break the problem into two parts. First, we seek for a low-dimensional representation of the solution manifold, which we obtain by training a deep autoencoder [84], $\Psi\circ\Psi'$. The encoder, Ψ', is used to map the solution manifold into a reduced feature space $\mathbb{R}^n$, while the decoder serves for the reconstruction task. Here we see a clear analogy with the Nonlinear Kolmogorov n-width as defined in DeVore et al. [54]. There, the authors define

$$\delta_n(\mathcal{S}) := \inf_{\substack{\Psi'\in\mathcal{C}(\mathcal{S},\,\mathbb{R}^n)\\ \Psi\in\mathcal{C}(\mathbb{R}^n,\,V)}} \sup_{u\in\mathcal{S}} \|u-\Psi(\Psi'(u))\|,$$

as a nonlinear counterpart of $d_n(\mathcal{S})$. In light of this, we introduce the concept of *minimal latent dimension*, denoted as $n_{\min}(\mathcal{S})$, which we define as the smallest n for which $\delta_n(\mathcal{S})=0$. By choosing this particular $n_{\min}(\mathcal{S})$ as latent dimension for the autoencoder, we are then able to perform a significant model reduction.

Once the autoencoder has been trained, we exploit the encoder Ψ' in order to represent each solution $u_{\boldsymbol{\mu}}$ through a low-dimensional vector $\mathbf{u}^n_{\boldsymbol{\mu}}\in\mathbb{R}^n$. We then train a third network $\phi:\Theta\to\mathbb{R}^n$ to learn the *reduced map* $\boldsymbol{\mu}\to\mathbf{u}^n_{\boldsymbol{\mu}}$. In this way, by connecting the architectures of ϕ and Ψ we obtain the complete model, $\Phi:=\Psi\circ\phi$, which we later term as DL-ROM (Deep Learning based Reduced Order Model, in the same spirit of previous works [70, 71]).

The novelty of our contribution is twofold. First, we develop a new constructive way of using neural networks to approximate the solution map and we test it on some numerical examples. Second, we prove theoretical results that motivate the choice of the ROM dimension. Indeed, despite the popularity of autoencoders, e.g. [70, 118, 143, 207], the choice of the latent dimension n is often handled by trial and error. In contrast, we establish precise bounds on n thanks to a rigorous theoretical analysis. More precisely, in Theorems 5.1 and 5.2, we investigate the link between the minimal latent dimension $n_{\min}(\mathcal{S})$ and the topological properties of $\mathcal{S}$. In Theorem 5.3 we explicitly bound $n_{\min}(\mathcal{S})$ in terms of the dimensionality of the parameter space. In particular, we show that $n_{\min}(\mathcal{S}) \leq 2p+1$ as soon as the parametric map is Lipschitz continuous. The theory is then applied to the case of second order elliptic PDEs, in Theorem 5.4, where we demonstrate how the parameters directly affect the value of the minimal latent dimension. Finally, in Theorem 5.5, we bound the model complexity in terms of the ROM accuracy, deriving suitable error estimates that are later confirmed experimentally.

The remainder of the Chapter is organized as follows. In Section 5.1 we introduce our general framework and briefly recall the driving ideas of linear reduction. In Section 5.2 we move to the nonlinear case, where we establish a solid theoretical background for the construction of the DL-ROM, with particular emphasis on minimal representations and parametrized PDEs. In Section 5.3 we dive into the details of our deep learning approach, thereby discussing the general construction and its numerical properties. In Section 5.4 we present some numerical results and assess the proposed methodology, while in Section 5.5 we show a first application on a prototype problem of oxygen transfer. Finally, to keep the Chapter self-contained, auxiliary mathematical results are reported at the end, in a dedicated Section.

5.1 General setting

Within the present Section we formally introduce the problem of reduced order modelling for parametrized PDEs. For later comparison, we also take the chance to recall the linear reduction technique known as Principal Orthogonal Decomposition [127, 162]. In the remainder of the Chapter, we make use of elementary notions coming from the areas of Functional Analysis, Numerical Analysis and Topology. We respectively refer to [2, 60], [163] and [86].

5.1.1 Reduced Order Models for parametrized PDEs

We are given a parameter space $\Theta \subset \mathbb{R}^p$, a Hilbert state space $(V, \|\cdot\|)$ and parameter dependent operators $a_{\boldsymbol{\mu}} : V \times V \to \mathbb{R}$ and $f_{\boldsymbol{\mu}} : V \to \mathbb{R}$. For each $\boldsymbol{\mu} \in \Theta$ we consider the variational problem

$$u \in V: \quad a_{\boldsymbol{\mu}}(u, v) = f_{\boldsymbol{\mu}}(v) \quad \forall v \in V. \tag{5.3}$$

We assume the problem to be well-posed, so that for each $\boldsymbol{\mu} \in \Theta$ there exists a unique solution $u = u_{\boldsymbol{\mu}} \in V$. Our interest is to define a ROM that is able to approximate the parametric map $\boldsymbol{\mu} \to u_{\boldsymbol{\mu}}$ efficiently. In general, the workflow goes as follows. First, one chooses a FOM, which we here assume to be based on Galerkin projections. This corresponds to fixing a so-called *high-fidelity discretization*, that is a finite dimensional subspace $V_h \subset V$, $\dim(V_h) = N_h$, used to replace the original trial space. Having chosen a basis for V_h, say $\{\varphi_i\}_{i=1}^{N_h}$, for each $\boldsymbol{\mu} \in \Theta$ one turns equation (5.3) into the (discrete) problem

$$\mathbf{u}_{\boldsymbol{\mu}}^h = [\mathrm{u}_{\boldsymbol{\mu},1}^h, \dots, \mathrm{u}_{\boldsymbol{\mu},N_h}^h] \in \mathbb{R}^{N_h}: \quad a_{\boldsymbol{\mu}}\left(\sum_{i=1}^{n} \mathrm{u}_{\boldsymbol{\mu},i}^h \varphi_i,\ v\right) = f_{\boldsymbol{\mu}}(v) \quad \forall v \in V_h. \tag{5.4}$$

The main purpose of the high-fidelity discretization is to reframe the original problem within a finite dimensional setting, without particular care on the computational cost (for now). Regarding the choice of V_h, we make the following assumption.

Assumption 5.1. *For any $\varepsilon > 0$ there exists $V_h := \operatorname{span}\{\varphi_i\}_i^{N_h} \subset V$ such that*

$$\sup_{\boldsymbol{\mu}\in\Theta} \left\| u_{\boldsymbol{\mu}} - \sum_{i=1}^{N_h} \mathrm{u}^h_{\boldsymbol{\mu},i}\varphi_i \right\| < \varepsilon$$

that is, the FOM accuracy can be bounded independently on the value of $\boldsymbol{\mu} \in \Theta$.

The above is a very common assumption in the literature, see e.g. [112], that allows us to formally replace V with V_h. The objective now becomes that of learning the map $\boldsymbol{\mu} \to \mathbf{u}^h_{\boldsymbol{\mu}}$ in a way that reduces the FOM cost. In particular, the construction of the ROM consists in finding a suitable map $\Phi : \mathbb{R}^p \to \mathbb{R}^{N_h}$ for which $\Phi(\boldsymbol{\mu}) \approx \mathbf{u}^h_{\boldsymbol{\mu}}$. To do so, the common practice is to make extensive use of the FOM during a preliminary offline stage, which results in the collection of the so-called *snapshots*, $\{\boldsymbol{\mu}_i, \mathbf{u}^h_{\boldsymbol{\mu}_i}\}_i \subset \mathbb{R}^p \times \mathbb{R}^{N_h}$. These snapshots are then processed in order to build the ROM. In this sense, the identification of Φ can be seen as a problem of Statistical Learning, as argued in [112]. The way Φ is defined from the data is what characterizes each ROM, its efficiency and accuracy.

5.1.2 Methods based on linear projections

Many state-of-the-art ROMs are built upon the use of linear reduction techniques, which are known to work particularly well for second order elliptic PDEs with affine coefficients [9]. The idea is the following. Having fixed a high-fidelity discretization, one considers the (discretized) solution manifold $\mathcal{S}^h := \{\mathbf{u}^h_{\boldsymbol{\mu}}\}_{\boldsymbol{\mu}\in\Theta}$ and tries to approximate it using linear subspaces. This translates into fixing a reduced dimension $n \in \mathbb{N}$ and searching for the orthonormal matrix $\mathbf{V} \in \mathbb{R}^{N_h\times n}$ that minimizes the errors $\|\mathbf{u}^h_{\boldsymbol{\mu}} - \mathbf{V}\mathbf{V}^T\mathbf{u}^h_{\boldsymbol{\mu}}\|$. In practice, the identification of such $\mathbf{V}$ is done empirically by exploiting the aforementioned snapshots $\{\boldsymbol{\mu}_i, \mathbf{u}^h_{\boldsymbol{\mu}_i}\}_{i=1}^N$, which is often achieved through the so-called Principal Orthogonal Decomposition (POD). In short, this latter approach considers all the FOM snapshots as columns of a matrix, $\mathbf{U} := [\mathbf{u}^h_{\boldsymbol{\mu}_1}, , \mathbf{u}^h_{\boldsymbol{\mu}_N}]$ and computes its singular value decomposition

$$\mathbf{U} = \tilde{\mathbf{U}}\Sigma\mathbf{W}^T$$

where $\Sigma = \operatorname{diag}(\sigma_1, .., \sigma_N)$ with $\sigma_1 \geq \ldots \geq \sigma_N \geq 0$. Then, in the POD approach, $\mathbf{V}$ is defined by extracting the first n columns of $\tilde{\mathbf{U}}$. It is well-known that this choice of $\mathbf{V}$ is optimal -in some sense- over the training sample $\{\mathbf{u}^h_{\boldsymbol{\mu}_i}\}_{i=1}^N$. We also mention that, while this version of the POD considers $\mathbb{R}^{N_h}$ as a Euclidean space, slight modifications allow to account for different (e.g. energy) norms.

Once $\mathbf{V}$ has been built, the solution manifold is projected onto the reduced space $\mathbb{R}^n$, and each FOM solution is associated with the corresponding low-dimensional representation, $\mathbf{u}^n_{\boldsymbol{\mu}} := \mathbf{V}^T\mathbf{u}^h_{\boldsymbol{\mu}}$. To be operational, the ROM then needs to implement a suitable algorithm that approximates the correspondence $\boldsymbol{\mu} \to \mathbf{u}^n_{\boldsymbol{\mu}}$. If we represent the latter as a map $\phi : \Theta \to \mathbb{R}^n$, then the ROM approximation of high-fidelity solutions can be written as $\Phi(\boldsymbol{\mu}) := \mathbf{V}\phi(\boldsymbol{\mu}) \approx \mathbf{u}^h_{\boldsymbol{\mu}}$. Within the literature, this step has been handled in multiple ways. In the Reduced Basis method [116, 162], particularly in the so-called POD-Galerkin method, ϕ is defined intrusively by projecting and solving equation (5.4) onto $\operatorname{span}(\mathbf{V})$. Depending on the parametrization and on the type of PDE, this procedure may turn to be too expensive, which is why several alternatives have been proposed, see e.g. [13, 147, 180]. Nonintrusive approaches for defining ϕ include Gaussian process regression [80], polynomial chaos expansions [93], neural networks [41, 40] and others.

Nevertheless, because of the linear approximation, these ROMs encounter substantial difficulties as soon as $\mathcal{S}$ has a Kolmogorov n-width, see equation (5.2), that decays slowly. In fact,

$$\sup_{\boldsymbol{\mu}\in\Theta} \|\mathbf{u}^h_{\boldsymbol{\mu}} - \mathbf{V}\phi(\boldsymbol{\mu})\| \geq \sup_{\boldsymbol{\mu}\in\Theta} \|\mathbf{u}^h_{\boldsymbol{\mu}} - \mathbf{V}\mathbf{V}^T\mathbf{u}^h_{\boldsymbol{\mu}}\| \geq d_n(\mathcal{S}^h) \geq d_n(\mathcal{S}) - \varepsilon,$$

where $\varepsilon > 0$ is the accuracy of the high-fidelity discretization. Therefore, if $d_n(\mathcal{S})$ decays slowly, one may be forced to consider large values of n, which in turn makes ϕ more expensive and harder to identify. A didactic example of slow decay is shown below, Example 5.1. As we argue in the next Section, one possible solution to this problem is given by nonlinear reduction techniques. However, despite this being a promising direction, only a few steps have been made so far, e.g. [23, 70, 118].

Example 5.1. *Let $\Omega := (-2,2)$. For any $x_0 \in \Omega$, let δ_{x_0} be the Dirac delta distribution centered at x_0. Consider the 2-dimensional parameter space $\Theta := \{\boldsymbol{\mu} = (\mu_1,\mu_2) \in [-1,1] \times [0,1] \mid -1 \le \mu_1 - \mu_2 \le \mu_1 + \mu_2 \le 1\}$, together with the differential problem below*

$$\begin{cases} -u'' = 2\delta_{\mu_1} - \delta_{\mu_1-\mu_2} - \delta_{\mu_1+\mu_2} & x \in \Omega \\ u(-2) = u(2) = 0 \end{cases}$$

For each $\boldsymbol{\mu} \in \Theta$, the corresponding solution $u_{\boldsymbol{\mu}}$ is a piecewise linear function with support given by $[\mu_1 - \mu_2, \mu_1 + \mu_2]$. In particular, $u_{\boldsymbol{\mu}}$ is a hat function with a peak of height μ_2 at $x = \mu_1$. Also, by direct calculation,

$$\|u_{\boldsymbol{\mu}}\|_{L^2(\Omega)} = \sqrt{\frac{2}{3}\mu_2^3}.$$

Let now $\mathcal{S} := \{u_{\boldsymbol{\mu}}\}_{\boldsymbol{\mu}\in\Theta} \subset V := L^2(\Omega)$ and fix any positive $n \in \mathbb{N}$. It is then easy to see that the functions

$$v_{i,n} := u_{\left(-1+\frac{i}{n}-\frac{1}{2n},\frac{1}{2n}\right)}, \quad i = 1,\dots,2n$$

are mutually orthogonal in $L^2(\Omega)$. As a consequence, the Kolmogorov n-width of $\mathcal{S}$ satisfies

$$d_n(\mathcal{S}) \ge d_n(\{v_{i,n}\}_{i=1}^{2n}) =$$

$$= d_n\left(\left\{\|v_{i,n}\|_{L^2(\Omega)}^{-1} v_{i,n}\right\}_{i=1}^{2n}\right) \|v_{i,n}\|_{L^2(\Omega)} = \frac{1}{\sqrt{2}}\|v_{i,n}\|_{L^2(\Omega)} = \frac{1}{2\sqrt{6}} n^{-3/2},$$

where the second last equality follows by noticing that the set $\{\|v_{i,n}\|_{L^2(\Omega)}^{-1} v_{i,n}\}_{i=1}^{2n}$ is isometric to the canonical basis of $\mathbb{R}^{2n}$ (see [150]). Therefore, $d_n(\mathcal{S})$ decays with a rate of at most $n^{-3/2}$, which is relatively slow when compared to the ideal case where the parametric map is analytic and the Kolmogorov n-width is known to decay exponentially, $d_n(\mathcal{S}) \sim e^{-\gamma n}$.

5.2 Nonlinear dimensionality reduction

In the present Section we formalize the idea of using nonlinear reduction techniques for the compression of the solution manifold. We start by introducing all concepts and results in an abstract fashion. Only at the end, Section 5.2.2, we rephrase the content in terms of parametrized PDEs.

5.2.1 Minimal dimension and nonlinear Kolmogorov n-width

Within this Section, we consider an abstract setting where $(V, \|\cdot\|)$ is a Hilbert space and $\mathcal{S} \subset V$ a generic subset. In particular, $\mathcal{S}$ needs not to be the solution manifold of a parametrized PDE and the theory is presented regardless of a possible discretization. We address the problem of finding a low-dimensional representation of $\mathcal{S}$ while minimizing the reconstruction error.

When V is finite-dimensional, the linear reduction described in Section 5.1.2 performs an encoding of $\mathcal{S}$ via the map $\mathbf{u} \to \mathbf{V}^T\mathbf{u} =: \mathbf{u}^n \in \mathbb{R}^n$, where n is the reduced dimension; the set is then recovered through $\mathbf{u}^n \to \mathbf{V}\mathbf{u}^n \approx \mathbf{u}$. Therefore, a possible generalization to the nonlinear case is to substitute $\mathbf{V}^T$ with some *encoder* $\Psi' : \mathcal{S} \to \mathbb{R}^n$ and $\mathbf{V}$ with a *decoder* $\Psi : \mathbb{R}^n \to V$. Of note, this is also an approach that easily extends to infinite-dimensional settings. Depending on the restrictions that we impose on Ψ' and Ψ, different reconstruction accuracies can be obtained.

Here, we only require Ψ' and Ψ to be continuous. This, naturally gives rise to the optimization problem below,

$$\delta_n(\mathcal{S}) := \inf_{\substack{\Psi' \in \mathcal{C}(\mathcal{S}, \mathbb{R}^n) \\ \Psi \in \mathcal{C}(\mathbb{R}^n, V)}} \sup_{u \in \mathcal{S}} \|u - \Psi(\Psi'(u))\|, \tag{5.5}$$

where $\mathcal{C}(X, Y)$ denotes the collection of all continuous maps from X to Y. As we mentioned in the Introduction, the above corresponds to the (continuous) Nonlinear Kolmogorov n-width as defined in [54]. It is clear that $d_n(\mathcal{S}) \geq \delta_n(\mathcal{S})$. Also, $\delta_n(\mathcal{S})$ is nonincreasing in n, which reflects the fact that better approximations can be achieved in higher dimensional spaces. However, in the context of reduced order modelling, smaller values of n are often preferable, as they allow for less expensive models. In this sense, whenever there exists a smallest dimension $n_{\min}$ that allows for an arbitrarily accurate reduction, i.e. $\delta_{n_{\min}}(\mathcal{S}) = 0$, we may want to focus on that one. For this reason, we introduce the notation

$$n_{\min}(\mathcal{S}) := \min\{n \in \mathbb{N} \mid \delta_n(\mathcal{S}) = 0\},$$

were we adopt the convention $\min \varnothing = +\infty$. We refer to $n_{\min}(\mathcal{S})$ as to the minimal latent dimension of $\mathcal{S}$. Clearly, when $V \cong \mathbb{R}^{N_h}$ is finite-dimensional, the above definition is of interest only if $n_{\min}(\mathcal{S}) \ll N_h$. Nevertheless, as we will see below, this is always the case as soon as $\mathcal{S}$ has an intrinsic low-dimensional structure. Indeed, the value of $n_{\min}(\mathcal{S})$ is strongly related to the topological properties of $\mathcal{S}$. For instance, in the case of compact sets, it is invariant under bicontinuous transformations. More precisely, we have the following.

Theorem 5.1. *Let V and W be two Hilbert spaces. Let $\mathcal{S} \subset V$ and $\mathcal{M} \subset W$ be two compact subsets. If $\mathcal{S}$ and $\mathcal{M}$ are homeomorphic, then $n_{\min}(\mathcal{S}) = n_{\min}(\mathcal{M})$.*

Proof. Since the situation is symmetric in $\mathcal{S}$ and $\mathcal{M}$, it is sufficient to prove that $n_{\min}(\mathcal{S}) \geq n_{\min}(\mathcal{M})$. If $n_{\min}(\mathcal{S}) = +\infty$, the inequality is obvious. Hence, we assume there exists some $n \in \mathbb{N}$ for which $\delta_n(\mathcal{S}) = 0$. By definition of infimum, there exists a sequence of encoding-decoding pairs $\{(\Psi'_j, \Psi_j)\}_{j \geq 0}$ in $\mathcal{C}(\mathcal{S}, \mathbb{R}^n) \times \mathcal{C}(\mathbb{R}^n, V)$ such that $\sup_{u \in \mathcal{S}} \|u - \Psi_j(\Psi'_j(u))\|_V \to 0$ as $j \to +\infty$. Let now $\phi : \mathcal{S} \to \mathcal{M}$ be bicontinuous (recall that the sets are homeomorphic). Since $\mathcal{S}$ is compact, ϕ admits a uniformly continuous extension $\tilde{\phi} : V \to W$ (cf. Theorem 1.12 in [18]). We are then allowed to consider the continuous maps $\tilde{\Psi}_j := \tilde{\phi} \circ \Psi_j : \mathbb{R}^n \to W$ and $\tilde{\Psi}'_j := \Psi'_j \circ \phi^{-1} : \mathcal{M} \to \mathbb{R}^n$. For ω a monotone modulus of continuity of $\tilde{\phi}$, we have

$$\begin{aligned}
\delta_n(\mathcal{M}) &\leq \lim_{j \to +\infty} \sup_{m \in \mathcal{M}} \|m - \tilde{\Psi}_j(\tilde{\Psi}'_j(m))\|_W = \\
&= \lim_{j \to +\infty} \sup_{m \in \mathcal{M}} \|m - \tilde{\phi}(\Psi_j(\Psi'_j(\phi^{-1}(m))))\|_W = \\
&= \lim_{j \to +\infty} \sup_{s \in \mathcal{S}} \|\tilde{\phi}(s) - \tilde{\phi}(\Psi_j(\Psi'_j(s)))\|_W \leq \\
&\leq \lim_{j \to +\infty} \sup_{s \in \mathcal{S}} \omega\left(\|s - \Psi_j(\Psi'_j(s))\|_V\right) \leq \\
&\leq \lim_{j \to +\infty} \omega\left(\sup_{s \in \mathcal{S}} \|s - \Psi_j(\Psi'_j(s))\|_V\right) = 0,
\end{aligned}$$

as $\omega(h) \downarrow 0$ whenever $h \downarrow 0$. This proves that $\delta_n(\mathcal{S}) = 0 \implies \delta_n(\mathcal{M}) = 0$ and hence $n_{\min}(\mathcal{S}) \geq n_{\min}(\mathcal{M})$. □

The minimal latent dimension is also related to the so-called topological dimension, or Lebesgue covering dimension. For a formal definition of the latter we refer to [56, 58]. In particular, if $\mathcal{S}$ has an intrinsic p-dimensional structure, then we are able to bound $n_{\min}(\mathcal{S})$ explicitly. Indeed, by classical results of Dimension Theory, the following theorem holds true.

Theorem 5.2. *Let V be a Hilbert space and $\mathcal{S} \subset V$ a compact subset. If $\mathcal{S}$ has topological dimension p, then $n_{\min}(\mathcal{S}) \le 2p+1$, and the infimum appearing in (5.5) is attained at all reduced dimensions $n \ge 2p+1$. Additionally, if $\mathcal{S}$ is a topological p-manifold, then the lower bound $n_{\min}(\mathcal{S}) \ge p$ also holds.*

Proof. We shall prove that $\delta_{2p+1}(\mathcal{S}) = 0$, and that the infimum is attained. To this end, we notice that $\mathcal{S}$ is compact and thus separable. Therefore, by the Menger–Nöbeling embedding theorem (see Theorem 1.11.4 in [58]), there exists a subset $A \subset \mathbb{R}^{2p+1}$ and a bicontinuous map $\phi : \mathcal{S} \to A$. By continuity, the set A is compact. In particular, ϕ^{-1} admits a continuous extension $\Psi : A \to V$. The existence of such an extension can be argued as in the proof Theorem 5.1, or using other results such as Dugundji extension theorem [57], with the advantage of generalizing Theorem 5.2 to the case of normed spaces. Next, we define $\Psi' := \phi$. Then, the pair (Ψ', Ψ) agrees with the definition of Nonlinear Kolmogorov n-width and it also yields a perfect reconstruction of $\mathcal{S}$. The first claim in the theorem follows. Assume now that $\mathcal{S}$ is a p-manifold and let $n < p$. By definition, there exists a bicontinuous map ϕ from the closed unit ball $B := \{x \in \mathbb{R}^p,\ |x| \le 1\}$ to a certain compact subset $U \subseteq \mathcal{S}$. Let

$$m := \min_{|x|=1} \|\phi(x) - \phi(-x)\|.$$

Due to compactness, the minimum is attained and thus $m > 0$ (recall that ϕ is bijective). We now prove that $\delta_n(\mathcal{S}) \ge m/2$, and therefore $n_{\min}(\mathcal{S}) \ge p$. Let $\Psi' : \mathcal{S} \to \mathbb{R}^n$ and $\Psi : \mathbb{R}^n \to V$ be continuous. We note that the composition $\Psi' \circ \phi$ is continuous from $B \subset \mathbb{R}^p \to \mathbb{R}^n$. Therefore, as $n < p$, by the Borsuk-Ulam theorem [28] we are granted the existence of a point $x^* \in B$, $|x^*| = 1$, for which $\Psi'(\phi(x^*)) = \Psi'(\phi(-x^*)) =: z$. It follows that,

$$\sup_{v \in \mathcal{S}} \|v - \Psi(\Psi'(v))\| \ge \sup_{v \in U} \|v - \Psi(\Psi'(v))\| = \sup_{x \in B} \|\phi(x) - \Psi(\Psi'(\phi(x)))\| \ge$$

$$\ge \max\left\{\|\phi(x^*) - \Psi(\Psi'(\phi(x^*)))\|,\ \|\phi(-x^*) - \Psi(\Psi'(\phi(-x^*)))\|\right\} \ge$$

$$\ge \frac{1}{2}\|\phi(x^*) - \Psi(z)\| + \frac{1}{2}\|\phi(-x^*) - \Psi(z)\| \ge \frac{1}{2}\|\phi(x^*) - \phi(-x^*)\| \ge \frac{m}{2}. \quad \square$$

We mention that, in the particular case of p-manifolds and under suitable smoothness assumptions, the bounds in Theorem 5.2 can be sharpened to $n_{\min}(\mathcal{S}) \le 2p$ or even $n_{\min}(\mathcal{S}) \le 2p-1$ in case p is not a power of 2. These are all consequences of the so-called Whitney embedding theorem and a few of its variants. We do not dive deeper into the matter but leave [184] as a reference for the interested reader. We also note that the intrinsic dimension of $\mathcal{S}$ does not uniquely determine the value of $n_{\min}(\mathcal{S})$. In particular, $\mathcal{S}$ may have topological dimension p but $n_{\min}(\mathcal{S}) > p$, coherently with Theorem 5.2. In this respect, we report below two simple examples.

Example 5.2. *Let $V = \mathbb{R}^2$ and $\mathcal{S} = \{\mathbf{x} \in V : |\mathbf{x}| = 1\}$ be the unit circle. Then, $\mathcal{S}$ is a one-dimensional manifold but $\delta_1(\mathcal{S}) = 1$ and $n_{\min}(\mathcal{S}) = 2$. To see this, consider any pair of continuous maps $\Psi' : \mathcal{S} \to \mathbb{R}$ and $\Psi : \mathbb{R} \to \mathbb{R}^2$. By the Borsuk-Ulam theorem, there exists a point $\mathbf{x} \in \mathcal{S}$ such that $\Psi'(\mathbf{x}) = \Psi'(-\mathbf{x})$. Therefore, being $\|\cdot\| = |\cdot|$ the Euclidean norm,*

$$\sup_{\mathbf{v} \in \mathcal{S}} \|\mathbf{v} - \Psi(\Psi'(\mathbf{v}))\| \ge \max\{|\mathbf{x} - \Psi(\Psi'(\mathbf{x}))|,\ |-\mathbf{x} - \Psi(\Psi'(-\mathbf{x}))|\} \ge$$

$$\ge \frac{1}{2}\left(|x - \Psi(\Psi'(\mathbf{x}))| + |-\mathbf{x} - \Psi(\Psi'(-\mathbf{x}))|\right) \ge \frac{1}{2}|\mathbf{x} - (-\mathbf{x})| = 1.$$

As Ψ' and Ψ are arbitrary, we conclude that $\delta_1(\mathcal{S}) \ge 1$. The equality is then obtained by considering the case in which both Ψ' and Ψ are identically zero.

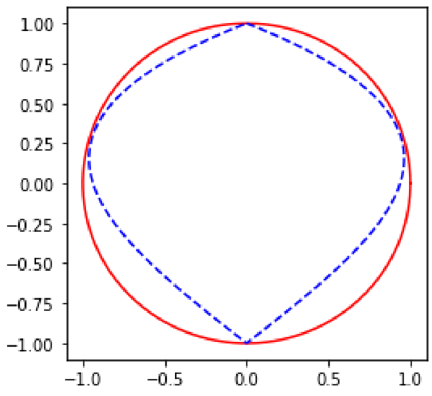


Figure 5.1: Reference picture for Example 5.3, Section 5.2.1. In blue (dashed line), the curve $\mu \to (20\mu^3 - 30\mu^2 + 10\mu,\ 2|1-2\mu| - 1)$, for $0 \le \mu \le 1$; in red (straight line), the unit circle. The two curves are clearly homeomorphic.

Example 5.3. *On the spatial domain $\Omega = (0,\pi)$, consider the boundary value problem*

$$\begin{cases} u'' = -u & x \in \Omega \\ u(0) = 10(2\mu^3 - 3\mu^2 + \mu) \\ u'(\pi) = 1 - 2|1-2\mu|, \end{cases}$$

where $\mu \in [0,1]$ is a parameter. Let us then consider the solution manifold $\mathcal{S} = \{u_\mu\}_{\mu\in[0,1]}$ as a subset of $V = L^2(\Omega)$. Then, $\mathcal{S}$ is a 1-dimensional manifold but its minimal latent dimension equals $n_{\min}(\mathcal{S}) = 2$. Indeed, explicitly expanding the solutions reads

$$u_\mu(x) = 10(2\mu^3 - 3\mu^2 + \mu)\cos x + (2|1-2\mu| - 1)\sin x.$$

It is then clear that, up to scaling of the L^2-norm, $\mathcal{S}$ can be isometrically identified with the curve $\mu \to (20\mu^3 - 30\mu^2 + 10\mu,\ 2|1-2\mu| - 1)$ in $\mathbb{R}^2$. But the latter curve is a compact manifold with positive nonlinear Kolmogorov 1-width, as it is homeomorphic to the unit circle (see Figure 5.1 and Theorem 5.1).

Remark. Here, we only considered the case of Hilbert spaces, which is the typical framework used for elliptic PDEs. However, as mentioned in the proof of Theorem 5.2, many of the above ideas and results can be adapted to the more general context of normed and Banach spaces.

5.2.2 Application to parametrized PDEs

Let us now consider the case of a PDE that depends on a vector of p parameters. We fix a parameter space $\Theta \subset \mathbb{R}^p$ and a Hilbert state space V. As before, for each $\boldsymbol{\mu} \in \Theta$ we denote the corresponding PDE solution with $u_{\boldsymbol{\mu}}$. Similarly, we define $\mathcal{S} = \{u_{\boldsymbol{\mu}}\}_{\boldsymbol{\mu}\in\Theta}$. Notice that we refer to $\mathcal{S}$ as the solution manifold even though, in fact, it is not granted that $\mathcal{S}$ is a manifold in the topological sense. This latter property can be recovered under additional hypotheses on the parameter space and the parametric map.

We consider the problem of finding a low-dimensional representation of $\mathcal{S}$ by means of nonlinear reduction. In particular, we wish to compress $\mathcal{S}$ as much as possible without paying in terms of accuracy, which corresponds to working with the minimal dimension $n_{\min}(\mathcal{S})$. To this end, we must take into account the fact that the dimension of the parameter space Θ influences the low-dimensional structure of $\mathcal{S}$, in fact, the latter is ultimately defined in terms of p scalar parameters.

Parallel to this, one may also exploit the parameters as additional tools during the dimensionality reduction process. This corresponds to replacing the solution manifold with the augmented set $\mathcal{S}_\Theta := \{(\boldsymbol{\mu}, u_{\boldsymbol{\mu}})\}_{\boldsymbol{\mu}\in\Theta} \subset \mathbb{R}^p \times V$, where $\boldsymbol{\mu}$ appears explicitly. The following Theorem provides some insights about both alternatives.

Theorem 5.3. *Let $\boldsymbol{\mu} \to u_{\boldsymbol{\mu}}$ be a map from a compact set $\Theta \subset \mathbb{R}^p$ to some Hilbert space V. Define the sets $\mathcal{S} := \{u_{\boldsymbol{\mu}}\}_{\boldsymbol{\mu}\in\Theta}$ and $\mathcal{S}_\Theta := \{(\boldsymbol{\mu}, u_{\boldsymbol{\mu}})\}_{\boldsymbol{\mu}\in\Theta}$. We have the following:*

a1) if the map $\boldsymbol{\mu} \to u_{\boldsymbol{\mu}}$ is Lipschitz continuous, then $n_{\min}(\mathcal{S}) \le 2p + 1$.

a2) if there exists at least an internal point $\boldsymbol{\mu}_0 \in \Theta$ where the correspondence $\boldsymbol{\mu} \to u_{\boldsymbol{\mu}}$ is locally injective, then $n_{\min}(\mathcal{S}) \ge p$.

a3) if the map $\boldsymbol{\mu} \to u_{\boldsymbol{\mu}}$ is continuous and injective, then $n_{\min}(\mathcal{S}) = n_{\min}(\Theta)$. In particular, $n_{\min}(\mathcal{S}) = p$ whenever Θ has nonempty interior.

b1) if the map $\boldsymbol{\mu} \to u_{\boldsymbol{\mu}}$ is continuous, then $n_{\min}(\mathcal{S}_\Theta) = n_{\min}(\Theta)$. In particular, $n_{\min}(\mathcal{S}_\Theta) = p$ whenever Θ has nonempty interior.

Proof. For the sake of brevity, let us define the map $u : \Theta \to V$ as $u(\boldsymbol{\mu}) := u_{\boldsymbol{\mu}}$.

a1) Let $\dim(\mathcal{S})$ and $\dim_H(\mathcal{S})$ be respectively the topological dimension (covering dimension) and the Hausdorff dimension of $\mathcal{S}$. A result due to Sznirelman, see Theorem 2.43 in [56], ensures that $\dim(\mathcal{S}) \le \dim_H(\mathcal{S})$. Since u is Lipschitz, we also have $\dim_H(\mathcal{S}) = \dim_H(u(\Theta)) \le \dim_H(\Theta) \le p$, as $\Theta \subset \mathbb{R}^p$ and $\dim_H$ is known to be Lipschitz subinvariant. Thus $\dim(\mathcal{S}) \le p$ and the conclusion follows by Theorem 5.2 (notice that, since Θ is compact and u is continuous, $\mathcal{S}$ is also compact).

a2) Let $B \subseteq \Theta$ be a closed ball centered at $\boldsymbol{\mu}_0$ such that $u_{|B}$, the restriction of u to B, is injective. Then, $u_{|B} : B \to u(B)$ is a continuous bijection between compact metric spaces, which is enough to grant the existence and the continuity of the inverse map $u_{|B}^{-1}$. In particular, the sets B and $u(B)$ are homeomorphic and, by Theorem 5.1, $p = n_{\min}(B) = n_{\min}(u(B))$. Since $n_{\min}(u(B)) \le n_{\min}(\mathcal{S})$, this proves (a2).

a3) As in the proof of a2), we notice that u admits a continuous inverse and is thus an homeomorphism between Θ and $\mathcal{S}$ (which are both compact). Then, $n_{\min}(\Theta) = n_{\min}(\mathcal{S})$ by Theorem 5.1. Finally, if Θ has nonempty interior, we may select a closed ball $B \subseteq \Theta$ and notice that $p = n_{\min}(B) \le n_{\min}(\Theta) \le p \implies n_{\min}(\Theta) = p$.

b1) Define the Hilbert space $\tilde{V} := \mathbb{R}^p \times V$ and the map $U : \Theta \to \tilde{V}$ as $U(\boldsymbol{\mu}) := (\boldsymbol{\mu}, u_{\boldsymbol{\mu}})$. Then U is both continuous and injective. Thus, by a3) we have $n_{\min}(\Theta) = n_{\min}(U(\Theta)) = n_{\min}(\mathcal{S}_\Theta)$.

□

Remark. Theorem 5.3 holds for a generic Hilbert-valued map, meaning that the correspondence $\boldsymbol{\mu} \to u_{\boldsymbol{\mu}}$ needs not to involve the solution of a PDE. Because of this generality, some hypotheses cannot be weakened. For instance, one cannot replace the requirement of Lipschitz continuity in statement (a1) with continuity, mainly because of space-filling curves. As a counterexample, consider the Hilbert space of real square summable sequences, $V = \ell^2$. Then, by a straightforward application of the Hahn–Mazurkiewicz theorem (see Theorem 3-30 in [86]), there exists a continuous map from the unit interval to ℓ^2 whose image $\mathcal{S}$ is the so-called Hilbert cube, informally $\mathcal{S} = \prod_{n=1}^{+\infty}[0, 1/n]$. Therefore, being $\Theta := [0,1]$ the parameter space, we have a case in which $p = 1$ but $n_{\min}(\mathcal{S}) = +\infty$. In fact, for each $n \in \mathbb{N}$, the Hilbert cube contains an homeomorphic copy of the n-dimensional unit cube I^n. Thus, $n = n_{\min}(I^n) \le n_{\min}(\mathcal{S})$ for all $n \ge 0$ and so $n_{\min}(\mathcal{S}) = +\infty$.

Before moving to the actual description of our Deep Learning approach, we conclude this Section with a practical application of the results we have presented so far. In particular, we focus on the case of second order elliptic PDEs.

Second Order Elliptic PDEs

In order to state the main result, we first provide some notation. We denote by Ω a bounded domain in $\mathbb{R}^d$ and by $\cdot$ the scalar product in $\mathbb{R}^d$. For $1 \le q < +\infty$, we denote by $L^q(\Omega)$ the Lebesgue space of q-integrable real-valued maps; when $q = +\infty$, $L^\infty(\Omega)$ is defined as the Banach space of essentially-bounded maps. Similarly, we define the spaces $L^q(\Omega, \mathbb{R}^d)$ and $L^q(\Omega, \mathbb{R}^{d\times d})$ in the Bochner sense, where $\mathbb{R}^d$ is considered with the Euclidean norm $|\cdot|$ and $\mathbb{R}^{d\times d}$ with the operator norm, $|\boldsymbol{A}|_{\mathbb{R}^{d\times d}} := \sup_{|\boldsymbol{\xi}|=1} |\boldsymbol{A}\boldsymbol{\xi}|$. Given $k \in \mathbb{N}$, $1 \le q < +\infty$, we write $W^{k,q}(\Omega)$ for the Sobolev space of all $w \in L^q(\Omega)$ that are k-times weakly differentiable with derivatives in $L^q(\Omega)$. We use $W^{k,q}_0(\Omega)$ to denote the subspace of all $w \in W^{k,q}(\Omega)$ that vanish on $\partial\Omega$, and we write $W^{-k,q}(\Omega)$ for the dual space of $W^{k,q}_0(\Omega)$ with respect to the duality product $\langle f, g\rangle \to \int_\Omega fg$. In order to prescribe Dirichlet boundary data, we also make use of the Sobolev-Slobodeckij spaces $W^{s,q}(\partial\Omega)$, where $s > 0$ is typically not an integer. All the aforementioned spaces are considered with their usual norms, see e.g. [60].

We define the sets of all admissible conductivity tensor-fields and transport fields, respectively $\Sigma(\Omega) \subset L^\infty(\Omega, \mathbb{R}^{d\times d})$ and $B(\Omega) \subset L^\infty(\Omega, \mathbb{R}^{d\times d})$, as follows. We let $\boldsymbol{\sigma} \in \Sigma(\Omega)$ if and only if it is uniformly elliptic, that is, there exists $\varepsilon > 0$ such that for almost all $x \in \Omega$ one has $\boldsymbol{\sigma}(x)\boldsymbol{\xi}\cdot\boldsymbol{\xi} \ge \varepsilon|\boldsymbol{\xi}|^2$ for all $\boldsymbol{\xi} \in \mathbb{R}^d$. We let $\boldsymbol{b} \in B(\Omega)$ if and only if it is differentiable and divergence free, that is, $\boldsymbol{b} \in \mathcal{C}^1(\Omega, \mathbb{R}^d)$ and $\nabla\cdot\boldsymbol{b} = 0$ in Ω. We endow both $\Sigma(\Omega)$ and $B(\Omega)$ with the infinity norm $\|\cdot\|_\infty$. We are now able to state the following.

Theorem 5.4. *Let $\Omega \subset \mathbb{R}^d$ be a bounded domain with Lipschitz boundary, and let $\Theta \subset \mathbb{R}^p$ be a compact subset with nonempty interior. Let $q \ge 2d/(d+2)$ be finite, and define the conjugate exponent $q' := q/(q-1)$. Moreover, let $\boldsymbol{\mu} \to \boldsymbol{\sigma}_{\boldsymbol{\mu}} \in \Sigma(\Omega)$, $\boldsymbol{\mu} \to \boldsymbol{b}_{\boldsymbol{\mu}} \in B(\Omega)$, $\boldsymbol{\mu} \to f_{\boldsymbol{\mu}} \in W^{-1,q'}(\Omega)$ be parameter dependent coefficients and $\boldsymbol{\mu} \to g_{\boldsymbol{\mu}} \in W^{1/q',q}(\partial\Omega)$ boundary data. For each $\boldsymbol{\mu} \in \Theta$, we define $u_{\boldsymbol{\mu}} \in W^{1,q}(\Omega)$ as the unique solution to the following second order elliptic PDE*

$$\begin{aligned} &u \in W^{1,q}(\Omega): \\ &\quad u_{|\partial\Omega} = g_{\boldsymbol{\mu}} \;\; \textit{and} \;\; \int_\Omega \boldsymbol{\sigma}_{\boldsymbol{\mu}}\nabla u\cdot\nabla w + \int_\Omega (\boldsymbol{b}_{\boldsymbol{\mu}}\cdot\nabla u)\, w = \int_\Omega f_{\boldsymbol{\mu}} w \quad \forall w \in W^{1,q'}_0(\Omega), \end{aligned}$$

Consider the solution manifold $\mathcal{S} := \{u_{\boldsymbol{\mu}}\}_{\boldsymbol{\mu}\in\Theta}$ as a subset of $V := L^2(\Omega)$. The following hold true:

i) if the dependence of $\boldsymbol{\sigma}_{\boldsymbol{\mu}}$, $\boldsymbol{b}_{\boldsymbol{\mu}}$, $f_{\boldsymbol{\mu}}$, $g_{\boldsymbol{\mu}}$ on $\boldsymbol{\mu}$ is Lipschitz continuous, then $n_{\min}(\mathcal{S}) \le 2p+1$.

ii) if $\boldsymbol{\sigma}_{\boldsymbol{\mu}}$, $\boldsymbol{b}_{\boldsymbol{\mu}}$, $f_{\boldsymbol{\mu}}$, $g_{\boldsymbol{\mu}}$ depend continuously on $\boldsymbol{\mu}$ and the solution map $\boldsymbol{\mu} \to u_{\boldsymbol{\mu}}$ is one-to-one, then $n_{\min}(\mathcal{S}) = p$.

Additionally, let $\mathcal{S}_\Theta := \{(\boldsymbol{\mu}, u_{\boldsymbol{\mu}})\}_{\boldsymbol{\mu}\in\Theta} \subset \mathbb{R}^d \times V$ be the augmented manifold. Then:

iii) if $\boldsymbol{\sigma}_{\boldsymbol{\mu}}$, $\boldsymbol{b}_{\boldsymbol{\mu}}$, $f_{\boldsymbol{\mu}}$, $g_{\boldsymbol{\mu}}$ depend continuously on $\boldsymbol{\mu}$, then $n_{\min}(\mathcal{S}_\Theta) = p$.

Proof. First of all, we notice that if the data $\boldsymbol{\sigma}_{\boldsymbol{\mu}}, \boldsymbol{b}_{\boldsymbol{\mu}}, f_{\boldsymbol{\mu}}, g_{\boldsymbol{\mu}}$ depend continuously on $\boldsymbol{\mu}$, then so does $u_{\boldsymbol{\mu}} \in W^{1,q}(\Omega)$. This is easily proven by composition (cf. Lemma 5.3 in the Appendix). Also, the compactness of Θ implies that of the subsets

$$\{\boldsymbol{\sigma}_{\boldsymbol{\mu}}\}_{\boldsymbol{\mu}\in\Theta} \subset \Sigma(\Omega), \quad \{\boldsymbol{b}_{\boldsymbol{\mu}}\}_{\boldsymbol{\mu}\in\Theta} \subset B(\Omega),$$
$$\{f_{\boldsymbol{\mu}}\}_{\boldsymbol{\mu}\in\Theta} \subset W^{-1,q'}(\Omega), \quad \{g_{\boldsymbol{\mu}}\}_{\boldsymbol{\mu}\in\Theta} \subset W^{1/q',q}(\partial\Omega).$$

Therefore, whenever the coefficients and the boundary data are Lipschitz continuous in $\boldsymbol{\mu}$, so is the solution map $\Theta \to W^{1,q}(\Omega)$ (by composition, cf. Lemma 5.3). Finally, since $q \ge 2d/(d+2)$, we have the embedding $W^{1,q}(\Omega) \hookrightarrow L^2(\Omega)$ according to classical Sobolev inequalities (cf. Theorem 5.4 in [2]). In particular, all the aforementioned properties are preserved if we consider the parametric map as taking values in $V := L^2(\Omega)$. Statements (i), (ii) and (iii) now directly follow from Theorem 5.3. $\square$

Remark. In Theorem 5.4, the PDE is firstly solved in the Banach space $W^{1,q}(\Omega)$ and the solution manifold is then embedded in the Hilbert space $L^2(\Omega)$. This construction allows for a large spectrum of PDEs where the solution $u_{\boldsymbol{\mu}}$ may exhibit singularities. A remarkable example is found for the dimensions $d = 2, 3$, where singular forces such as Dirac delta functions produce solutions $u_{\boldsymbol{\mu}} \notin H^1(\Omega) := W^{1,2}(\Omega)$ [34]. In these cases, the above Theorem still applies, e.g. with $q' = 4$ and $q = 4/3$ (cf. Morrey embedding). Nevertheless, we shall point out that in the Hilbert case, $q' = q = 2$, it is possible to restrict the state space to $V = H^1(\Omega) \subset L^2(\Omega)$. Note also that in this case the condition $q \geq 2d/(d+2)$ is trivially satisfied for all $d \geq 1$, coherently with the fact that $H^1(\Omega)$ always embeds in $L^2(\Omega)$.

5.3 Learning the solution manifold by means of neural networks

We now present our Deep-Learning approach to Reduced Order Modelling (DL-ROM). We shall first demonstrate the general idea, discussing both the theoretical and numerical properties of the DL-ROM. Then, respectively in Sections 5.3.1 and 5.3.2, we will dive deeper into the design choices for the nonlinear dimensionality reduction and the approximation of the reduced map. For all the basic definitions concerning neural networks and their training, we refer to the introductory Chapter at the beginning of the dissertation.

We are given a parameter space $\Theta \subset \mathbb{R}^p$, a parameter dependent PDE and a high-fidelity FOM $\boldsymbol{\mu} \to \mathbf{u}_{\boldsymbol{\mu}}^h \in \mathbb{R}^{N_h}$. Our purpose is to approximate the solution map by means of a suitable neural network $\Phi : \mathbb{R}^p \to \mathbb{R}^{N_h}$. For the sake of simplicity, through the whole section, we make the following assumption.

Assumption 5.2. *All DNNs use the same activation function $\rho : \mathbb{R} \to \mathbb{R}$ for the hidden layers, where ρ is Lipschitz continuous and not a polynomial. The parameter space Θ is compact and the parametric map $\boldsymbol{\mu} \to \mathbf{u}_{\boldsymbol{\mu}}^h$ is continuous.*

A typical activation function satisfying the above requirements is the so-called α-leaky ReLU, i.e. $\rho(x) = x\mathbf{1}_{[0,+\infty)}(x) + \alpha x \mathbf{1}_{(-\infty,0)}(x)$ where $\alpha > 0$ is fixed. In order to build Φ, we mimic the two steps paradigm of the Reduced Basis method, yielding the workflow depicted in Figure 5.2. This corresponds to introducing the three networks below,

$$\Psi' : \mathbb{R}^{N_h} \to \mathbb{R}^n, \qquad \Psi : \mathbb{R}^n \to \mathbb{R}^{N_h}$$

$$\phi : \mathbb{R}^p \to \mathbb{R}^n.$$

The first two, respectively the encoder Ψ' and the decoder Ψ, serve for the nonlinear dimensionality reduction of the solution manifold $\mathcal{S}^h := \{\mathbf{u}_{\boldsymbol{\mu}}^h\}_{\boldsymbol{\mu} \in \Theta} \subset \mathbb{R}^{N_h}$, which we map onto $\mathbb{R}^n$. According to our previous analysis, we set the latent dimension to be $n := n_{\min}(\mathcal{S}^h)$. As discussed in Theorem 5.3, this often translates to $n \leq 2p + 1$, resulting in a massive reduction whenever $p \ll N_h$. The purpose of the third network is to approximate the reduced parametric map $\mathbb{R}^p \ni \boldsymbol{\mu} \to \Psi'(\mathbf{u}_{\boldsymbol{\mu}}^h) \in \mathbb{R}^n$, so that the final ROM is obtained by composition of ϕ and Ψ.

At the very end, the role of the encoder Ψ' is only auxiliary as the DL-ROM ultimately results in a single network $\Phi := \Psi \circ \phi$. However, we believe that our construction significantly facilitates the practical problem of designing the architectures. This is because the three networks have very different purposes. Ψ' and Ψ are required to learn the intrinsic characteristics of the solutions, so their complexity is related to the richness of the solution manifold and the geometrical properties of the solutions. Conversely, ϕ needs to understand the interplay between solutions and parameters, which can result in a very complicated

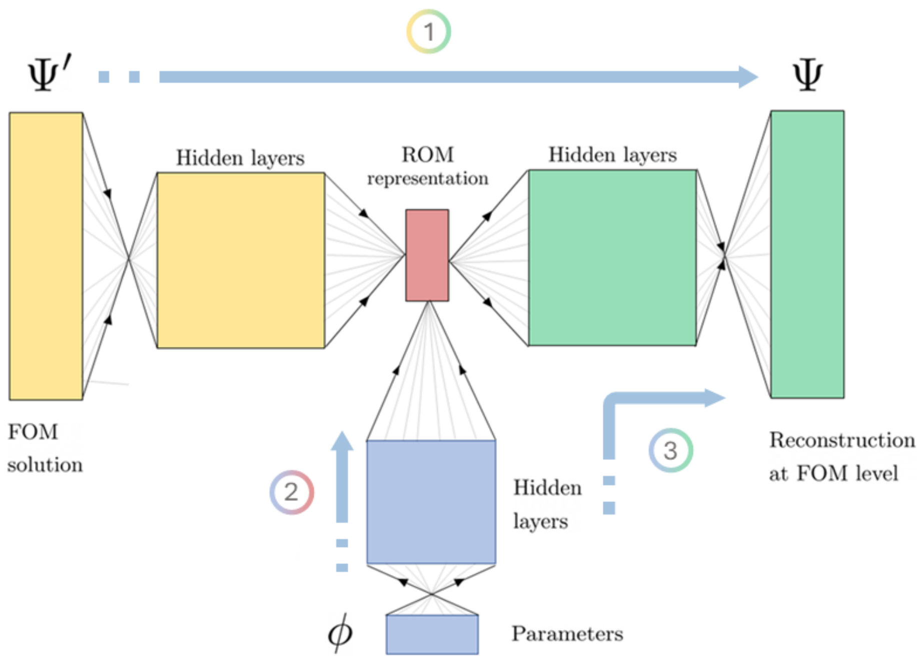


Figure 5.2: Workflow of the DL-ROM approach. The whole process consists of three neural networks, Ψ', Ψ and ϕ. First, the autoencoder $\Psi \circ \Psi'$ learns to compress and reconstruct the solution manifold by exploiting the available highfidelity snapshots (step 1). This allows the encoder to yield a low-dimensional representation of the solutions, $\mathbf{u}_{\boldsymbol{\mu}}^h \in \mathbb{R}^{N_h} \to \mathbf{u}_{\boldsymbol{\mu}}^n := \Psi'(\mathbf{u}_{\boldsymbol{\mu}}^h) \in \mathbb{R}^n$. Then, ϕ is trained to learn the reduced map $\boldsymbol{\mu} \to \mathbf{u}_{\boldsymbol{\mu}}^n$ (step 2). Finally, the composition $\Phi := \Psi \circ \phi$ defines the DL-ROM approximation of the parameter-to-state map (step 3).

relation even if the solution manifold is fairly simple (e.g. linear). As the design of DNN architectures is still far from obvious, we believe that this perspective can be of help in practical implementations. Nonetheless, this splitting of the ROM also allows for a few considerations on the numerical errors, as we shall discuss at the end in Section 5.3.3.

5.3.1 Dimensionality reduction

We propose two alternative ways for compressing the solution manifold. The first one is completely unsupervised, in the sense that it only operates on the solutions irrespectively of the parameter values, and it is based on the use of autoencoders. The second one is a variation of the previous where we explicitly include $\boldsymbol{\mu}$ in the encoding process. We detail them below.

Autoencoder approach

According to the reasoning in Section 5.3, we let $n := n_{\min}(\mathcal{S}^h)$ and we introduce two DNN architectures, an encoder $\Psi' : \mathbb{R}^{N_h} \to \mathbb{R}^n$ and a decoder $\Psi : \mathbb{R}^n \to \mathbb{R}^{N_h}$, which we design as follows. In principle, the encoder can be very simple, as its only purpose is to provide a different representation for each solution. The hard job is left to the decoder that needs to perform the reconstruction. In this sense, a plain design choice can be $\Psi'(\mathbf{u}) := \rho(\mathbf{W}\mathbf{u} + \mathbf{b})$, i.e. to use a degenerate architecture with no hidden layers. Conversely, designing the decoder requires a little extra caution. If Ω is an hypercube, a good choice is to employ dense layers at the beginning and conclude with a block of convolutional layers, as in [70, 118]. This allows the decoder to account for spatial correlations and be sufficiently expressive without growing too much in complexity (for now, this is just a rule of thumb suggested by the intuition, however we shall make it rigorous in the next Chapter, see Theorem 6.1). Indeed, convolutional layers have been proven to be very effective in image reconstruction tasks, and we see a clear analogy with our setting when Ω is an hypercube. More complicated geometries may require different strategies, but the terminal part of the decoder should still consist of sparse layers of some sort, such as those in Graph Convolutional Networks [172] or Mesh-Informed Neural Networks, see Chapter 7. The expressiveness of the decoder may be increased in several ways. Empirically, we see that interesting results can be obtained for fixed depths but varying number of *channels* in the convolutional layers. Once again, we shall justify this fact in the next Chapter, Theorem 6.1.

After having fixed the DNN architecture, we optimize the autoencoder by minimizing the loss function below

$$\mathscr{L}(\Psi', \Psi) = \frac{1}{N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} \ell(\mathbf{u}^h_{\boldsymbol{\mu}_i},\ \Psi(\Psi'(\mathbf{u}^h_{\boldsymbol{\mu}_i}))),$$

where ℓ is a suitable measure of discrepancy. A classical choice is to consider squared errors, $\ell(\mathbf{y}, \hat{\mathbf{y}}) = \|\mathbf{y} - \hat{\mathbf{y}}\|^2$, in order to favor differentiability of the loss function. However, other metrics, such as relative errors $\ell(\mathbf{y}, \hat{\mathbf{y}}) = \|\mathbf{y} - \hat{\mathbf{y}}\| / \|\mathbf{y}\|$, can be used as well. The minimization of the loss function is handled via stochastic gradient descent, mainly using batching strategies and first order optimizers. For further details about the training of DNN models, we refer to the preliminary Chapter at the beginning of the dissertation.

Transcoder-decoder approach

As an alternative, we also propose a different architecture where the encoder is replaced with a *transcoder* $\Psi'_\mu : \mathbb{R}^p \times \mathbb{R}^{N_h} \to \mathbb{R}^n$. The idea is to facilitate the encoding by making explicitly use of the parameters, so that different solutions are more likely to have different latent representations. This is clearly linked with Theorem 5.3.b1, and has the advantage of always enabling a maximal reduction, as we can now set $n = p = n_{\min}(\{\boldsymbol{\mu}, \mathbf{u}^h_{\boldsymbol{\mu}}\}_{\boldsymbol{\mu}\in\Theta})$. We define the decoder exactly as before, so that $\mathbf{u}^h_{\boldsymbol{\mu}} \approx \Psi(\Psi'_\mu(\boldsymbol{\mu}, \mathbf{u}^h_{\boldsymbol{\mu}}))$. We refer to the combined architecture, $\Psi \circ \Psi'_\mu$, as to a *transcoder-decoder*. In practice, the transcoder-decoder is analogous to an autoencoder but has p additional neurons in the input layer, which is where we pass the parameters. To design the architectures, we follow the same rule of thumb as before. In general, we give more weight to the decoder, where we employ deep convolutional networks, while we use lighter architectures for the transcoder. For instance, in the limit case of 0-depth, the latter becomes of the form $\Psi'_\mu(\boldsymbol{\mu}, \mathbf{u}) = \rho(\mathbf{W}'\boldsymbol{\mu} + \mathbf{W}\mathbf{u} + \mathbf{b})$. During the offline stage, the transcoder-decoder is trained over the snapshots by minimizing the loss function below,

$$\mathscr{L}(\Psi'_\mu, \Psi) = \frac{1}{N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} \ell(\mathbf{u}^h_{\boldsymbol{\mu}_i},\ \Psi(\Psi'_\mu(\boldsymbol{\mu}_i, \mathbf{u}^h_{\boldsymbol{\mu}_i}))),$$

where ℓ is as before. The two approaches, autoencoder and transcoder-decoder, adopt different perspectives and provide different advantages. The first one is completely based on the solution manifold, so it is likely to reflect intrinsic properties of $\mathcal{S}^h$. On the other hand, the transcoder-decoder ensures a maximal compression, the latent dimension being always equal to p. In particular, the latent coordinates can be seen as an alternative parametrization of the solution manifold. In this sense, we say that Ψ'_μ performs a transcoding.

5.3.2 Approximation of the reduced map

The second step in the DL-ROM pipeline is to approximate the reduced map $\mathbb{R}^p \ni \boldsymbol{\mu} \to \mathbf{u}^n_{\boldsymbol{\mu}} \in \mathbb{R}^n$, where either $\mathbf{u}^n_{\boldsymbol{\mu}} := \Psi'(\mathbf{u}^h_{\boldsymbol{\mu}})$ or $\mathbf{u}^n_{\boldsymbol{\mu}} := \Psi'_\mu(\boldsymbol{\mu}, \mathbf{u}^h_{\boldsymbol{\mu}})$, depending on the adopted approach. As we noted in Section 5.3, the reduced map is continuous, as it is given by the composition of $\boldsymbol{\mu} \to \mathbf{u}^h_{\boldsymbol{\mu}}$ and Ψ' (resp. Ψ'_μ), hence it can be approximated uniformly by some ρ-DNN $\phi : \mathbb{R}^p \to \mathbb{R}^n$. In general, we do not impose a particular structure on ϕ, rather we use a generic fully connected network with dense layers. To design the architecture in terms of number of layers and neurons, we rely on Theorem 5.5 (next subsection) and on the

Algorithm 5.1: DL-ROM training.

```
Input  : Training snapshots $\{\boldsymbol{\mu}_i, \mathbf{u}^h_{\boldsymbol{\mu}_i}\}_{i=1}^N$, reduced dimension $n$, optimizers
         Optimizer$_1$, Optimizer$_2$, number(s) of epochs $E_1$, $E_2$, batch size(s) $S_1$,
         $S_2$, encoding type useparameters (boolean), discrepancy measures $\ell_1$, $\ell_2$.
Output: Neural network $\Phi$ approximating the parametric map.

$\Psi'_0, \Psi_0 \leftarrow$ Initialize encoder/transcoder and decoder with latent dimension $n$
$e \leftarrow 0$ Initialize epochs counter
$B_1 \leftarrow N/S_1$    // number of batches
if useparameters then
    $\mathbf{v}_i \leftarrow [\boldsymbol{\mu}_i, \mathbf{u}^h_{\boldsymbol{\mu}_i}]$    // transcoder case
else
    $\mathbf{v}_i \leftarrow \mathbf{u}^h_{\boldsymbol{\mu}_i}$    // encoder case
end
while $e < E_1$ do
    shuffle training data $\{\mathbf{v}_i, \mathbf{u}^h_{\boldsymbol{\mu}_i}\}_{i=1}^N$;
    for $m = 1 : B_1$ do
        $\mathbf{v}^{\text{batch}} \leftarrow [\mathbf{v}_{(m-1)S_1+1}, .., \mathbf{v}_{mS_1}]$
        $\mathbf{u}^{\text{batch}} \leftarrow [\mathbf{u}^h_{\boldsymbol{\mu}_{(m-1)S_1+1}}, .., \mathbf{u}^h_{\boldsymbol{\mu}_{mS_1}}]$
        loss $\leftarrow \frac{1}{S_1}\sum_{i=1}^{S_1} \ell_1\left(\mathbf{u}^{\text{batch}}_i, \Psi_{eB_1+m-1}(\Psi'_{eB_1+m-1}(\mathbf{v}^{\text{batch}}_i))\right)$
        $\Psi'_{eB_1+m}, \Psi_{eB_1+m} \leftarrow$ Optimizer$_1$(loss, $\Psi'_{eB_1+m-1}, \Psi_{eB_1+m-1}$)
    end
    $e \leftarrow e + 1$
end

$\phi_0 \leftarrow$ Initialize reduced map DNN
$e \leftarrow 0$ Reset epochs counter
$B_2 \leftarrow N/S_2$    // number of batches
$\mathbf{u}^n_{\boldsymbol{\mu}_i} \leftarrow \Psi'(\mathbf{v}_i)$    // define training data for $\phi$
while $e < E_2$ do
    shuffle training data $\{\boldsymbol{\mu}_i, \mathbf{u}^n_{\boldsymbol{\mu}_i}\}_{i=1}^N$;
    for $m = 1 : B_2$ do
        $\boldsymbol{\mu}^{\text{batch}} \leftarrow [\boldsymbol{\mu}_{(m-1)S_2+1}, .., \boldsymbol{\mu}_{mS_2}]$
        $\mathbf{u}^{n,\text{batch}} \leftarrow [\mathbf{u}^n_{\boldsymbol{\mu}_{(m-1)S_2+1}}, .., \mathbf{u}^n_{\boldsymbol{\mu}_{mS_2}}]$
        loss $\leftarrow \frac{1}{S_2}\sum_{i=1}^{S_2} \ell_2\left(\mathbf{u}^{n,\text{batch}}_i, \phi_{eB_2+m-1}(\boldsymbol{\mu}^{\text{batch}}_i)\right)$
        $\phi_{eB_2+m} \leftarrow$ Optimizer$_2$(loss, $\phi_{eB_2+m-1}$)
    end
    $e \leftarrow e + 1$
end

$\Phi \leftarrow \Psi_{E_1B_1} \circ \phi_{E_2B_2}$

return $\Phi$
```

underlying theoretical results available in the literature, e.g. [27, 52, 79, 153, 157, 182]. In order to train ϕ we minimize the objective function below

$$\mathscr{L}(\phi) = \frac{1}{N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} \ell(\mathbf{u}^n_{\boldsymbol{\mu}_i}, \phi(\boldsymbol{\mu}_i))$$

where, once again, ℓ is some discrepancy measure (this time having inputs in $\mathbb{R}^n \times \mathbb{R}^n$). Notice that the optimization of the above only involves ϕ, as the weights and biases of Ψ' (resp. Ψ'_μ) are frozen. At the end of the whole process, which we summarized in Algorithm 1, we let $\Phi := \Psi \circ \phi$.

Now the DL-ROM is fully operational, and for each new $\boldsymbol{\mu} \in \Theta$ we can approximate online the corresponding solution $\Phi(\boldsymbol{\mu}) \approx \mathbf{u}^h_{\boldsymbol{\mu}}$ almost effortlessly, with very little computational cost. Also, the model can be efficiently evaluated on multiple parameter values simultaneously. In fact, as DNNs are ultimately based on elementary linear algebra, it possible to stack together multiple parameter vectors $\mathbf{M} = [\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_l]$ in a single matrix and directly return the corresponding list of ROM approximations $\Phi(\mathbf{M}) \approx [\mathbf{u}^h_1, \dots, \mathbf{u}^h_l]$.

Remark. We mention that, in the case $n = p$, an interesting alternative for ϕ could be provided by the so-called ODE-nets [40]. In fact, if the reduced map happens to be injective, then Θ and $\{\mathbf{u}^n_{\boldsymbol{\mu}}\}_{\mu \in \Theta}$ define two homeomorphic sets of coordinates. Even though homeomorphisms can be approximated by classical DNNs, we note that fully connected unconstrained networks can easily result in noninvertible models. In this sense, an alternative architecture which ensures the existence and continuity of ϕ^{-1} would be appealing. ODE-nets enjoys such property and have been proven to be universal approximators for homeomorphisms [205]. However, the development and implementation of ODE-nets is still in its infancy so we did not investigate this further.

5.3.3 Error analysis

We conclude this Section with some considerations about the numerical errors entailed by the DL-ROM approach. We notice that, due to assumption (5.2), all the networks in the DL-ROM pipeline are Lipschitz continuous. Also, without loss of generality, we can assume that Ψ has a Lipschitz constant equal to 1. In fact, for any $C > 0$, the maps $\tilde{\Psi}'(\mathbf{x}) := C\Psi'(\mathbf{x})$ and $\tilde{\Psi}(\mathbf{x}) := \Psi(\mathbf{x}/C)$ define the same autoencoder as Ψ' and Ψ. As a consequence, the worst-case approximation error of the DL-ROM

$$\mathcal{E}_A := \sup_{\mu \in \Theta} \|\mathbf{u}^h_{\boldsymbol{\mu}} - \Psi(\phi(\boldsymbol{\mu}))\|,$$

can be bounded as $\mathcal{E}_A \le \mathcal{E}_R + \mathcal{E}_P$, the latter being respectively the reconstruction error and the parametric error,

$$\mathcal{E}_R := \sup_{\mathbf{u}^h \in \mathcal{S}^h} \|\mathbf{u}^h - \Psi(\Psi'(\mathbf{u}^h))\|, \qquad \mathcal{E}_P := \sup_{\mu \in \Theta} |\Psi'(\mathbf{u}^h_{\boldsymbol{\mu}}) - \phi(\boldsymbol{\mu})|.$$

where $|\cdot|$ is the Euclidean norm, while we recall that $\|\cdot\|$ comes from the metric originally chosen over the state space $V_h \subset V$. In fact,

$$\sup_{\mu \in \Theta} \|\mathbf{u}^h_{\boldsymbol{\mu}} - \Psi(\phi(\boldsymbol{\mu}))\| \le \sup_{\mu \in \Theta} \|\mathbf{u}^h_{\boldsymbol{\mu}} - \Psi(\Psi'(\mathbf{u}^h_{\boldsymbol{\mu}}))\| + \sup_{\mu \in \Theta} \|\Psi(\Psi'(\mathbf{u}^h_{\boldsymbol{\mu}})) - \Psi(\phi(\boldsymbol{\mu}))\| \le$$
$$\le \sup_{\mathbf{u}^h \in \mathcal{S}^h} \|\mathbf{u}^h - \Psi(\Psi'(\mathbf{u}^h))\| + \sup_{\mu \in \Theta} |\Psi'(\mathbf{u}^h_{\boldsymbol{\mu}}) - \phi(\boldsymbol{\mu})|.$$

We remark that both $\mathcal{E}_R$ and $\mathcal{E}_P$ can be made arbitrarily small. In fact, as proven by Pinkus back in 1999 [159], DNNs are dense in the space of continuous functions defined over compact domains (note that here our assumption on ρ is crucial). Therefore, since Θ is compact and

$\boldsymbol{\mu} \to \Psi'(\mathbf{u}^h_{\boldsymbol{\mu}})$ is continuous, the parametric error can become as small as possible. Similarly, the reconstruction error can get closer and closer to the limit value $\delta_n(\mathcal{S}^h) = 0$. In fact, we can approximate the reconstruction provided by any two continuous maps $\Psi'_* : \mathcal{S}^h \to \mathbb{R}^n$ and $\Psi_* : \mathbb{R}^n \to \mathbb{R}^{N_h}$ using DNNs. To see this, fix any $\varepsilon > 0$ and let $\mathcal{V} := \Psi_*(\mathcal{S}^h)$ be the embedded solution manifold. Since $\mathcal{V}$ is compact, the aforementioned density result ensures the existence of some DNN Ψ that approximates Ψ_* over $\mathcal{V}$ upto an error of ε. Similarly, there exists a DNN Ψ' that approximates Ψ'_* over $\mathcal{S}^h$ upto an error of ε/C, where $C > 0$ is the Lipschitz constant of Ψ. Then, for any $\mathbf{u}^h \in \mathcal{S}^h$ one has

$$\begin{aligned}
&\|\mathbf{u}^h - \Psi(\Psi'(\mathbf{u}^h))\| \le \\
&\quad \le \|\mathbf{u}^h - \Psi_*(\Psi'_*(\mathbf{u}^h))\| + \|\Psi_*(\Psi'_*(\mathbf{u}^h)) - \Psi(\Psi'_*(\mathbf{u}^h))\| + \|\Psi(\Psi'_*(\mathbf{u}^h)) - \Psi(\Psi'(\mathbf{u}^h))\| \le \\
&\qquad \le \|\mathbf{u}^h - \Psi_*(\Psi'_*(\mathbf{u}^h))\| + \varepsilon + C|\Psi'_*(\mathbf{u}^h)) - \Psi'(\mathbf{u}^h)| \le \\
&\qquad\qquad \le \|\mathbf{u}^h - \Psi_*(\Psi'_*(\mathbf{u}^h))\| + 2\varepsilon.
\end{aligned}$$

This shows that $\mathcal{E}_R$ can reach the limit value $\delta_n(\mathcal{S}^h)$. In particular, since we decided to let $n = n_{\min}(\mathcal{S}^h)$, the reconstruction error can get arbitrarily close to zero.

In general, all these considerations suggest a two-step approach where we first train the autoencoder $\Psi \circ \Psi'$ and then the reduced map ϕ, as we discussed in Sections 5.3.1 and 5.3.2. Nevertheless, an additional analysis about the networks complexity is needed. In fact, while the DL-ROM can reach any level of accuracy, the size of the networks involved may grow quickly, making their optimization problematic. The result below provides a first answer to such question.

Theorem 5.5. *Under Assumption 5.2, let ρ be the ReLU activation function. Assume that the map $\boldsymbol{\mu} \to \mathbf{u}^h_{\boldsymbol{\mu}}$ is Lipschitz continuous for some constant $L > 0$, and that the infimum in* (5.5) *is attained, i.e. there exists two continuous maps $\Psi'_* : \mathbb{R}^{N_h} \to \mathbb{R}^n$ and $\Psi_* : \mathbb{R}^n \to \mathbb{R}^{N_h}$ such that*

$$\Psi_*(\Psi'_*(\mathbf{u})) = \mathbf{u} \qquad \forall \mathbf{u} \in \mathcal{S}^h.$$

Additionally, assume that Ψ'_* and Ψ_* are s-times differentiable, $s \ge 2$, and have bounded derivatives. Let

$$C_1 = \sup_{|\boldsymbol{\alpha}| \le s} \sup_{\mathbf{u} \in \mathbb{R}^{N_h}} |D^{\boldsymbol{\alpha}} \Psi'_*(\mathbf{u})|, \qquad C_2 = \sup_{|\boldsymbol{\alpha}| \le s} \sup_{\boldsymbol{\nu} \in \mathbb{R}^n} \|D^{\boldsymbol{\alpha}} \Psi_*(\boldsymbol{\nu})\|.$$

For any $0 < \varepsilon < 1$, let $m \in \mathbb{N}$ be the first integer for which $d_m(\mathcal{S}^h) < \varepsilon$. Then, for some constant $c = c(\Theta, L, C_1, C_2, p, n, s)$, there exists a DL-ROM with a decoder Ψ having at most

i) $cm^{1+n/(s-1)}\varepsilon^{-n/(s-1)}\log(m/\varepsilon) + mN_h$ active weights

ii) $c\log(m/\varepsilon)$ layers

and a reduced map ϕ having at most

iii) $c\varepsilon^{-p}\log(1/\varepsilon)$ active weights

iv) $c\log(1/\varepsilon)$ layers

such that the approximation error satisfies $\mathcal{E}_A < 2\varepsilon$.

Proof. Let $\mathcal{V} := \Psi'_*(\mathcal{S}^h) \subset \mathbb{R}^n$ be the embedded solution manifold. The latter is a compact subset of diameter at most $\text{diam}(\mathcal{V}) \le LC_1\text{diam}(\Theta)$. Let m be as in the Theorem. Then there exists an orthonormal matrix $\mathbf{V} \in \mathbb{R}^{N_h \times m}$ such that $\|\mathbf{u} - \mathbf{V}\mathbf{V}^T\mathbf{u}\| < \varepsilon$ for all $\mathbf{u} \in \mathcal{S}^h$. Define $F : \mathbb{R}^n \to \mathbb{R}^m$ as $F(\boldsymbol{\nu}) := \mathbf{V}^T\Psi_*(\boldsymbol{\nu})$. Then F is s-times differentiable and

$$\sup_{\boldsymbol{\nu} \in \mathcal{V}} |D^{\boldsymbol{\alpha}} F(\boldsymbol{\nu})| \le \sup_{\boldsymbol{\nu} \in \mathcal{V}} \|D^{\boldsymbol{\alpha}} \Psi_*(\boldsymbol{\nu})\| \le C_2,$$

for any multi-index $\boldsymbol{\alpha}$ with $0 \le |\boldsymbol{\alpha}| \le s$. In particular, as a direct consequence of Theorem 4.1 in [78], there exists a ReLU DNN $\psi : \mathbb{R}^n \to \mathbb{R}^m$ of depth at most $C \log(m/\varepsilon)$ and active weights at most $mC(m\varepsilon)^{-n/(s-1)} \log(m/\varepsilon)$ such that for all $j = 1, \dots, m$ one has

$$\sup_{\boldsymbol{\nu} \in \mathcal{V}} |F_j(\boldsymbol{\nu}) - \psi_j(\boldsymbol{\nu})| \le \frac{\varepsilon}{2m}, \qquad \operatorname*{ess\,sup}_{\boldsymbol{\nu},\, \boldsymbol{\nu}' \in \mathcal{V}} \frac{|(F_j - \psi_j)(\boldsymbol{\nu}) - (F_j - \psi_j)(\boldsymbol{\nu}')|}{|\boldsymbol{\nu} - \boldsymbol{\nu}'|} \le \frac{\varepsilon}{2m}$$

where $C > 0$ is a constant depending on C_2, n, s and $\operatorname{diam}(\mathcal{V})$ (thus on Θ, L and C_1), wherease F_j and ψ_j are the jth components of the two vector-valued maps. In particular, we also have a control on the Lipschitz constant of ψ, which we can bound by $C_2 + \varepsilon/(2m) \le C_2 + 1$. We now define the decoder DNN $\Psi(\boldsymbol{\nu}) := \mathbf{V}\psi(\boldsymbol{\nu})$ as a network with no activation on the output layer, so that for any $\boldsymbol{\nu} \in \mathcal{V}$ we have

$$\begin{aligned}\|\Psi_*(\boldsymbol{\nu}) - \Psi(\boldsymbol{\nu})\| \le \|\Psi_*(\boldsymbol{\nu}) - \mathbf{V}\mathbf{V}^T\Psi_*(\boldsymbol{\nu})\| + \|\mathbf{V}\mathbf{V}^T\Psi_*(\boldsymbol{\nu}) - \Psi(\boldsymbol{\nu})\| \le \\ \le \varepsilon + |F(\boldsymbol{\nu}) - \psi(\boldsymbol{\nu})| \le \frac{3}{2}\varepsilon\end{aligned}$$

as $\Psi_*(\boldsymbol{\nu}) \in \mathcal{S}^h$. Clearly, Ψ has the same depth of ψ up to one layer, and it comes with mN_h additional weights. Define now $\phi_* : \mathbb{R}^p \to \mathbb{R}^n$ as $\phi_*(\boldsymbol{\mu}) := \Psi'_*(\mathbf{u}^h_{\boldsymbol{\mu}})$. Then ϕ_* is Lipschitz continuous with constant at most equal to LC_1 and is bounded in norm by C_1. Thus, we can apply again Theorem 4.1 in [78], this time with respect to the infinity norm, to obtain a DNN $\phi : \mathbb{R}^p \to \mathbb{R}^n$ such that for all $\boldsymbol{\mu} \in \Theta$

$$|\phi_{*,j}(\boldsymbol{\mu}) - \phi_j(\boldsymbol{\mu})| < \frac{\varepsilon}{2n(C_2 + 1)} \qquad \forall j = 1, \dots, n$$

where ϕ has at most $n\tilde{C}(nC_2\varepsilon)^{-p}\log(nC_2/\varepsilon)$ weights and at most $\tilde{C}\log(nC_2/\varepsilon)$ layers, $\tilde{C}$ being a constant only dependent on Θ, L and C_1. In particular, if we let $\tilde{C}$ absorb the dependence with respect to n and C_2, and we set $c := \max\{C, \tilde{C}\}$, then the architectures of Ψ and ϕ satisfy the claimed requirements. Finally, we note that $\mathbf{u}^h_{\boldsymbol{\mu}} = \Psi_*(\Psi'_*(\mathbf{u}^h_{\boldsymbol{\mu}})) = \Psi_*(\phi_*(\boldsymbol{\mu}))$, due to our hypothesis on the perfect embedding. Therefore, the approximation error for the DL-ROM with decoder Ψ and reduced map ϕ is bounded by

$$\begin{aligned}\mathcal{E}_A &\le \sup_{\boldsymbol{\mu} \in \Theta} \|\Psi_*(\phi_*(\boldsymbol{\mu})) - \Psi(\phi_*(\boldsymbol{\mu}))\| + \sup_{\boldsymbol{\mu} \in \Theta} \|\Psi(\phi_*(\boldsymbol{\mu})) - \Psi(\phi(\boldsymbol{\mu}))\| \le \\ &\le \sup_{\boldsymbol{\nu} \in \mathcal{V}} \|\Psi_*(\boldsymbol{\nu}) - \Psi(\boldsymbol{\nu})\| + (C_2 + 1) \sup_{\boldsymbol{\mu} \in \Theta} |\phi_*(\boldsymbol{\mu}) - \phi(\boldsymbol{\mu})| \le \\ &\le \frac{3}{2}\varepsilon + (C_2 + 1)\frac{\varepsilon}{2(C_2 + 1)} = 2\varepsilon.\end{aligned}$$

$\square$

Theorem 5.5 suggests that the DL-ROM approach can take advantage of intrinsic regularities in the solution manifold, even if the parameter-to-solution map is just Lipschitz continuous. This situation reflects the case in which although the solutions depend in a complicated way with respect to the parameters, the solution operator has good analytical properties. For instance, it is known that the solution operator of elliptic PDEs is analytic with respect to the coefficients [8, 88]. Thus, we can think of $\boldsymbol{\mu} \to \phi(\boldsymbol{\mu})$ as a change of coordinates that enables a smooth description of the solutions.

Secondly, we note that an important role is played by the parameter m. This is in agreement with other results in the literature, see e.g. Theorem 4.3 in [112], and it suggests a link between the DL-ROM complexity and the linear Kolmogorov m-width. We may interpret m as an *equivalent linear dimension*: in fact, the DL-ROM accuracy in Theorem 5.5 is roughly equivalent to the optimal one achievable via projections on m-dimensional subspaces. In this sense, we can think of m as being the number of modes in a Reduced Basis approach or, analogously, the number of trunk nets in a DeepONet based ROM [126, 115].

In the case of DL-ROMs, the value of m does not affect the latent dimension but has an impact on the DNNs complexity: the slower $d_m(\mathcal{S}^h)$ decays, the more degrees of freedom in the DNN architecture and, consequently, the higher the number of training snapshots required for the optimization. Conversely, if the linear width decays mildly, then a mix of linear and nonlinear reduction may be an interesting choice, as in the recently proposed POD-DL-ROM approach [71].

Nevertheless, we mention that the complexity bounds for the decoder are suboptimal in the way they include the FOM dimension N_h. In fact, the extra contribute mN_h comes from the choice of considering the state space V_h as consisting of vectors rather than functions. In particular, we expect that better estimates can be found if the solutions are smooth with respect to the space variable $\mathbf{x} \in \Omega$. This goes in favor of architectures that explicitly account for space dependency, such as convolutional layers (see the next Chapter), or even mesh-free approaches, such as DeepONets.

5.4 Numerical experiments

We now present some numerical results obtained with our DL-ROM approach. So far, neural networks have shown remarkable performances in the approximation of the parametric map at least in those contexts where classical POD-based methods succeed, e.g. [24, 72]. There is now an increasing interest in understanding how and if NNs can be of help in more challenging situations. In the case of transport problems, some theoretical and numerical results are now appearing in the literature, see respectively [113] and [70].

Here, we focus on parameter dependent second order elliptic PDEs, which can considered as a prototypical class of problems for forthcoming applications. The first test case concerns an advection-diffusion problem with a singular source term. The PDE depends on 7 scalar parameters which affect the equation both in a linear and nonlinear fashion. We consider two variants of the same problem, one of which is transport-dominated. The idea is to study the behavior of the DL-ROM when dealing with space singularities, as that is a feature also shared by the oxygen transfer model in Section 4.1.

As second test case, we consider a stochastic Poisson equation. The main difference with respect to the previous case is that the equation is parametrized by a stochastic process, with PDE formally depending on an infinite-dimensional parameter. In order to apply the DL-ROM approach, we consider a suitable truncation of the Karhunen–Loève expansion of the stochastic process. In this case, the problem under study can be seen as a prototype model for diffusion phenomena in porous media, which is the typical mathematical framework encountered when describing the perfusion of organs at the macroscale [168].

In all our experiments we consider $V = L^2(\Omega)$ as state space, and we quantify the ROM performance via the Mean Relative Error (MRE)

$$\mathbb{E}_{\boldsymbol{\mu}\sim\mathbb{P}}\left[\frac{\|\mathbf{u}_{\boldsymbol{\mu}}^h - \Phi(\boldsymbol{\mu})\|_{L^2(\Omega)}}{\|\mathbf{u}_{\boldsymbol{\mu}}^h\|_{L^2(\Omega)}}\right], \tag{5.6}$$

where $\Phi = \Psi \circ \phi$ is the DL-ROM network, and $\mathbb{P}$ is some probability measure defined over the parameter space Θ. We estimate (5.6) with a Monte Carlo average computed over 1000 unseen snapshots (test set). To evaluate whether there is a gap in performance between training and testing, we also compute the MREs over the training set. For an easier comparison, in all our experiments, we fix the latter to have size $N_{\text{train}} = 9000$.

We highlight that, differently from our worst-case approach in Sections 5.2 and 5.3, Equation (5.6) adopts a probabilistic perspective in evaluating the model performance. As this is a common choice in many applications, see e.g. [72, 70, 136], we have decided to adopt it for our numerical experiments. Additionally, resorting to expectations rather than suprema, allows us to handle general unbounded parameter spaces. In fact, our theoretical results can be easily extended to such settings if one adopts a probabilistic perspective. We shall briefly comment on this fact when discussing our second experiment, Section 5.4.2.

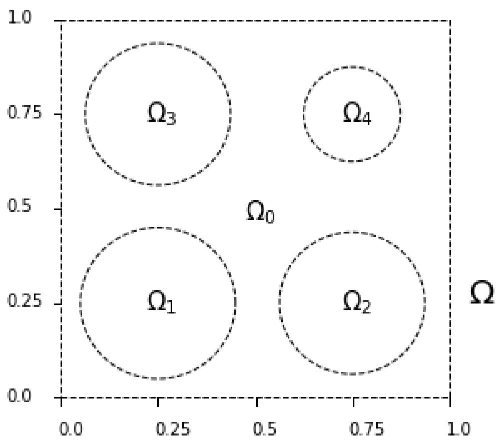


Figure 5.3: Decomposition of the unit square $\Omega = (0,1)^2$ according to the conventions adopted in the first numerical experiment, Section 5.4.1.

All our experiments were implemented in Python 3 and ran over GPUs. Specifically, we used the FEniCS library[1] to run the FOMs and obtain the high-fidelity snapshots, while the construction and the training of the DL-ROM was handled in Pytorch[2].

5.4.1 Stationary advection-diffusion with singular source

Problem definition

On the spatial domain $\Omega = (0,1)^2$, we define the subdomains $\{\Omega_i\}_{i=0}^4$ as in Figure 5.3. We consider the following parameter dependent PDE in weak form

$$\frac{1}{10}\int_{\Omega_0}\nabla u\cdot\nabla w+\sum_{i=1}^{4}\int_{\Omega_i}\mu_i\nabla u\cdot\nabla w+C\int_{\Omega}\left(\cos\mu_5\frac{\partial u}{\partial x_1}w+\sin\mu_5\frac{\partial u}{\partial x_2}w\right)=$$

$$=w(\mu_6,\mu_7)\quad\forall w\in\mathcal{C}_0^\infty(\Omega)$$

with Dirichlet boundary condition $u_{|\Omega}=1$. The above corresponds to a stationary advection-diffusion equation where: the conductivity field $\sigma_{\boldsymbol{\mu}} := 0.1+\sum_{i=1}^4\mu_i\mathbf{1}_{\Omega_i}$ is piecewise constant with values that change parametrically within the circular subdomains; the transport field $\boldsymbol{b}_{\boldsymbol{\mu}} := (C\cos\mu_5, C\sin\mu_5)$ has a parametrized direction while it is uniform in space and has a fixed intensity $C>0$; finally, the source term $f_{\boldsymbol{\mu}}$ is a Dirac delta located at the parameter dependent coordinates (μ_6,μ_7). Globally, the PDE depends on 7 parameters that we consider to be varying in the parameter space $\Theta=[0,1]^4\times[0,2\pi]\times[0.1,0.9]^2$, which we endow with a uniform probability distribution $\mathbb{P}$. We note that the PDE does not admit solutions in $H^1(\Omega)$ because of the singularity introduced by the Dirac delta. Nevertheless, the variational problem is well-posed in the Banach space $W^{1,4/3}(\Omega)\hookrightarrow L^2(\Omega)$, see e.g. [34]. We are hence allowed to consider the solution manifold $\mathcal{S} := \{u_{\boldsymbol{\mu}}\}_{\boldsymbol{\mu}\in\Theta}$ as a subset of the Hilbert space $L^2(\Omega)$.

We analyze two different settings. In the first case we fix the transport field intensity to be $C=0.5$, so that the diffusion and the advection act over the same scale. Then, we consider a transport-dominated case where $C=40$.

[1]https://fenicsproject.org/
[2]https://pytorch.org/

Discretization and Full Order Model

As FOM, we employ Lagrange piecewise linear finite elements over a triangular mesh. Prior to the discretization, we provide a Gaussian approximation of the Dirac delta as

$$f_{\boldsymbol{\mu}}^{\epsilon}(x_1,x_2) := \frac{1}{2\pi\epsilon^2}\exp\left(\frac{-(x_1-\mu_6)^2-(x_2-\mu_7)^2}{2\epsilon^2}\right).$$

We shall write $u_{\boldsymbol{\mu}}^{\epsilon}$ for the solutions of this smoothed problem and $\mathcal{S}^{\epsilon}$ for the corresponding solution manifold. We see that the following claim holds (for the details see the Appendix, Section 5.7)

Claim 1. $\sup_{\boldsymbol{\mu}\in\Theta}\|u_{\boldsymbol{\mu}}-u_{\boldsymbol{\mu}}^{\epsilon}\|_{L^2(\Omega)}\to 0$ as $\epsilon\to 0$.

In particular, $\mathcal{S}^{\epsilon}$ approximates $\mathcal{S}$ uniformly. From here on, we shall fix $\epsilon = 1/420$ and formally replace $\mathcal{S}$ with $\mathcal{S}^{\epsilon}$. Next, we discretize the variational problem through P1-Finite Elements over a triangular mesh. Using the classical estimates from the FEM theory, e.g. [163], it is not hard to see that Assumption 5.1 is satisfied within the state space $L^2(\Omega)$. Here, we fix the mesh size to be $h = 1/210$, which results in a high-fidelity space $V_h \cong \mathbb{R}^{N_h}$ of dimension $N_h = 44521$. We exploit the FOM to generate respectively $N_{train} = 9000$ and $N_{test} = 1000$ random snapshots.

DL-ROM design and training

In the construction of the DL-ROM, we do not make a distinction between the case of mild and strong advection, respectively $C = 0.5$ and $C = 40$. In this way, we can see more clearly whether the intensity of the transport field affects the ROM performance. For the dimensionality reduction, we explore both the two alternatives presented in Section 5.3.1. For the autoencoder, we choose to consider the original solution manifold as a reference for the latent dimension, thus we let $n := n_{\min}(\mathcal{S})$. Thanks to Theorem 5.3, the claim below holds true.

Claim 2. $n_{\min}(\mathcal{S}) = p = 7$.

The proof is straightforward and we leave the details to the Appendix, Section 5.7. Note that in this way, regardless of the encoding strategy, we are fixing the reduced dimension to be $n = 7$. Since $N_h = 44521$, this corresponds to a compression of almost 99.98%.

The networks architectures are detailed in Tables 5.1.a, 5.1.b (encoding step) and Table 5.1.c (decoding step). The encoder and the transcoder are particularly light, as they actually consist of a single dense layer. In contrast, Ψ is far more complex, with a depth of $l = 4$. The proposed architecture is closely related to the ones adopted in [70, 118], upto to some specifics dictated by the problem itself (namely the fact the high-fidelity mesh consists of a 211×211 square). The decoder makes use of transposed convolutional layers, choice that is mainly motivated by two reasons: (i) convolutional layers correspond to sparse operators, and are more easy to deal with in the case of high-dimensional data, (ii) 2D convolutions best describe spatially localized behaviors so they are a natural choice when the data itself is defined on a spatial domain. We shall also remark that the decoder architecture is given in terms of a hyperparameter $m\in\mathbb{N}$, $m > 0$, which controls the number of channels in the convolutional layers. This was done in order to investigate how the network complexity impacts the reconstruction error, and allows for a direct comparison with linear methods such as the POD. We analyze the performance of the networks for different values of m separately, namely $m = 4, 8, 16, 32$.

Prior to training, the networks are initialized differently depending on the encoding type. In the autoencoder case, we initialize both Ψ' and Ψ according to the (Gaussian) He initialization [83]. Conversely, we initialize the transcoder in such a way that $\Psi'_{\mu}(\boldsymbol{\mu},u_{\boldsymbol{\mu}}) = \boldsymbol{\mu}$. This is equivalent to using the parameters as first guess for the intrinsic coordinates: then, during the training, and depending on the decoder needs, Ψ'_{μ} will have the possibility of

Table 5.1: Architectures for the Advection-Diffusion problem (Section 5.4.1). Tables (a) and (b) refer to the encoder and transcoder, respectively. Table (c) reports the decoder architecture. Transp. Conv. = Transposed Convolutional layer, Conv. = Convolutional layer. CNNs hyperparameters are defined along the lines of Definitions 6.1 and 6.2. For a more detailed explanation we refer to the Pytorch library documentation. The complexity of the decoder architecture is tuned accordingly to the value of $m \in \mathbb{N}$. Table (d) refers to the change of coordinates DNN, ϕ. All layers use the 0.1-leaky ReLU activation.

a) Ψ'

Layer	Input	Output	dof
Dense	44521	7	311654

b) Ψ'_μ

Layer	Input	Output	dof
Dense	44528	7	311703

c) Ψ

Layer	Input	Output	Kernel	Stride	dof
Dense	7	$288m$	-	-	$2304m$
Transp. Conv.	$8m \times 6 \times 6$	$4m \times 20 \times 20$	10	2	$3200m^2 + 4m$
Transp. Conv.	$4m \times 20 \times 20$	$2m \times 48 \times 48$	10	2	$800m^2 + 2m$
Transp. Conv.	$2m \times 48 \times 48$	$m \times 102 \times 102$	8	2	$128m^2 + m$
Transp. Conv.	$m \times 102 \times 102$	$1 \times 211 \times 211$	9	2	$81m + 1$

d) ϕ

Layer	Input	Output	dof
Dense	7	1024	7168
Dense	1024	512	524288
Dense	512	256	131072
Dense	256	7	1792

finding other representations. As discrepancy measure for the loss function, we use squared errors, $\ell(\mathbf{y}, \hat{\mathbf{y}}) = \|\mathbf{y} - \hat{\mathbf{y}}\|^2$. We train the autoencoder (resp. transcoder-decoder) using the AdamW optimizer [125], with learning rate 10^{-4}, weight-decay 10^{-2}, moments coefficients $\beta_1 = 0.99$, $\beta_2 = 0.999$ and adjustment $\varepsilon = 10^{-8}$. We perform the gradient descent using batches of 50 and for a total of 1000 epochs. At the end of this first training session, we pick the best performing architecture and continue the construction of the DL-ROM from there.

Table 5.1.d reports the architecture for our third network, ϕ. We initialize ϕ using the He initialization and proceed with its training according to Section 5.3.2. We consider again a loss function based on square errors, and we perform the gradient descent using the same optimizer as before, only changing the learning rate to 10^{-3}.

Numerical results

Figure 5.4 reports the results limited to the dimensionality reduction, that is the first step in the DL-ROM pipeline. There, we compare the performance of autoencoders, transcoder-decoders and POD in terms of model complexity. In general, regardless of whether $C = 0.5$ or $C = 40$, both nonlinear methods show interesting results, with training errors close or below 1%. Unsurprisingly, as the networks grow in complexity, the gap between training and test errors becomes larger, highlighting the need for more samples and a tendency towards overfitting (see the autoencoder in case $C = 40$). Still, transcoders seem to mitigate this phenomenon, possibly because they provide more information in the latent space.

For POD, the degrees of freedom are defined as the number of entries in the projection

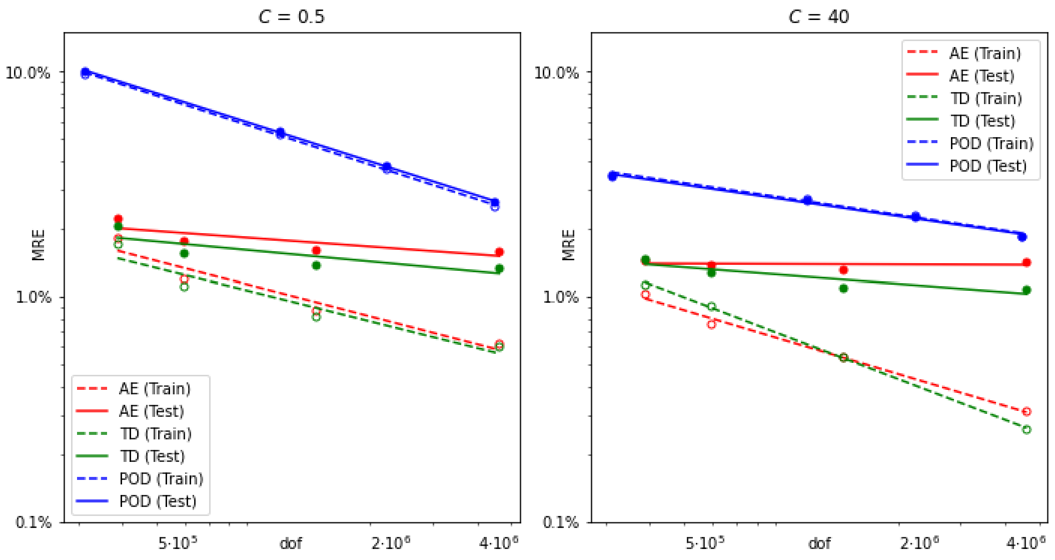


Figure 5.4: Error decay in terms of network complexity for the Advection-Diffusion problem (Section 5.4.1). Plots are reported in loglog scale. Case $C = 0.5$ on the left, $C = 40$ on the right. Lines are drawn by considering the least-square fit $\log \mathrm{MRE} \approx \beta_0 + \beta_1 \log \mathrm{dof}$. Dashed-lines = training errors, straight lines = test errors. AE = autoencoder (red), TD = transcoder-decoder (green). The MREs reported refer to the architectures in Table 5.1 with $m = 4, 8, 16, 32$. POD-projection errors are reported in blue. POD degrees of freedom (dof) are computed as nN_h and correspond to the number of entries in the projection matrix $\mathbf{V}$. Remark: for POD the reduced dimension increases with the complexity. Conversely, in the DL-ROM approach the reduced dimension is always fixed to $n = p = 7$.

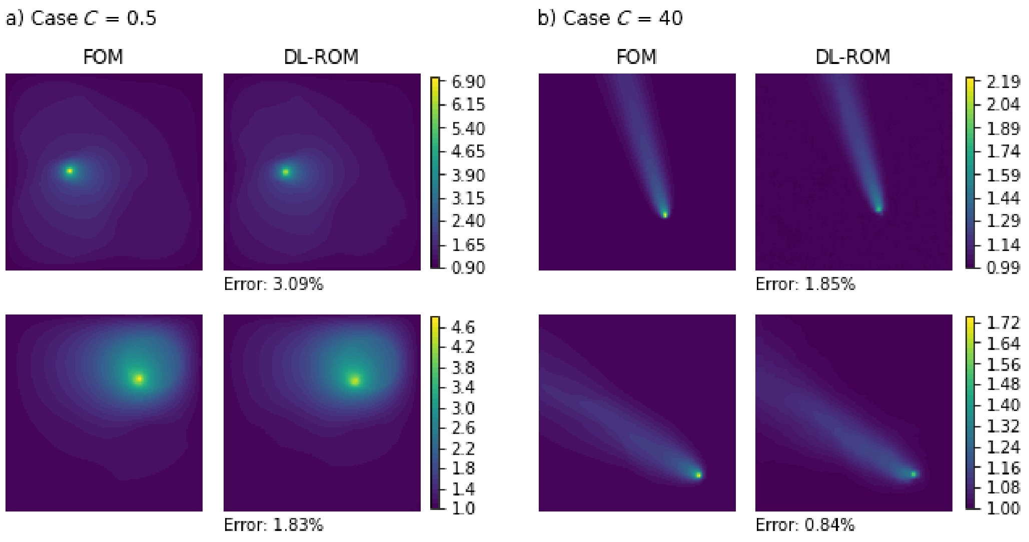


Figure 5.5: DL-ROM results for the Advection-Diffusion problem (Section 5.4.1). Panels (a) and (b) respectively refer to the case of mild and strong advection. In both cases, the picture shows two examples extracted from the test set (one for each row) and compares the high-fidelity solution (first column) with the DL-ROM approximation (second column).

C	Data	MRE	Equivalent POD modes
0.5	Train	1.05%	428
0.5	Test	2.01%	164
40	Train	0.31%	more than 1000
40	Test	1.23%	316

Table 5.2: DL-ROM performance for the Advection-Diffusion problem (Section 5.4.1). The final model $\Phi := \Psi(\phi)$ was constructed by choosing the decoder with highest performance (cf. Figure 5.4). Equivalent POD modes = minimum number of POD-modes needed by any POD-based ROM to outperform the DL-ROM.

matrix $\mathbf{V}$, while the errors are computed as $\|\mathbf{u}-\mathbf{V}\mathbf{V}^T\mathbf{u}\|/\|\mathbf{u}\|$ (relative projection error). In particular, the MREs reported in Figure 5.4 provide a lower-bound for all POD-based ROMs. Interestingly, all the curves show a similar trend. This goes to support our conjecture that the decoder complexity may be linked with the Kolmogorov n-width (Section 5.3). More precisely, linear methods can improve the accuracy by adding Δn modes, i.e. $\Delta n N_h$ degrees of freedom, but they also have to increase the ROM dimension. Conversely, in the DL-ROM approach, we can obtain a similar boost by investing the same degrees of freedom in the decoder, without having to modify the latent dimension. This is in agreement with Theorem 5, where we proved that $\mathcal{O}(m^{1+n/(s-1)}\log(m))$ active weights are sufficient for the decoder to match the accuracy of any projection method with m modes. Of note, if we assume the solution manifold to be infinitely smooth and we let $s \to +\infty$, then we may conjecture the decoder complexity to behave as $\mathcal{O}(m\log(m))$. As matter of fact, this is what we observe in the picture, at least for the training errors. In fact, if the red lines were to be perfectly parallel to the blue ones, that would reflect a scenario in which the decoder complexity grows as $\mathcal{O}(m)$.

On the same time, Figure 5.4 goes to show that the upper bounds in Theorem 5 are suboptimal. In fact, both in the case of mild and strong advection, the nonlinear reduction is able to outperform POD with fewer degrees of freedom, i.e. without the extra contribute mN_h in the decoder. As we shall discuss in the next Chapter, this is made possible by the use of convolutional layers.

Let us now move to the actual approximation of the parametric map. To this end, we trained our third network ϕ on the basis of the best performing transcoder-decoder ($m = 32$). Numerical results for the complete DL-ROM $\Phi := \Psi \circ \phi$ are in Table 5.2 and Figure 5.5. In general, the results are satisfactory, with test errors near 2%. Both in the case of mild and strong advection, we note that POD-based ROMs require more than 300 modes to achieve the same accuracy. This makes intrusive ROMs, such as POD-Galerkin, too expensive to be used online. Conversely, the DL-ROM approach provides an appealing alternative. Indeed, while the whole offline stage took around 4 hours, the model is extremely fast when used online: solving the PDE for 1000 different values of the parameters (simultaneously) requires less than 2 milliseconds on GPU.

5.4.2 Stochastic Poisson equation

Problem definition

On the spatial domain $\Omega = (0,1)^2$, we consider a Gaussian process W with constant mean $w = -\log(10)$ and covariance kernel $\text{Cov}(\mathbf{x},\mathbf{y}) = 10\exp(-4|\mathbf{x}-\mathbf{y}|^2)$. The latter is used to model the stochastic Poisson equation below,

$$\begin{cases} -\nabla \cdot \left(e^{W(\omega)}\nabla u\right) = |\mathbf{x}|^2 & \text{in } \Omega, \\ u = 0 & \text{on } \partial\Omega. \end{cases}$$

Here, for each event ω, the map $W(\omega):\Omega\to\mathbb{R}$ denotes the corresponding path of the stochastic process W. The above problem can be seen as a parameter dependent PDE that depends on (countably) infinite many parameters. To see this, we recall that there exist positive real numbers $\{\sqrt{\lambda_i}\}_{i=1}^{+\infty}$, orthonormal functions $\{\zeta_i\}_{i=1}^{+\infty}\subset L^2(\Omega)$ and independent standard gaussians $\{X_i\}_{i=0}^{+\infty}$ such that

$$W = w + \sum_{i=1}^{+\infty}\sqrt{\lambda_i}X_i\zeta_i$$

almost surely. The latter is the so-called Karhunen-Loève expansion of W. We assume the λ_i coefficients to be nonincreasing in i. In order to cast the problem into our framework, we approximate W by truncating the aforementioned expansion at some index k. More precisely, we define $\Theta:=\mathbb{R}^k$ and $W_{\boldsymbol{\mu}}^k$ as

$$W_{\boldsymbol{\mu}}^k(x) := w + \sum_{i=1}^{k}\sqrt{\lambda_i}\mu_i\zeta_i(x).$$

Thanks to the usual continuity results and the convergence ensured by the Karhunen-Loève expansion, the impact of this substitution on the PDE can be made arbitrarily small with k. We note that, by construction, the probability distribution to be considered over the parameter space is the Gaussian distribution $\mathbb{P}$ of density

$$G(\boldsymbol{\mu}) := (2\pi)^{-k/2}\,\mathrm{e}^{-\frac{1}{2}|\boldsymbol{\mu}|^2}.$$

Discretization and Full Order Model

On Ω we define a triangular mesh of size $h=10^{-2}$, over which we construct the high-fidelity space of piecewise linear Finite Elements V_h. The corresponding FOM dimension is $N_h=10121$. To approximate the Karhunen-Loève expansion of W, we project and solve over V_h the following eigenvalue problem.

$$\int_\Omega \mathrm{Cov}(\mathbf{x},\mathbf{y})\zeta_i(\mathbf{y})d\mathbf{y} = \lambda_i\zeta_i(\mathbf{x}).$$

In particular, we compute the first k eigenvalues λ_i and corresponding eigenfunctions $\zeta_i\in V_h$ for which

$$0.9 \le \frac{\sum_{i=1}^{k}\lambda_i}{\sum_{i=1}^{+\infty}\lambda_i} = \frac{1}{10}\sum_{i=1}^{k}\lambda_i,$$

where the last equality is easily deduced by the covariance kernel, as $\sum_{i=1}^{+\infty}\lambda_i=\int_\Omega\mathrm{Cov}(\mathbf{x},\mathbf{x})d\mathbf{x}=\int_{[0,1]^2}10d\mathbf{x}=10$. Figure 5.6 shows how the normalized eigenvalues $\lambda_k/10$ decay with k. This procedure results in the choice of the truncation index $k=38$. From a statistical point of view, we say that $W_{\boldsymbol{\mu}}^k$ explains at least 90% of the variability in W. We run the FOM to generate respectively $N_{\text{train}}=9000$ and $N_{\text{test}}=1000$ snapshots, where the parameter values are sampled from Θ independently and according to a k-variate standard Gaussian distribution.

DL-ROM design

We note that, in this case, the parameter space Θ is not compact, as it is unbounded. Nevertheless, since Θ has finite measure with respect to $\mathbb{P}$, it is straightforward to adapt the reasoning in Section 5.3 to this context. For instance, the error defined in (5.6) can be made arbitrarily small provided that Φ is sufficiently accurate within some compact subdomain $\Theta_M:=\{\boldsymbol{\mu}\in\mathbb{R}^p \text{ s.t. } |\boldsymbol{\mu}|<M\}$. For a more in depth discussion about the regularity of the parametric map in the case of stochastic coefficients we refer to [8].

For the dimensionality reduction, we employ a transcoder-decoder. This is to ensure a maximal compression, as the number of parameters is already mildly large. The network

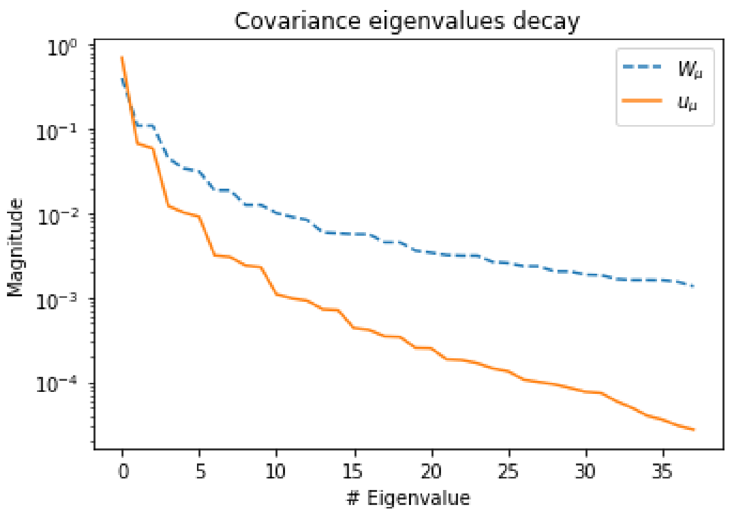


Figure 5.6: Normalized eigenvalues decay for the covariance matrices of the parameters, i.e. the random field $W_{\boldsymbol{\mu}}$, and the solutions $u_{\boldsymbol{\mu}}$. The eigenvalues are normalized with respect to the total variance of the two fields, so that their cumulative sum converges to 1. The y-axis is reported in logarithmic scale. Note that the solutions are sampled by considering the truncated field $W^k_{\boldsymbol{\mu}}$ as parameter.

topology is reported in Tables 5.3.a, 5.3.b. Coherently with the chosen approach, we fix the reduced dimension to be $n := k = p = 38$. In general, the architecture is very similar to those considered in Section 5.4.1, the only difference being in the specifics of the convolutional layers.

As before, we adopt the He initialization for the decoder while we force the initial state of the transcoder to behave as $\Psi'_\mu(\boldsymbol{\mu}, u_{\boldsymbol{\mu}}) = \boldsymbol{\mu}$. We train $\Psi \circ \Psi'_\mu$ using stochastic gradient descent with minibatches of size 10 and for a total of 1200 epochs. In this case, we observe that snapshots come in rather different scales when compared one another. For this reason, we choose to define the loss function in terms of relative errors, $\ell(\mathbf{y}, \hat{\mathbf{y}}) := \|\mathbf{y} - \hat{\mathbf{y}}\| / \|\mathbf{y}\|$. To optimize the latter we employ the Adamax optimizer [102], with default parameters and learning rate of 10^{-3}. Here, the choice of Adamax over AdamW is motivated by the fact that the former is known to be more stable. Table 5.3.c reports the architecture for reduced map network, ϕ. We train ϕ using the Adamax optimizer with batch size 50 and learning rate 5e-3, for a total of 5000 epochs. Here also we use relative errors as discrepancy measures. The whole offline stage of the DL-ROM took around 4 hours.

Numerical results

The dimensionality reduction is satisfactory, with mean relative errors of 1.10% and 2.57% respectively on the training and test sets. Conversely, the approximation of the reduced map was more challenging, see Table 5.4 and Figure 5.7. While the final model is able to approximate the parameter-to-state map with an error of 4.69% over the training set, the inaccuracy increases to 12.50% on the test set. This is a situation in which the solution manifold is relatively simple (even linear subspaces provide good approximations, cf. Figure 5.6), but the parameter dependency is complicated. Therefore, while 9000 snapshots are sufficient for the training the transcoder-decoder, they are not enough for ϕ to generalize well. Another reason is that the parameter space is very large, and ϕ has to face the curse of dimensionality. Possible ways to overcome this drawback without having to generate more samples would be to exploit low-discrepancy sequences, as in [137], or use physics-informed approaches at the reduced level, as in [41].

Table 5.3: Architectures for the Stochastic Poisson equation (Section 5.4.2). Tables (a) and (b) together describe the transcoder-decoder, while (c) concerns the change of coordinates map ϕ. All layers are considered with the 0.1-leaky ReLU activation.

a) Ψ'_{μ}

Layer	Input	Output	dof
Dense	10239	38	389120

b) Ψ

Layer	Input	Output	Kernel	Stride	dof
Dense	38	18432	-	-	718848
Transp. Conv.	$512 \times 6 \times 6$	$256 \times 12 \times 12$	2	2	524544
Transp. Conv.	$256 \times 12 \times 12$	$128 \times 24 \times 24$	2	2	131200
Transp. Conv.	$128 \times 24 \times 24$	$64 \times 48 \times 48$	2	2	32832
Transp. Conv.	$64 \times 48 \times 48$	$1 \times 101 \times 101$	7	2	3137

c) ϕ

Layer	Input	Output	dof
Dense	38	256	9984
Dense	256	128	32896
Dense	128	64	8256
Dense	64	38	2470

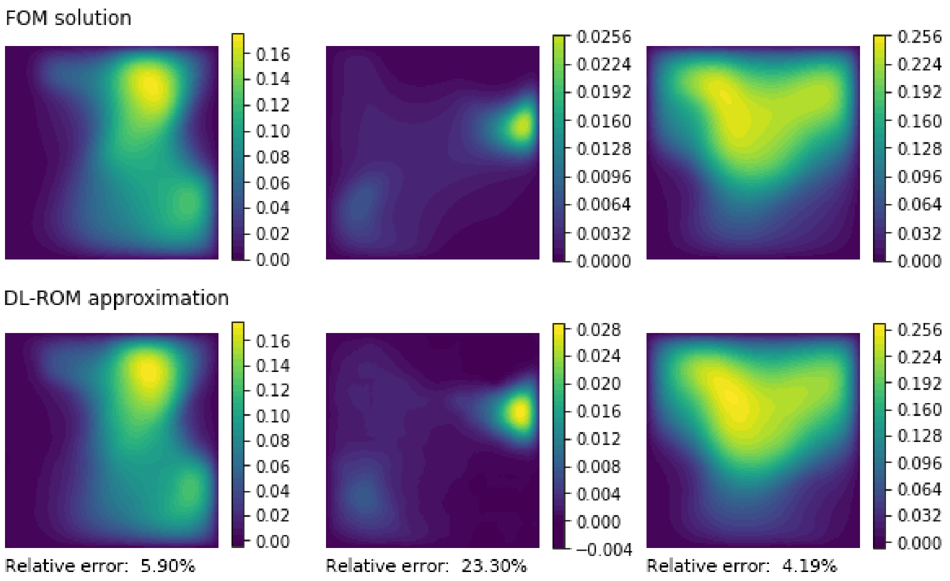


Figure 5.7: DL-ROM results for the Stochastic Poisson equation (Section 5.4.2). The picture shows three examples coming from the test set. The first row reports the high-fidelity solutions, while the second row displays the corresponding DL-ROM approximations. Relative errors are also reported.

Data	MRE	Equivalent POD modes
Train	4.69%	39
Test	12.50%	17

Table 5.4: DL-ROM performance for the Stochastic Poisson problem (Section 5.4.2). Equivalent POD modes = minimum number of POD-modes needed by any POD-based ROM to outperform the DL-ROM.

5.4.3 On alternative Deep Learning approaches

The Reduced Order Modelling literature is becoming more and more flourishing, with a large variety of techniques being developed. It is thus important to understand how the DL-ROM approach relates to other Deep Learning based ROMs. In particular, we would like to comment on those alternative strategies that are significantly different in spirit. One approach, is to directly approximate the correspondence $(\boldsymbol{\mu}, \mathbf{x}) \to u_{\boldsymbol{\mu}}(\mathbf{x})$ using a single DNN, namely $\Phi : \mathbb{R}^{p+d} \to \mathbb{R}$. This has the advantage of yielding a mesh-free ROM that can be trained on pointwise observations. Also, since Φ is scalar-valued, light architectures are expected to be sufficiently expressive. However, this approach has a few drawbacks. In fact, if $u_{\boldsymbol{\mu}}$ is not highly regular with respect to $\mathbf{x}$, e.g. not Lipschitz, then one may require a very deep (or large) architecture to obtain reliable approximations, and thus many samples too. For example, we have tested this approach on the Advection-Diffusion problem using a scalar-valued architecture with 7 hidden layers of constant width (50 neurons). We have used the same snapshots available for the DL-ROM, where each high-fidelity solution now contributed with a total of N_h observations. Despite all our efforts, we were unable to obtain a sufficiently accurate network, as we always obtained relative errors above 20% (both for the training and test sets). We believe that the main drawback is given by the singular source, which generates high gradients in the PDE solution. We also mention that, despite the light architecture, training Φ was quite expensive. In fact, each computation of the squared error $\|u_{\boldsymbol{\mu}}^h - \Phi(\boldsymbol{\mu}, \cdot)\|_{L^2(\Omega)}^2$ required N_h evaluations of the DNN. Even though these can be computed in parallel by stacking all quadrature nodes in a single matrix, the cost of the backpropagation step increases substantially, as keeping track of all the gradients becomes challenging.

An alternative strategy is given by DeepONets [126], which are now becoming very popular. DeepONets are primarily employed for learning operators in infinite dimensions, but they have a natural adaptation to the case of finite-dimensional parameters. In fact, the first step in the DeepONet pipeline is to encode the input through p sensors, which allows us to formally recast the problem into one with p scalar parameters. With this set up, DeepONets are still a mesh-free approach, but they consider an approximation of the form $u_{\boldsymbol{\mu}}(\mathbf{x}) = \Psi(\mathbf{x}) \cdot \phi(\boldsymbol{\mu})$, where $\cdot$ is the dot product, while $\Psi : \mathbb{R}^d \to \mathbb{R}^n$ and $\phi : \mathbb{R}^p \to \mathbb{R}^n$ are two neural networks, respectively called the *trunk net* and the *branch net*. The main advantages of DeepONets are the following. First of all, as they are intrinsically mesh-free, it is possible to train them on sparse pointwise data. Secondly, as they decouple the dependency between $\boldsymbol{\mu}$ and $\mathbf{x}$, it is possible to bound their complexity and to estimate their generalization capabilities in ways that are specific to this approach, see e.g. [115]. Finally, due to their original construction, they can be a natural choice when the input parameters are actually sensor observations of some functional input. However, DeepONets have their limitations as well. In fact, despite sharing some terminology with the DL-ROM approach, such as *encoder* and *decoder*, they ultimately rely on a linear strategy for representing solutions. To see this, let $\{\mathbf{x}_i\}_{i=1}^{N_h}$ be the nodes in the high-fidelity mesh. Then, the DeepONet approximation over these vertices is $\mathbf{V}\phi(\boldsymbol{\mu})$, where $\mathbf{V} := [\Psi(\mathbf{x}_1), \dots, \Psi(\mathbf{x}_{N_h})]^T \in \mathbb{R}^{N_h \times n}$. As a consequence, the choice of n is subject to the behavior of the linear Kolmogorov n-width. For instance, to match the DL-ROM accuracy in the Advection-Diffusion problem with $C = 40$, a DeepONet architecture would require $n \geq 300$, which may hinder its actual implementation. Also, due to the poor regularity with respect to the $\mathbf{x}$ variable, training Ψ

may be a challenging task.

Conversely, the DL-ROM approach treats solutions as single objects, $\mathbf{u}_{\boldsymbol{\mu}}^h \in V_h$. While this clearly results in a loss of information, the space dependency of solutions can be partially recovered by interchanging nonlocal and local operators (respectively, dense and convolutional layers) in the ROM pipeline. Finally, thanks to the use of nonlinear reduction techniques, the DL-ROM can overcome some of the difficulties implied by the Kolmogorov n-width. Of course, though, our approach has some limitations too. First of all, it is mesh-constrained, as it is bounded to the existence of a high-fidelity model. Secondly, it mostly relies on convolutional layers, which makes it less obvious to adapt the current implementation to non-cubic domains. Finally, the approach was originally designed for the case $p \ll N_h$. Even though infinite-dimensional parameters spaces can be handled as in Section 5.4.2, better strategies may be available, see e.g. the recent work in [69].

5.5 A first application to oxygen transfer models

Model description

Let $\Omega : [0,1]^2$ be the unit square, which we use to represent a small portion of biological tissue, and let $u : \Omega \to [0,+\infty)$ be the oxygen distribution all over the domain. We assume that the tissue is perfused by ν vessels, $\Lambda_i \subset \Omega$, $i = 1,\dots,\nu$, which we represent as straight lines of length $\geq l_0$. We model the oxygen transfer between the vessels and the tissue with the oversimplified diffusion equation below,

$$\begin{cases} -\Delta u = \sum_{i=1}^{\nu} \alpha_i \delta_{\Lambda_i} & \text{in } \Omega \\ -\nabla u \cdot \mathbf{n} = \beta u & \text{on } \partial\Omega \end{cases} \tag{5.7}$$

which we understand in the weak sense. Here, β is a fixed constant associated to the Robin boundary condition. Conversely, for each $i = 1,\dots,\nu$, we have denoted by δ_{Λ_i} the singular measure whose action is characterized by the following identity

$$\int_\Omega v(\mathbf{x})\delta_{\Lambda_i}(d\mathbf{x}) = \int_{\Lambda_i} v(\mathbf{s})d\mathbf{s}, \tag{5.8}$$

holding for all $v \in \mathcal{C}(\overline{\Omega})$. Finally, the coefficients α_i describe the amount of oxygen carried by each vessel alone.

Over all, the model parameters are: the total number of capillaries $\nu \in \mathbb{N}$, the location of the blood vessels Λ_i, which we let vary in

$$\mathscr{V} := \left\{\Lambda = \{t\mathbf{p}_1 + (1-t)\mathbf{p}_2\}_{t\in[0,1]} \text{ s.t. } \mathbf{p}_i \in \partial\Omega,\ \Lambda \cap \partial\Omega = \{\mathbf{p}_1, \mathbf{p}_2\},\ |\mathbf{p}_1 - \mathbf{p}_2| \geq l_0\right\},$$

and the intensities $\alpha_i \geq 0$. Given $\{\Lambda_i\}_{i=1,\dots,\nu} \subset \mathscr{V}$ and $\{\alpha_i\}_{i=1,\dots,\nu} \subset [0,+\infty)$, with little abuse of notation let us write $u_{\alpha_1\Lambda_1,\dots,\alpha_\nu\Lambda_\nu}$ for the corresponding solution to (5.7). By linearity of (5.7), it is straightforward to see that

$$u_{\alpha_1\Lambda_1,\dots,\alpha_\nu\Lambda_\nu} = \sum_{i=1}^{\nu} \alpha_i u_{\Lambda_i}. \tag{5.9}$$

In other words, it is sufficient to study the case of a single vessel with normalized intensity in order to obtain a comprehensive understanding of (5.7). Of course, this comes from the fact that the latter model is extremely oversimplified. Nevertheless, the consequence is that, in order to build a suitable ROM, we can restrict to the following parameter space, and correspond solution manifold,

$$\Theta := \mathscr{V}, \qquad \mathcal{S} := \{u_\Lambda\}_{\Lambda\in\Theta}.$$

Discretization and Full Order Model

In order to discretize system (5.7), we first provide an approximation to the line integral in (5.8), which we employ to replace the 1D-2D formulation. In particular, we consider the smoothed approximation below,

$$\int_{\Lambda} v(\mathbf{s})d\mathbf{s} \approx \int_{\Omega} v(\mathbf{x})\sigma_{\Lambda}(\mathbf{x})d\mathbf{x}, \tag{5.10}$$

where

$$\sigma_{\Lambda}(\mathbf{x}) := \frac{1}{\epsilon^2}\max\left\{\epsilon - \text{dist}(\mathbf{x},\Lambda), 0\right\}, \quad \text{with} \quad \text{dist}(\mathbf{x},\Lambda) := \inf_{\mathbf{y}\in\Lambda}|\mathbf{x}-\mathbf{y}|$$

Here, $\epsilon > 0$ is a smoothness parameter that we fix to $\epsilon = 0.01$. It is not hard to prove that, for each fixed $v \in \mathcal{C}(\overline{\Omega})$, the right-hand side of equation (5.10) converges to the left hand-side as $\epsilon \downarrow 0^+$. We refer to the appendix for a detailed and more general proof: see Lemma 5.1, where Λ is allowed to be any finite union of straight segments. Finally, in order to discretize the PDE, we employ continuous P1 finite elements on a structured triangular grid consisting of $N_h = 10201$ nodes. Finally, we endow the Θ with a uniform probability distribution.

DL-ROM design

In the single vessel scenario, the oxygen model in (5.7) only depends on two scalar parameters, $p = 2$, as the only variable entity is $\Lambda \in \mathscr{V}$, whose location is uniquely determined by the extremes $\mathbf{p}_1, \mathbf{p}_2$, both moving across the one-dimensional manifold $\partial\Omega$. Therefore, our rule of thumb would suggest picking a latent dimension of $2 \leq n \leq 5$ for our autoencoder approach. However, in this case one can be more precise and say that $n_{\min}(\mathcal{S}) \leq 3$. Furthermore, the infimum in Equation (5.5) is attained for $n = 3$ but not for $n = 2$, which suggests choosing the former to ensure a much stable optimization of the autoencoder architecture. These observations come from the fact that, in this case, the solution manifold is topologically equivalent to the so-called Möbius strip, a well-known example of a nonorientable surface that can be embedded in $\mathbb{R}^3$ but not in $\mathbb{R}^2$ (see Figure 5.8 for a "visual proof" of such equivalence). In light of this, we proceed with the autoencoder approach, letting the latent dimension be equal to $n = 3$. We design the architectures of Ψ', Ψ and ϕ along the same lines of the previous experiments, cf. Table 5.5. This time, however, we introduce an exogenous layer in the third network, ϕ, to enforce our a priori knowledge of the problem. The idea goes as follows. Suppose that we parametrize the vessel Λ using the extremes $\mathbf{p}_1$ and $\mathbf{p}_2$. Then, we can describe each $\mathbf{p}_i$ by the angle $\theta_i \in [0, 2\pi]$ formed by the revolution of the point around the center of the square. With this parametrization, the reduced map becomes $\phi = \phi(\theta_1, \theta_2)$. However, if we let ϕ be any DNN, we lose the guarantee that same basic properties of the FOM are reflected by the ROM. For instance, we would like ϕ to be periodic and symmetric with respect to its arguments, that is

$$\phi(\theta_1 + 2k\pi, \theta_2 + 2j\pi) = \phi(\theta_1, \theta_2) = \phi(\theta_2, \theta_1) \tag{5.11}$$

for all $k, j \in \mathbb{Z}$. In fact, there is an intrinsic redundancy in the way the angles θ_1 and θ_2 parametrize Λ: swapping the extreme points yields the same vessel, as well as performing a full rotation of 360°. To account for this, we enforce a better parametrization of Λ in the first layer of ϕ: we let $\phi = \tilde{\phi} \circ \phi_0$, where $\tilde{\phi} : \mathbb{R}^4 \to \mathbb{R}^n$ is a DNN, while ϕ_0 is defined as

$$\phi_0(\theta_1, \theta_2) \to \begin{bmatrix} \cos\theta_1 + \cos\theta_2 \\ \cos\theta_1 \cdot \cos\theta_2 \\ \sin\theta_1 + \sin\theta_2 \\ \sin\theta_1 \cdot \sin\theta_2 \end{bmatrix}.$$

The advantage of this transformation is twofold. On the one hand, it ensures that (5.11) holds. On the other hand, it results in no loss of information as it is always possible to recover the (unordered) pair $\{\theta_1, \theta_2\}$ given the output $\phi_0(\theta_1, \theta_2)$.

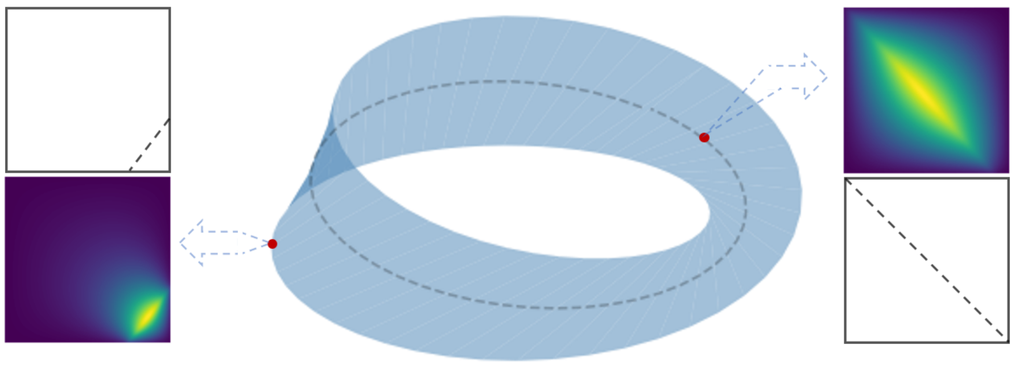


Figure 5.8: Visual representation of the single-vessel solution manifold, Section 5.5. In this case, parameters and solutions are in a one-to-one correspondence, and the solution manifold is topologically equivalent to a Möbius strip. Points along the center line of the strip correspond to vessels that pass through the center of the domain, while boundary points represent configurations where the vessel reaches the minimal length allowed. Traveling across the boundary of the Möbius strip requires two loops, while a single revolution is sufficient at the center line: similarly, rotating a small vessel around the domain center requires a full circle, while a half twist is sufficient for vessels crossing the center. Intuitively, the nonorientability of the Möbius strip is reflected by the fact that, in the solution manifold, swapping the extremes of a vessel leaves the situation unchanged (in analogy to the so-called symmetric product of the circle, see e.g. [29]).

Finally, concerning the model training, we train the DL-ROM on $N_{\text{train}} = 1500$ random snapshots and test its performance over another $N_{\text{test}} = 500$ instances. This time, we train all the architectures with the L-BFGS optimizer, with learning rate equal to 1, no batching and for a total of 500 epochs.

Results

Results are shown in Figure 5.9 and Table 5.6, where we compare DL-ROM approximations and FOM solutions. We mention that, while the results reported in Table 5.6 correspond to the single vessel scenario (coherently with our training strategy), the pictures in Figure 5.9 refer to the general case, where we exploited the superposition formula in Equation (5.9).

In general, the performance is satisfactory and the DL-ROM achieves an L^2-relative error of 5% over the test set. As for the previous experiments, we also report the linear benchmark associated to POD-based ROMs. In general, the results are in agreement with our findings in Section 5.4: the DL-ROM approach is able to recover the ground truth with good accuracy, preserving the main properties of the solutions; conversely, linear methods encounter significant difficulties and require a larger number of modes to capture all the relevant features. In terms of training cost, thanks to the L-BFGS optimizer, which allowed us to avoid the use of batching strategies, we were able to train all the networks in the DL-ROM pipeline in less than 15 minutes (GPU time).

Nevertheless, we must mention that building such a reduced order model for (5.7) is, without any doubt, redundant and of little interest. In fact, (5.7) describes an oversimplified scenario that carries no useful insights for the clinical applications. Furthermore, at this level, there is no need to introduce a surrogate model, as the FOM can already be made extremely efficient. In fact, one could pre-assemble the stiffness matrix associated to the diffusion operator and, online, only assemble the right-hand-side vector with little computational cost. We have presented this example only for didactic purposes, highlighting how DL-ROMs can be of help where linear methods fail. In the next Chapters, we shall consider more complicated (but also more meaningful) models for oxygen transfer, and build

a) Ψ'_{μ}

Layer	Input	Output	dof
Dense	10201	3	30606

b) Ψ

Layer	Input	Output	Kernel	Stride	dof
Dense	3	400	-	-	1600
Dense	400	1152	-	-	461952
Transp. Conv.	$128 \times 3 \times 3$	$64 \times 14 \times 14$	8	3	524352
Transp. Conv.	$64 \times 14 \times 14$	$32 \times 47 \times 47$	8	3	131104
Transp. Conv.	$32 \times 47 \times 47$	$1 \times 101 \times 101$	9	2	2593

c) ϕ

Layer	Input	Output	dof
Exogenous	3	4	0
Dense	4	50	250
Dense	50	50	2550
Dense	50	50	2550
Dense	50	3	153

Table 5.5: Architectures for the simplified oxygen transfer model (Section 5.5). Tables (a) and (b) report the autoencoder architecture, while (c) concerns the change of coordinates map ϕ. All layers are considered with the 0.1-leaky ReLU activation.

Data	MRE	Equivalent POD modes
Train	4.49%	74
Test	5.34%	62

Table 5.6: DL-ROM performance for the simplified oxygen transfer model (Section 5.5). Equivalent POD modes = minimum number of POD-modes needed by POD-based ROMs to outperform the DL-ROM. MRE = Mean relative error, computed accordingly to the L^2-norm.

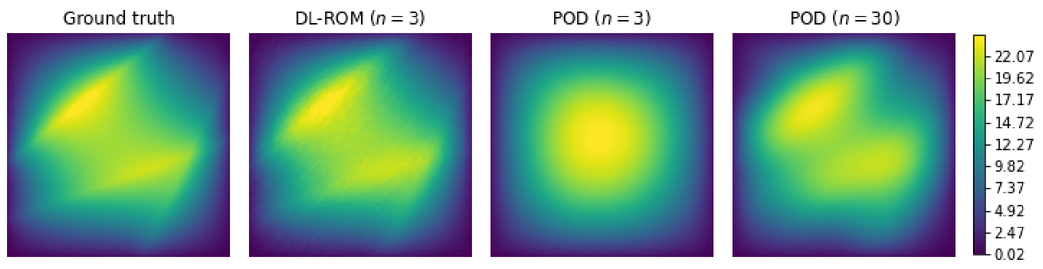


Figure 5.9: Comparison of different ROMs for the simplified oxygen model, Equation (5.7). The ground truth is $u = u_{\lambda_1} + u_{\lambda_2} + 2u_{\Lambda_3}$, where all the Λ_i come from the test set. Here, n is the reduced dimension, respectively for the DL-ROM and the POD approach. Relative errors are: 1.63% (DL-ROM), 14.34% (POD, $n = 3$) and 3.63% (POD, $n = 30$), respectively.

corresponding strategies to tackle them. In fact, the as-is DL-ROM will not be sufficient, the reason being a significant increase in the parametric dimension p. For instance, at the end of Chapter 7, we will consider a more realistic setting in which Λ is a generic piecewise linear graph with a seemingly biological structure. In that case, larger values of p are required to parametrize the network architecture, which, as in Section 5.4.2, may affect the approximation power of the DL-ROM.

5.6 Conclusions

In this Chapter, we developed a novel Deep Learning approach for reduced order modelling of parameter dependent PDEs, DL-ROM, where the solution map is approximated by a deep neural network Φ. Our construction is based on the use of autoencoders, which we employ as a nonlinear alternative to other reduction techniques such as the POD. In the DL-ROM approach, we choose the latent dimension to be the smallest one granting arbitrary accuracy. The value of such dimension was investigated in detail in Section 5.2. There, we proved some theoretical results, respectively Theorem 5.3 and Theorem 5.4, that can be used as guidelines for practical applications. Further insights on the potential of the DL-ROM approach were discussed in Theorem 5.5, Section 5.3. There, we provided explicit error estimates that were later confirmed via empirical evidence (cf. Section 5.4.1).

The results obtained in our experiments are promising. The DL-ROM appears to be a captivating alternative to traditional ROMs, especially in challenging situations where linear models fail. Our first test case, Section 5.4.1, shows that the method behaves well in the presence of singularities, both under diffusive or transport-dominated circumstances. Good results are also obtained for high dimensional parameter spaces, Section 5.4.2, even though it becomes harder to handle the generalization error. The latter can be either improved by increasing the number of training samples or by including physical terms in the loss function. While we wish to investigate this further in future works, we acknowledge that multiple researchers are now working on this topic, e.g. [41, 136].

In principle, being completely nonintrusive and data-driven, the proposed approach can be readily applied to nonlinear PDEs and more complicated systems. Also, at the cost of treating time as an additional parameter, as in [70], one may extend the DL-ROM approach to time dependent problems. However, some changes have to be made in order to extrapolate over time, for instance by enforcing those properties that are typical of dynamical systems (e.g. the existence of underlying semi-groups). We leave all these considerations for future works.

We conclude with a few comments on the computational cost. While the offline stage is clearly expensive, our design choices allow for a significant reduction in the model complexity, which results in architectures that are easier to train (cf. e.g. [70, 72]). Nevertheless, the DL-ROM is extremely fast when used online. This makes the method suited for demanding tasks with multiple queries, as the ones typical of sensitivity analysis, uncertainty quantification and multiscale methods.

5.7 Technical proofs

Proofs of the claims in Section 5.4

Proof of Claim 1. Let $\boldsymbol{\mu} \in \Theta$. For the sake of brevity, define $\mathbf{x}_{\boldsymbol{\mu}} := (\mu_6, \mu_7) \in \Omega$. We shall recall that, by Morrey's embedding theorem [60], we have $W_0^{1,4}(\Omega) \hookrightarrow \mathcal{C}^{0,1/2}(\Omega)$, the latter being the space of 1/2-Hölder maps. As a consequence, for any $w \in W_0^{1,4}(\Omega)$, we have

$$\left|w(\mathbf{x}_{\boldsymbol{\mu}}) - \int_\Omega f_{\boldsymbol{\mu}}(\mathbf{z})^\epsilon w(\mathbf{z})d\mathbf{z}\right| = \left|\int_\Omega (w(\mathbf{x}_{\boldsymbol{\mu}}) - w(\mathbf{z}))f^\epsilon_{\boldsymbol{\mu}}(\mathbf{z})d\mathbf{z}\right| \le$$
$$\le \int_\Omega |w(\mathbf{x}_{\boldsymbol{\mu}}) - w(\mathbf{z})| f^\epsilon_{\boldsymbol{\mu}}(\mathbf{z})d\mathbf{z} \le C'\|w\|_{W_0^{1,4}(\Omega)} \int_{\mathbb{R}^2} |\mathbf{x}_{\boldsymbol{\mu}} - \mathbf{z}|^{1/2} f^\epsilon_{\boldsymbol{\mu}}(\mathbf{z})d\mathbf{z}$$

for a constant $C' > 0$ independent on both w and $\boldsymbol{\mu}$. The change of variables $\mathbf{y} := (\mathbf{z} - \mathbf{x}_{\boldsymbol{\mu}})/\epsilon$ then yields

$$\left| w(\mathbf{x}_{\boldsymbol{\mu}}) - \int_{\Omega} f_{\boldsymbol{\mu}}(\mathbf{z})^{\epsilon} w(\mathbf{z}) d\mathbf{z} \right| \leq .. \leq C' \|w\|_{W_0^{1,4}(\Omega)} \epsilon^{1/2} \int_{\mathbb{R}^2} |\mathbf{y}|^{1/2} G(\mathbf{y}) d\mathbf{y}$$

where G is the probability density of the standard normal distribution in $\mathbb{R}^2$. By passing at the supremum over w with $\|w\|_{W_0^{1,4}(\Omega)} = 1$ and $\boldsymbol{\mu} \in \Theta$ we get

$$\sup_{\boldsymbol{\mu} \in \Theta} \|f_{\boldsymbol{\mu}} - f_{\boldsymbol{\mu}}^{\epsilon}\|_{W^{-1,4}(\Omega)} \leq C'' \epsilon^{1/2}$$

for some constant $C'' > 0$. By classical stability estimates for elliptic PDEs, see e.g. Lemma 5.2, we then have $\sup_{\boldsymbol{\mu} \in \Theta} \|u_{\boldsymbol{\mu}} - u_{\boldsymbol{\mu}}^{\epsilon}\|_{W^{1,4/3}(\Omega)} \leq 10 C'' \epsilon^{1/2}$, as $\boldsymbol{\sigma}_{\boldsymbol{\mu}}(\mathbf{x}) \geq 10^{-1}$ for all $\boldsymbol{\mu} \in \Theta$. Up to the embedding the solution manifold in $L^2(\Omega)$, the claim now follows. $\square$

Proof of Claim 2. The idea is to re-parametrize the solution manifold, as the given parametrization suffers from the lack of injectivity. In fact, both $\mu_5 = 0$ and $\mu_5 = 2\pi$ return the same advective field (and we cannot exclude one extreme, or Θ would lose its compactness). To do so, let S^1 be the unit circle in $\mathbb{R}^2$. We define the hypercylinder $\Theta' := [0,1]^4 \times S^1 \times [0.1, 0.9]^2$. We will adopt a seven component notation as before, even though $\Theta' \subset \mathbb{R}^8$, as $\boldsymbol{\mu}_5 \in S^1$ is now 2-dimensional. We re-parametrize the coefficients of the PDE in terms of this new coordinates in the obvious way, especially for $\boldsymbol{\sigma}_{\boldsymbol{\mu}'}$ and $f_{\boldsymbol{\mu}'}$. For the advective field we let $\boldsymbol{b}_{\boldsymbol{\mu}'} := \boldsymbol{\mu}_5$. We shall now prove that: (i) the new parameter space satisfies $n_{\min}(\Theta') = 7$, (ii) the new parametric map $\boldsymbol{\mu}' \to u_{\boldsymbol{\mu}'}$ is continuous and (iii) injective.

Proof that $n_{\min}(\Theta') = 7$. Consider the map $\phi : \Theta' \to \mathbb{R}^7$ given by

$$\phi(\boldsymbol{\mu}') = (\boldsymbol{\mu}_5'(1 + \mu_1'),\ \mu_2',\ \mu_3',\ \mu_4',\ \mu_6',\ \mu_7').$$

Then the image $\phi(\Theta') = \{\mathbf{z} \in \mathbb{R}^2 : 1 \leq |\mathbf{z}| \leq 2\} \times [0,1]^3 \times [0.1, 0.9]^2 \subset \mathbb{R}^7$ has nonempty interior. In particular, $n_{\min}(\phi(\Theta')) = 7$. Since ϕ clearly admits a continuous inverse, $\phi^{-1} : \phi(\Theta') \to \Theta'$, we conclude that $n_{\min}(\Theta') = 7$. $\square$

Proof that the parametric map $\boldsymbol{\mu}' \to u_{\boldsymbol{\mu}'}$ *is continuous.* Clearly $\boldsymbol{\sigma}_{\boldsymbol{\mu}'}$ and $\boldsymbol{b}_{\boldsymbol{\mu}'}$ depend continuously on $\boldsymbol{\mu}'$. Using again the embedding $W_0^{1,4}(\Omega) \hookrightarrow \mathcal{C}^{0,1/2}(\Omega)$ as in the proof of Claim 1, it is also easy to see that the map $\boldsymbol{\mu}' \to f_{\boldsymbol{\mu}'}$ is $\Theta' \to W^{-1,4}(\Omega)$ Hölder continuous. By composition (see Lemma 5.3), we then obtain the continuity of the parametric map. $\square$

Proof that the parametric map $\boldsymbol{\mu}' \to u_{\boldsymbol{\mu}'}$ *is injective.* Let $\boldsymbol{\mu}', \boldsymbol{\mu}'' \in \Theta'$ and assume that $u \in W^{1,4/3}(\Omega)$ is a solution for both parameters, that is $u = u_{\boldsymbol{\mu}'} = u_{\boldsymbol{\mu}''}$. Classical results on inner regularity of solutions to elliptic PDEs ensure that $u_{\boldsymbol{\mu}'}$ is locally H^1 at all points except at the location of the Dirac delta $f_{\boldsymbol{\mu}'}$. The analogue holds for $u_{\boldsymbol{\mu}''}$, so clearly it must be $\mu_6' = \mu_6''$ and $\mu_7' = \mu_7''$ in order for the solutions to coincide. Next, let $w \in \mathcal{C}_0^{\infty}(\Omega_0)$ and extend it to zero on $\Omega \setminus \Omega_0$. Using w as test function for the equations of both $\boldsymbol{\mu}'$ and $\boldsymbol{\mu}''$ and then subtracting term by term yields

$$C \int_{\Omega_0} (\boldsymbol{b}_{\boldsymbol{\mu}'} - \boldsymbol{b}_{\boldsymbol{\mu}''}) \cdot \nabla u w = 0.$$

As w is arbitrary, it follows that ∇u is orthogonal to $(\boldsymbol{b}_{\boldsymbol{\mu}'} - \boldsymbol{b}_{\boldsymbol{\mu}''})$ on Ω_0. In particular, if $\boldsymbol{b}_{\boldsymbol{\mu}'} \neq \boldsymbol{b}_{\boldsymbol{\mu}''}$, then u must be constant along the direction $(\boldsymbol{b}_{\boldsymbol{\mu}'} - \boldsymbol{b}_{\boldsymbol{\mu}''})$ within Ω_0. But, because of the boundary conditions, this would make u identically constant near at least one edge of $\partial\Omega$. However, this is a contradiction. In fact, $u_{|\partial\Omega} \equiv 1$, thus classical maximum principles ensure that $u > 1$ a.e. in Ω (see e.g. Lemma 5.4). It follows that $\boldsymbol{b}_{\boldsymbol{\mu}'} = \boldsymbol{b}_{\boldsymbol{\mu}''}$ and so $\boldsymbol{\mu}_5' = \boldsymbol{\mu}_5''$. We now notice that, by subtracting the equations for $\boldsymbol{\mu}'$ and $\boldsymbol{\mu}''$, we have

$$\sum_{i=1}^{4} (\mu_i' - \mu_i'') \int_{\Omega_i} \nabla u \cdot \nabla w = 0 \quad \forall w \in \mathcal{C}_0^{\infty}(\Omega).$$

Fix any $i \in \{1,2,3,4\}$ and let $v \in \mathcal{C}^\infty(\Omega_i)$. Define $w \in \mathcal{C}^\infty(\Omega_i)$ to be any of the strong solutions to the PDE $-\Delta w = v$ with homogeneous Neumann boundary condition on $\partial\Omega_i$. Since the subdomains are clearly separated, it is possible to extend w on the whole domain Ω so that w is still smooth but also vanishes on $\partial\Omega$ and on Ω_j for all $j \neq i$. Using such w in the last identity above and integrating by parts yields

$$0 = (\mu_i' - \mu_i'')\int_{\Omega_i} \nabla u \cdot \nabla w = (\mu_i' - \mu_i'')\int_{\Omega_i} u(-\Delta w) = (\mu_i' - \mu_i'')\int_{\Omega_i} uv.$$

Now assume that $\mu_i' \neq \mu_i''$. Then $\int_{\Omega_i} uv = 0$ for all $v \in \mathcal{C}^\infty(\Omega_i) \implies u_{|\Omega_i} \equiv 0$, contradiction. Then $\mu_i' = \mu_i''$ and thus $\boldsymbol{\mu}' = \boldsymbol{\mu}''$, as claimed. $\square$

Thanks to the resultes above, Claim 2 now follows by Theorem 5.3. $\square$

Auxiliary results and proofs for Section 5.5

Lemma 5.1. *Let Ω be a Lipschitz domain. Let $\Lambda \subset \overline{\Omega}$ be the union of finitely many segments, where each segment intersects $\partial\Omega$ in at most two points (the extremes). For any fixed $v \in \mathcal{C}(\overline{\Omega})$ one has*

$$\frac{1}{\epsilon^2}\int_\Omega v(\mathbf{x})\max\{\epsilon - \operatorname{dist}(\mathbf{x},\Lambda), 0\}\,d\mathbf{x} \to \int_\Lambda v(\mathbf{s})d\mathbf{s} \quad \text{as } \epsilon \downarrow 0^+.$$

Proof. Let φ_Λ^ϵ be the (unscaled) kernel

$$\varphi_\Lambda^\epsilon(\mathbf{x}) := \max\{\epsilon - \operatorname{dist}(\mathbf{x},\Lambda), 0\}.$$

By definition, we note that φ_Λ^ϵ vanishes outside of the set $\Lambda + B(0,\epsilon) := \{\mathbf{x} + \epsilon\mathbf{v} \mid \mathbf{x} \in \Lambda,\ \mathbf{v} \in B(0,1)\}$. We now proceed in three steps.

Step 1. We start by proving that the lemma holds whenever Λ is composed by a single segment. Without loss of generality, we let $\Lambda = [0,1] \times \{0\}$. For the sake of simplicity, we further assume that $\Lambda \cap \partial\Omega = \varnothing$. The case in which Λ has an extreme on the boundary can be handled similarly by exploiting the Lipschitz regularity of $\partial\Omega$. Let $\epsilon < \operatorname{dist}(\Lambda, \partial\Omega)$, so that $\Lambda + B(0,\epsilon) \subset \Omega$. By direct computation we have

$$\int_\Omega v(\mathbf{x})\varphi_\Lambda^\epsilon(\mathbf{x})d\mathbf{x} = \frac{1}{\epsilon^2}\int_{\Lambda + B(0,\epsilon)} v(\mathbf{x})\varphi_\Lambda^\epsilon(\mathbf{x})d\mathbf{x} =$$

$$= \frac{1}{\epsilon^2}\int_{A_\epsilon \cup B_\epsilon} v(\mathbf{x})\varphi_\Lambda^\epsilon(\mathbf{x})d\mathbf{x} + \frac{1}{\epsilon^2}\int_{[0,1]\times[-\epsilon,\epsilon]} v(\mathbf{x})\varphi_\Lambda^\epsilon(\mathbf{x})d\mathbf{x}$$

where A_ϵ and B_ϵ are two half circles of radius ϵ respectively centered at the extremes of the segment Λ. It is easy to see that the first contribute vanishes as $\epsilon \downarrow 0^+$. In fact, since $||\varphi_\Lambda^\epsilon||_\infty = \epsilon$,

$$\left|\frac{1}{\epsilon^2}\int_{A_\epsilon \cup B_\epsilon} v(\mathbf{x})\varphi_\Lambda^\epsilon(\mathbf{x})d\mathbf{x}\right| \le \frac{1}{\epsilon^2}||v||_\infty \cdot \epsilon|A_\epsilon \cup B_\epsilon| = \pi\epsilon||v||_\infty.$$

Conversely, for the second term we have

$$\int_0^1 \frac{1}{\epsilon^2}\int_{-\epsilon}^{\epsilon} v(x_1,x_2)\varphi_\Lambda^\epsilon(x_1,x_2)dx_2dx_1 = \int_0^1 \frac{1}{\epsilon^2}\int_{-\epsilon}^{\epsilon} v(x_1,x_2)(\epsilon - |x_2|)dx_2dx_1 =$$

$$= \int_0^1 \frac{1}{\epsilon^2}\int_{-1}^{1} v(x_1,\epsilon z)(\epsilon - \epsilon|z|)\epsilon dz dx_1 = \int_0^1\int_{-1}^{1} v(x_1,\epsilon z)(1 - |z|)dzdx_1.$$

By letting $\epsilon \downarrow 0^+$ we then get

$$\int_0^1\int_{-1}^{1} v(x_1,0)(1-|z|)dzdx_1 = \left(\int_\Lambda v(\mathbf{s})d\mathbf{s}\right)\left(\int_{-1}^{1}(1-|z|)dz\right) = \int_\Lambda v(\mathbf{s})d\mathbf{s}.$$

Step 2. Let $\Lambda = L_1 \cup \cdots \cup L_n$ be given by the union of n segments. For any $i = 1, \ldots, n$, let $\hat{L}_i := \{\mathbf{x} \in \Omega \mid \text{dist}(\mathbf{x}, L_i) < \text{dist}(\mathbf{x}, \Lambda \setminus L_i)\}$. We prove the following auxiliary result,

$$|(\Omega \setminus \hat{L}_i) \cap (L_i + B(0,\epsilon))| = o(\epsilon^2).$$

To see this, we note that, upto sets of measure zero,

$$(\Omega \setminus \hat{L}_i) \cap (L_i + B(0,\epsilon)) = (\hat{L}_1 \cup \ldots \hat{L}_{i-1} \cup \hat{L}_{i+1} \cup \ldots \hat{L}_n) \cap (L_i + B(0,\epsilon)).$$

It is then sufficient to prove that $|\hat{L_j} \cap (L_i + B(0,\epsilon))| = o(\epsilon^2)$ for all j independently. If $L_i \cap L_j = \varnothing$, the proof is trivial. Conversely, if the two segments intersect, let θ be the angle between the two lines. It is easy to see that the intersection $\hat{L_j} \cap (L_i + B(0,\epsilon))$ is contained in a triangle of height ϵ and width $\epsilon/\tan(\theta/2) + \epsilon \tan(\theta/2)$. The conclusion follows.

Step 3. Let $\Lambda = L_1 \cup \cdots \cup L_n$ and define the regions $\hat{L}_1, \ldots \hat{L}_n$ as in the previous step. Fix any $v \in \mathcal{C}(\Omega)$. Then

$$\frac{1}{\epsilon^2}\int_{\Omega} v(\mathbf{x})\varphi_{\Lambda}^{\epsilon}(\mathbf{x})d\mathbf{x} = \frac{1}{\epsilon^2}\sum_{i=1}^{n}\int_{\hat{L}_i} v(\mathbf{x})\varphi_{\Lambda}^{\epsilon}(\mathbf{x})d\mathbf{x} = \sum_{i=1}^{n}\frac{1}{\epsilon^2}\int_{\hat{L}_i} v(\mathbf{x})\varphi_{L_i}^{\epsilon}(\mathbf{x})d\mathbf{x}.$$

Therefore, we can prove the original claim by showing that $\frac{1}{\epsilon^2}\int_{\hat{L}_i} v(\mathbf{x})\varphi_{L_i}^{\epsilon}(\mathbf{x})d\mathbf{x} \to \int_{L_i} v(\mathbf{s})d\mathbf{s}$ for each i. At this purpose, fix any $i = 1, \ldots, n$. We have

$$\left|\frac{1}{\epsilon^2}\int_{\Omega} v(\mathbf{x})\varphi_{L_i}^{\epsilon}(\mathbf{x})d\mathbf{x} - \frac{1}{\epsilon^2}\int_{\hat{L}_i} v(\mathbf{x})\varphi_{L_i}^{\epsilon}(\mathbf{x})d\mathbf{x}\right| \le \frac{1}{\epsilon^2}\int_{\Omega \setminus \hat{L}_i} |v(\mathbf{x})||\varphi_{L_i}^{\epsilon}(\mathbf{x})|d\mathbf{x} \le$$

$$\le \frac{1}{\epsilon^2}\|v\|_{\infty} \cdot \epsilon|(\Omega \setminus \hat{L}_i) \cap (L_i + B(0,\epsilon))| = o(\epsilon)$$

and the conclusion follows from Step 1. □

Auxiliary results on Partial Differential Equations

Lemma 5.2. *Let $(V, \|\cdot\|_V)$ and $(W, \|\cdot\|_W)$ be two Banach spaces, with W reflexive. Let $(W^*, \|\cdot\|_*)$ be the dual space of W and define $(\mathfrak{B}(V,W), |||\cdot|||)$ as the normed space of bounded bilinear forms $V \times W \to \mathbb{R}$, where*

$$|||a||| := \sup_{\substack{\|v\|_V = 1 \\ \|w\|_W = 1}} |a(v,w)|.$$

Let $\mathfrak{B}_c(V,W) \subset \mathfrak{B}(V,W)$ be the subset of coercive bounded bilinear forms, i.e. $a \in \mathfrak{B}(V,W)$ for which

$$\lambda(a) := \inf_{\|v\|_V = 1}\sup_{\|w\|_W = 1} |a(v,w)| > 0, \qquad \text{and} \qquad \inf_{\|w\|_W = 1}\sup_{\|v\|_V = 1} |a(v,w)| > 0.$$

Then,

i) λ is $\mathfrak{B}(V,W) \to \mathbb{R}$ continuous

ii) For each $a \in \mathfrak{B}_c(V,W)$ and $F \in W^*$ there exists a unique $u = u_{a,F} \in V$ such that $a(v,w) = F(w)$ for all $w \in W$. Furthermore, u depends continuously on both a and F. In particular:

$$\|u_{a,F} - u_{a',F'}\|_V \le \frac{1}{\lambda(a)}\left(\|F - F'\|_* + \frac{1}{\lambda(a')}|||a - a'||| \cdot \|F'\|_*\right) \tag{5.12}$$

for all $a, a' \in \mathfrak{B}_c(V,W)$ and $F, F' \in W^*$.

Proof. i) Let $a, a' \in \mathfrak{B}(V, W)$. For every $v \in V$ and $w \in W$ with $\|v\|_V = \|w\|_W = 1$ we have

$$a(v,w) = a'(v,w) + (a - a')(v,w) \leq |a'(v,w)| + |||a - a'|||.$$

Since the above holds for both w and $-w$, we actually have $|a(v,w)| \leq |a'(v,w)| + |||a - a'|||$, and thus $\lambda(a) \leq \lambda(a') + |||a - a'|||$. As the situation is symmetric in a and a', it follows that $|\lambda(a) - \lambda(a')| \leq |||a - a'|||$. In particular, λ is Lipschitz-continuous.

ii) Given $a \in \mathfrak{B}_c(V, W)$ and $F \in W^*$, the existence and the uniqueness of $u_{a,F}$ follow from a Banach space version of the Lions-Lax-Milgram theorem (see Lemma 3.1. in [88]). Furthermore, one also has the stability estimate $\|u_{a,F}\|_V \leq (\lambda(a))^{-1}\|F\|_*$.

To get the inequality in (5.12), let $a, a' \in \mathfrak{B}(V, W)$, $F, F' \in W^*$ and $u := u_{a,F}$, $u' := u_{a',F'}$. Then $a(u,w) = F(w)$ and $a'(u',w) = F'(w)$ for all $w \in V$. We subtract these two identities to get

$$\begin{aligned} &a(u,w) - a'(u',w) = F(w) - F'(w) \\ &\implies a(u - u', w) = (F - F')(w) + (a' - a)(u', w). \end{aligned}$$

It follows that, for all $w \in W$, one has $a(u - u', w) \leq \|F - F'\|_* \|w\|_W + |||a' - a||| \cdot \|u'\|_V \|w\|_W$. By linearity, using both w and $-w$, we conclude that

$$|a(u - u', w)| \leq \|F - F'\|_* \|w\|_W + |||a - a'||| \cdot \|u'\|_V \|w\|_W \qquad \forall w \in W.$$

In particular, passing at the supremum over $\|w\|_W = 1$ yields

$$\lambda(a)\|u - u'\|_V \leq \|F - F'\|_* + |||a - a'||| \cdot \|u'\|_V.$$

Now, we may apply the stability estimate for $\|u'\|_V$ and divide by $\lambda(a)$ to get (5.12). Finally the latter, together with (i), shows that $u' \to u$ as soon as $a' \to a$ and $F' \to F$. □

For the next Lemma, we consider the notation introduced in Section 5.2.2.1.

Lemma 5.3. *Let $\Omega \subset \mathbb{R}^d$ be a bounded domain. Let $1 < q < +\infty$ and define the conjugate exponent $q' := (q-1)^{-1}q$. For each $\sigma \in \Sigma(\Omega)$, $b \in B(\Omega)$, $f \in W^{-1,q'}(\Omega)$ and $g \in W^{1/q',q}(\partial\Omega)$ let $u = u_{\sigma,b,f,g}$ be the unique solution to the following variational problem*

$$\begin{aligned} &u \in W^{1,q}(\Omega): \\ &u_{|\partial\Omega} = g \quad \text{and} \quad \int_\Omega \boldsymbol{\sigma}\nabla u \cdot \nabla w + \int_\Omega (\boldsymbol{b} \cdot \nabla u)\, w = \int_\Omega f w \quad \forall w \in W_0^{1,q'}(\Omega). \end{aligned}$$

Then, the solution map $(\boldsymbol{\sigma}, \boldsymbol{b}, f, g) \to u_{\boldsymbol{\sigma},\boldsymbol{b},f,g}$ is: (i) continuous, (ii) Lipschitz continuous on all compact subsets.

Proof. Before moving the actual proof, we shall recall that there exists a bounded linear operator $T : W^{1/q',q}(\partial\Omega) \to W^{1,q}(\Omega)$ for which $Tg_{|\partial\Omega} = g$, namely a right-inverse of the trace operator (see [146]). In particular, there exists a constant $\tilde{C} > 0$ such that $||Tg||_{W^{1,q}(\Omega)} \leq \tilde{C}||g||_{W^{1/q',q}(\partial\Omega)}$.

For the sake of brevity, we let $V := W_0^{1,q}(\Omega)$, $W := W_0^{1,q'}$ and define W^* as the dual space of $W_0^{1,q'}$ endowed with the operator norm. As in Lemma 5.2, we also let $\mathfrak{B}(V, W)$ be the collection of all bounded bilinear maps $V \times W \to \mathbb{R}$ equipped with the corresponding operator norm. Similarly, we define $\mathfrak{B}_c(V, W)$ to be the subset of coercive bounded bilinear maps. We introduce the following operators:

$$\begin{aligned} \mathcal{A}:\ & L^\infty(\Omega, \mathbb{R}^{d\times d}) \times L^\infty(\Omega, \mathbb{R}^d) \longrightarrow \mathfrak{B}(V, W) \\ & (\boldsymbol{\sigma}, \boldsymbol{b}) \longrightarrow a_{\boldsymbol{\sigma},\boldsymbol{b}} := \mathcal{A}(\boldsymbol{\sigma}, \boldsymbol{b}), \end{aligned}$$

$$\begin{aligned} \mathcal{F}:\ & L^\infty(\Omega, \mathbb{R}^{d\times d}) \times L^\infty(\Omega, \mathbb{R}^d) \times W^{-1,q'}(\Omega) \times W^{1/q',q}(\partial\Omega) \longrightarrow W^* \\ & (\boldsymbol{\sigma}, \boldsymbol{b}, f, g) \longrightarrow F_{\boldsymbol{\sigma},\boldsymbol{b},f,g} := \mathcal{F}(\boldsymbol{\sigma}, \boldsymbol{b}, f, g), \end{aligned}$$

where,

$$a_{\boldsymbol{\sigma},\boldsymbol{b}}(v,w) := \int_\Omega \boldsymbol{\sigma}\nabla v\cdot\nabla w + \int_\Omega (\boldsymbol{b}\cdot\nabla v)\,w,$$
$$F_{\boldsymbol{\sigma},\boldsymbol{b},f,g}(w) := \int_\Omega \boldsymbol{\sigma}\nabla Tg\cdot\nabla w + \int_\Omega (\boldsymbol{b}\cdot Tg)w + \int_\Omega fw.$$

We claim that:

1) The operator $\mathcal{A}$ is linear and continuous. Also, $a_{\boldsymbol{\sigma},\boldsymbol{b}} \in \mathfrak{B}_c(V,W)$ for all choices of $\boldsymbol{\sigma}\in\Sigma(\Omega)$ and $\boldsymbol{b}\in B(\Omega)$.

2) The operator $\mathcal{F}$ continuous. Also, it is Lipschitz continuous when restricted to any compact subset of its domain.

We shall now prove these claims. First of all, let $C>0$ be the Poincàre constant for the domain Ω and the exponent q'. Then, it is straightforward to see that

$$\begin{aligned}|a_{\boldsymbol{\sigma},\boldsymbol{b}}(v,w)| \le{}& \|\boldsymbol{\sigma}\|_{L^\infty(\Omega,\mathbb{R}^{d\times d})}\|v\|_{W_0^{1,q}(\Omega)}\|w\|_{W_0^{1,q'}(\Omega)}\\ &+ C\|\boldsymbol{b}\|_{L^\infty(\Omega,\mathbb{R}^d)}\|v\|_{W_0^{1,q}(\Omega)}\|w\|_{W_0^{1,q'}(\Omega)},\end{aligned}$$

for all $v\in V$ and $w\in W$. In particular, $\mathcal{A}$ is both linear and bounded, thus continuous. Let now $\boldsymbol{\sigma}\in\Sigma(\Omega)$, $\boldsymbol{b}\in B(\Omega)$ and define $\varepsilon=\varepsilon(\boldsymbol{\sigma})>0$ to be the ellipticity constant of $\boldsymbol{\sigma}$. We notice that if $\varphi\in\mathcal{C}_0^\infty(\Omega)$, then φ is both an element of V and W. Also, integrating by parts yields

$$\begin{aligned}a_{\boldsymbol{\sigma},\boldsymbol{b}}(\varphi,\varphi) &= \int_\Omega \boldsymbol{\sigma}\nabla\varphi\cdot\nabla\varphi + \int_\Omega \boldsymbol{b}\cdot(\varphi\nabla\varphi) \ge\\ &\ge \varepsilon\|\varphi\|_{W_0^{1,q}(\Omega)}\|\varphi\|_{W_0^{1,q'}(\Omega)} + \int_\Omega \boldsymbol{b}\cdot\nabla\left(\frac{1}{2}\varphi^2\right) =\\ &= \varepsilon\|\varphi\|_{W_0^{1,q}(\Omega)}\|\varphi\|_{W_0^{1,q'}(\Omega)} - \frac{1}{2}\int_\Omega \operatorname{div}(\boldsymbol{b})\varphi^2 =\\ &= \varepsilon\|\varphi\|_{W_0^{1,q}(\Omega)}\|\varphi\|_{W_0^{1,q'}(\Omega)},\end{aligned}$$

as $\boldsymbol{b}$ is divergence free. It follows that for each $\varphi\in\mathcal{C}_0^\infty(\Omega)$ with $\varphi\neq 0$

$$\sup_{\substack{\psi\in\mathcal{C}_0^\infty(\Omega)\\ \|\psi\|_{W_0^{1,q'}(\Omega)}=1}} |a_{\boldsymbol{\sigma},\boldsymbol{b}}(\varphi,\psi)| \ge a_{\boldsymbol{\sigma},\boldsymbol{b}}\left(\varphi, \|\varphi\|^{-1}_{W_0^{1,q'}(\Omega)}\varphi\right) \ge \varepsilon\|\varphi\|_{W_0^{1,q}(\Omega)}$$

and, similarly,

$$\sup_{\substack{\psi\in\mathcal{C}_0^\infty(\Omega)\\ \|\psi\|_{W_0^{1,q}(\Omega)}=1}} |a_{\boldsymbol{\sigma},\boldsymbol{b}}(\psi,\varphi)| \ge a_{\boldsymbol{\sigma},\boldsymbol{b}}\left(\|\varphi\|^{-1}_{W_0^{1,q}(\Omega)}\varphi, \varphi\right) \ge \varepsilon\|\varphi\|_{W_0^{1,q'}(\Omega)}.$$

Since $a_{\boldsymbol{\sigma},\boldsymbol{b}}$ is continuous and $\mathcal{C}_0^\infty(\Omega)$ is both dense in V and W, by the above we conclude that $a_{\boldsymbol{\sigma},\boldsymbol{b}}\in\mathfrak{B}_c(V,W)$. This proves claim (1).

We now move to (2). For each $\boldsymbol{\sigma},\boldsymbol{b},f,g$ and $w\in W$ we have

$$\begin{aligned}|F_{\boldsymbol{\sigma},\boldsymbol{b},f,g}(w)| \le{}& \|\boldsymbol{\sigma}\|_{L^\infty(\Omega,\mathbb{R}^{d\times d})}\|Tg\|_{W^{1,q}(\Omega)}\|w\|_{W_0^{1,q'}(\Omega)}\\ &+ C\|\boldsymbol{b}\|_{L^\infty(\Omega,\mathbb{R}^d)}\|Tg\|_{W^{1,q}(\Omega)}\|w\|_{W_0^{1,q'}(\Omega)}\\ &+ \|f\|_{W^{-1,q'}(\Omega)}\|w\|_{W_0^{1,q'}(\Omega)}.\end{aligned}$$

In particular, for all $w\in W$ with unitary norm,

$$|F_{\boldsymbol{\sigma},\boldsymbol{b},f,g}(w)| \le \tilde{C}\|g\|_{W^{1/q',q}(\partial\Omega)}\left(\|\boldsymbol{\sigma}\|_{L^\infty(\Omega,\mathbb{R}^{d\times d})} + C\|\boldsymbol{b}\|_{L^\infty(\Omega,\mathbb{R}^d)}\right) + \|f\|_{W^{-1,q'}(\Omega)}.$$

From here, arguing by linearity easily yields (2).

Finally, for each $\boldsymbol{\sigma} \in \Sigma(\Omega), \boldsymbol{b} \in B(\Omega), f \in W^{-1,q'}(\Omega), g \in W^{1/q',q}(\partial\Omega)$ let $\tilde{u}_{\boldsymbol{\sigma},\boldsymbol{b},f,g} \in V = W_0^{1,q}(\Omega)$ be the unique solution to the variational problem

$$a_{\boldsymbol{\sigma},\boldsymbol{b},f,g}(\tilde{u}, w) = F_{\boldsymbol{\sigma},\boldsymbol{b},f,g}(w) \quad \forall w \in W.$$

At this regard, we notice that $W = W_0^{1,q'}(\Omega)$ is reflexive, in fact $1 < q < +\infty$ implies $1 < q' < +\infty$. Therefore, by Lemma 5.2, we know that $\tilde{u}_{\boldsymbol{\sigma},\boldsymbol{b},f,g}$ exists unique and it depends continuously (by composition) on $(\boldsymbol{\sigma}, \boldsymbol{b}, f, g)$. Furthermore, as clear from inequality (5.12) in Lemma 5.2, the correspondence $(\boldsymbol{\sigma}, \boldsymbol{b}, f, g) \to \tilde{u}_{\boldsymbol{\sigma},\boldsymbol{b},f,g}$ is Lipschitz continuous on every compact subset of the product space $\Sigma(\Omega) \times B(\Omega) \times W^{-1,q'}(\Omega) \times W^{1/q',q}(\partial\Omega)$. This is easily deduced by the properties of $\mathcal{A}$ and $\mathcal{F}$ as well as by the fact that compactness is preserved under continuous transformations. Finally, we notice that

$$u_{\boldsymbol{\sigma},\boldsymbol{b},f,g} = \tilde{u}_{\boldsymbol{\sigma},\boldsymbol{b},f,g} + Tg.$$

The conclusion follows. □

Lemma 5.4. *Consider the context and notation in Lemma 5.3. If $g \equiv c \in \mathbb{R}$ and $f > 0$ in the distributional sense, then $u > c$ a.e. in Ω.*

Proof. This simply derives from maximum principles. We first prove the case $c = 0$. Let $\eta \in \mathcal{C}_0^\infty(\Omega)$ be such that $\eta > 0$ everywhere in Ω. Let $w \in H_0^1(\Omega)$ be the solution to the following adjoint variational problem:

$$\int_\Omega \boldsymbol{\sigma}^T \nabla w \cdot \nabla v - \int_\Omega (\boldsymbol{b} \cdot \nabla w) v = \int_\Omega \eta v \quad \forall v \in \mathcal{C}_0^\infty(\Omega).$$

Within this regular case, the classical maximum principle states $w > \max w_{|\Omega} = 0$ in Ω, see e.g. Theorem 2 in [42]. Now we notice that $w \in W_0^{1,q'}(\Omega)$, as the PDE also admits a unique solution in that space. Thus, by density, we are allowed to consider u as test function for w and viceversa. Doing so and subtracting the equations for u and w yields

$$\int_\Omega \eta u = \int_\Omega f w,$$

since $\boldsymbol{\sigma}\nabla u \cdot \nabla w = \boldsymbol{\sigma}^T \nabla w \cdot \nabla u$ and the advective terms cancel out using the integration by parts formula (recall that $\boldsymbol{b}$ is divergence free while both u and w vanish on $\partial\Omega$). The above shows that $\int_\Omega \eta u > 0$, as the right hand side is positive by hypothesis. As η was arbitrary, we conclude that $u > 0$ a.e. in Ω. Let now $c \neq 0$. It is elementary to see that $u = c + u_0$, where u_0 solves the variational problem with homogenous boundary conditions. The conclusion follows. □

6 | Convolutional Neural Networks and their role in operator learning

After autoencoders, one of the main ingredients of the DL-ROM approach presented in Chapter 5, are Convolutional Neural Networks (CNNs). For this reason, we devote this sixth Chapter to the study of these architectures. More precisely, we focus on the approximation capabilities of CNNs in the context of Operator Learning, emphasizing their ability in the reconstruction of high dimensional outputs, a situation that is of particular interest for applications involving parametrized PDEs. Surprisingly, despite the rapidly evolving literature of constructive approximation applied to Deep Learning, at the best of our knowledge, there are no mathematical results about CNNs towards this direction. Our purpose for this Chapter is thus to extend the state-of-the-art by deriving novel approximation bounds for CNN models. To do so, we shall exploit an intimate connection between convolutions and the Fourier transform. This research was carried out during my last year of PhD, in collaboration with Dr. Fresca S., and it was recently published in Franco et al., "Approximation bounds for convolutional neural networks in operator learning", Neural Networks [66].

If the dimensions into play become very large, the total number of weights and biases in a DNN model can quickly become computationally intractable. In fact, given input-output dimensions $m, n \geq 1$, and an activation function ρ, the collection of all possible layers

$$\text{Dense}(m, n, \rho) := \{\mathbf{v} \to \rho(\mathbf{W}\mathbf{v} + \mathbf{b}) \mid \mathbf{W} \in \mathbb{R}^{n\times m},\ \mathbf{b} \in \mathbb{R}^n\}$$

is a set parametrized by $mn + n$ parameters. Therefore, even after having fixed the architecture, it can be very hard to identify the correct layer that suits our needs.

For this reason, whenever dealing with high-dimensional regimes, data scientists avoid the use of layers that are as general as the ones in Definition 0.1. Instead, they restrict their attention to subclasses $\mathcal{S} \subseteq \text{Dense}(m, n, \rho)$ that are usually obtained by imposing constraints on the weight matrix $\mathbf{W}$. One main example of this fact are the so-called *convolutional layers*: there, the weight matrix is required to have a suitable structure so that the linear map $\mathbf{v} \to \mathbf{W}\mathbf{v}$ can be reproduced by a cross-correlation operator. In practice, this forces sparsity patterns in the weight matrix and ensures that multiple entries have unique common values (see Figure 6.1). While it is tempting to provide a rigorous mathematical description of these objects, the notation can quickly get extremely heavy. For this reason, it might be worth to start with some heuristics before getting to their formal definition.

The idea of convolutional layers was first introduced to handle image-like data, which are typically stored in 3D tensors, $\mathbb{V} \in \mathbb{R}^{c\times m_2\times m_3}$: one dimension to specify the RGB channel (red, green or blue), and the remaining two to span the pixel values. In that case, researchers proposed the use of convolutional layers to account for spatial dependence: image-like inputs are not just generic vectors in $\mathbb{R}^{cm_2m_3} \cong \mathbb{R}^{c\times m_2\times m_3}$, instead they enjoy specific properties that we can exploit in order to define lighter architectures, see Figure 6.1. While this is

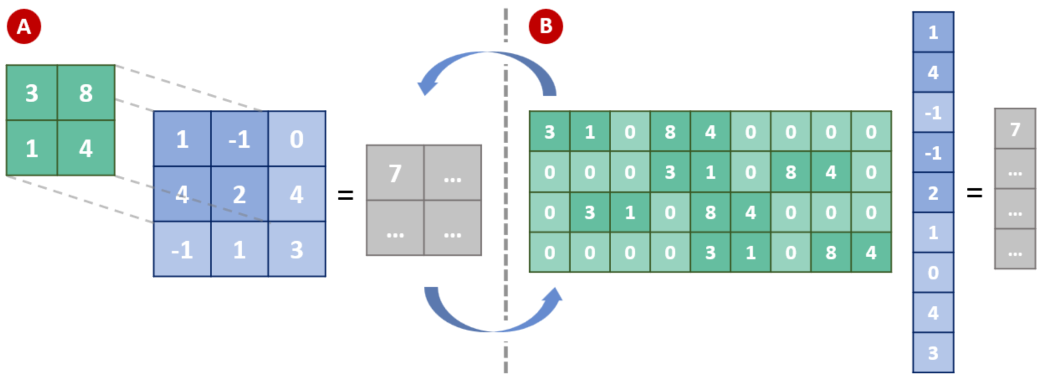


Figure 6.1: Visualization of a 2D convolutional layer (linear part). Panel (A) shows the action of the layer as a cross-correlation operator, while panel (B) uses a matrix-vector representation, highlighting how CNNs are actually particular DNNs.

for 2D convolutional layers, the idea easily generalizes to any spatial dimension. In general, a dD convolutional layer assumes the input to have an underlying d-dimensional structure and thus only accepts inputs of the form

$$\mathbb{R}^{cm_1\cdot\dots\cdot m_d} \cong \mathbb{R}^{c\times m_1\times\dots\times m_d}.$$

The process of moving back and forth from the vectorized representation, $\mathbb{R}^{cm_1\cdot\dots\cdot m_d}$, to the tensor-like one, $\mathbb{R}^{c\times m_1\times\dots\times m_d}$ is called *reshaping*. The advantage of this operation is that it allows us to compute the action of a convolutional layer without actually assembling its weight matrix $\mathbf{W}$.

In order to fix the ideas, let us restrict to the case $d = 1$, where we can more easily report the formal definition of convolutional layers in mathematical terms. To this end, we shall use the following notation to describe tensor objects. Given $\mathbf{A} \in \mathbb{R}^{n_1\times\dots\times n_d}$, we write $\mathbf{A}_{i_1,\dots,i_p}$ for the subtensor in $\mathbb{R}^{n_{p+1}\times\dots\times n_d}$ obtained by fixing the first p dimensions along the specified axis, where $1 \le i_j \le n_j$. We can now define in a formal way the concept of a 1D convolutional layer and its transpose counterpart. These definitions are given along the lines of the existing literature, and correspond to those adopted by the Python library `Pytorch`.

Definition 6.1. *Let $m, m's, t, d$ be positive integers and let g be a common divisor of m and m'. A 1D Convolutional layer with m input channels, m' output channels, grouping number g, kernel size s, stride t, dilation factor d and activation function $\rho : \mathbb{R} \to \mathbb{R}$, is a map of the form*

$$\Phi : \mathbb{R}^{m\times n} \to \mathbb{R}^{m'\times\left\lfloor\frac{n-d(s-1)-1}{t}+1\right\rfloor}$$

whose action on a given input $\mathbf{X} \in \mathbb{R}^{m\times n}$ is defined as

$$\Phi(\mathbf{X})_{k'} = \rho\left(\sum_k \mathbf{W}_{k',k} \otimes_{t,d} \mathbf{X}_k + \mathbf{B}_{k'}\right),$$

where $1 \le k' \le m'$, while the sum index k runs as below,

$$k = \lfloor g(k'-1)/m \rfloor m/g + 1, \dots, (\lfloor g(k'-1)/m \rfloor + 1)\, m/g.$$

Here, $\mathbf{W} \in \mathbb{R}^{m'\times(m/g)\times s}$ and $\mathbf{B} \in \mathbb{R}^{m'\times\left\lfloor\frac{n-d(s-1)-1}{t}+1\right\rfloor}$ are the weight tensor and the bias term, respectively. The symbol $\otimes_{t,d}$ denotes the cross-correlation operator with stride t and dilation d. That is, for any $\mathbf{w} \in \mathbb{R}^s$ and $\mathbf{x} \in \mathbb{R}^n$ one has $\mathbf{w} \otimes_{t,d} \mathbf{x} \in \mathbb{R}^{\left\lfloor\frac{n-d(s-1)-1}{t}+1\right\rfloor}$, where

$$(\mathbf{w} \otimes_{t,d} \mathbf{x})_j := \sum_{i=1}^{s} w_i x_{(j-1)t+(i-1)d+1}.$$

The default values for stride and dilation are $t = 1$, $d = 1$. For this reason, with little abuse of notation, one says that Φ has no stride and no dilation to intend that $t = 1$, $d = 1$. Similarly, we assume $g = 1$ whenever the grouping number is not declared explicitly.

Definition 6.2. *Let $m, m's, t, d$ be positive integers and let g be a common divisor of m and m'. A 1D Transposed Convolutional layer with m input channels, m' output channels, grouping number g, kernel size s, stride t, dilation factor d and activation function $\rho : \mathbb{R} \to \mathbb{R}$, is a map of the form*

$$\Phi : \mathbb{R}^{m \times n} \to \mathbb{R}^{m' \times (n-1)t + d(s-1)+1}$$

whose action on a given input $\mathbf{X} \in \mathbb{R}^{m \times n}$ is defined as

$$\Phi(\mathbf{X})_{k'} = \rho\left(\sum_k \mathbf{W}_{k,k'} \circledast^{\top}_{t,d} \mathbf{X}_k + \mathbf{B}_{k'}\right),$$

where $1 \le k' \le m'$, while the sum index k runs as below,

$$k = \lfloor g(k'-1)/m \rfloor m/g + 1, \dots, (\lfloor g(k'-1)/m \rfloor + 1)\, m/g.$$

Here, $\mathbf{W} \in \mathbb{R}^{(m/g) \times m' \times s}$ and $\mathbf{B} \in \mathbb{R}^{m' \times (n-1)t - d(s-1)+1}$ are the weight tensor and the bias term, respectively. The symbol $\circledast^{\top}_{t,d}$ denotes the transposed cross-correlation operator with stride t and dilation d. That is, for any $\mathbf{w} \in \mathbb{R}^s$ and $\mathbf{x} \in \mathbb{R}^n$ one has $\mathbf{w} \circledast^{\top}_{t,d} \mathbf{x} \in \mathbb{R}^{(n-1)t - d(s-1)+1}$, where

$$\left(\mathbf{w} \circledast^{\top}_{t,d} \mathbf{x}\right)_j := \sum_i w_{\left\lfloor \frac{(i-1)t+1-j}{d} \right\rfloor + 1} x_i,$$

the sum index i running as below,

$$i = \left\lfloor \frac{j-1}{t} + 1 \right\rfloor, \dots, \left\lfloor \frac{(s-1)d + j - 1}{t} + 1 \right\rfloor.$$

Definition 6.3. *A Convolutional Neural Network (CNN) is any map that, up to reshaping operations, can be written as the composition of (transposed) convolutional layers. In particular, as (transposed) convolutional layers can be seen as a special case of dense layers, every CNN is a DNN.*

As we mentioned, the power of CNNs lies in that they can tackle high dimensional objects without requiring too many degrees of freedom. This makes them suited for handling image-like data, but also more complex objects such as discretized functions. In particular, as we discussed in Chapter 5, they can be a natural choice when dealing with parameter dependent PDEs or, more in general, with problems of *operator learning*. Indeed, CNNs have reported remarkable successes in these fields, and our experience with these architectures was extremely positive (cf. Chapter 5, Section 5.4). However, the current literature still lacks: (i) a clear understanding of the role played by CNN hyperparameters, (ii) rigorous error bounds such as the ones by Yarotski in Theorem 0.3. Our purpose for this Chapter is to remedy this shortcoming. To this end, we shall directly state our main result in the next Section, after a brief contextualization. However, as the proof is quite technical and requires the derivation of several preliminary results, we postpone its demonstration to Section 6.4. Finally, even though mathematical proofs already speak for themselves, we have decided to include some additional numerical experiments for the empirical validation of our error bounds. We report them in Section 6.2. In general, we mention that all the analyses presented in this Chapter refer a work that we recently published, see [66].

6.1 Understanding CNNs: a new interpretable approximation bound

Convolutional neural networks have become very popular after their tremendous success in computer vision, with applications ranging from image processing to generative models for images generation [55, 188]. From a mathematical point of view, image-like data are equivalent to discrete functional signals defined over rectangular domains and vice versa. Indeed, each continuous function $u : [0,1]^2 \to \mathbb{R}$ can be discretized in matrix form as

$$\mathbf{U} := \begin{bmatrix} u(x_{1,1}) & \dots & u(x_{1,N}) \\ \dots & \dots & \dots \\ u(x_{N,1}) & \dots & u(x_{N,N}) \end{bmatrix} \in \mathbb{R}^{N\times N},$$

where $\{x_{i,j}\}_{i,j=1,\dots,N}$ are the vertices of some uniform grid defined over the unit square. In light of this, CNNs have been recently employed for tasks that go beyond computer vision, such as operator learning and reduced order modelling of parameter-dependent PDEs [65, 70, 71, 118, 143].

As an example, let $\Omega = (0,1)^d$ and assume we are given an operator $\boldsymbol{\mu} \to u_{\boldsymbol{\mu}}$ that maps a finite dimensional input $\boldsymbol{\mu} \in \mathbb{R}^p$ onto some functional signal $u_{\boldsymbol{\mu}} : \Omega \to \mathbb{R}$. As we already discussed in Chapter 5, this is a classical set-up in parameter-dependent PDE models, where each parameter instance is associated with the corresponding PDE solution. Once a suitable, discrete grid of points $\{x_{i_1,\dots,i_d}\}_{i_1,\dots,i_d=1,\dots,N} \subset \Omega$ has been introduced, the operator of interest can be expressed as

$$\mathbb{R}^p \ni \boldsymbol{\mu} \to \mathbf{U}_{\boldsymbol{\mu}} \in \mathbb{R}^{N\times\dots\times N},$$

denoting by $\mathbf{U}_{\boldsymbol{\mu}}^{i_1,\dots,i_d} \approx u_{\boldsymbol{\mu}}(x_{i_1,\dots,i_d})$. The final goal is then to construct a neural network model

$$\Phi : \mathbb{R}^p \to \mathbb{R}^{N\times\dots\times N}, \text{ such that } \Phi(\boldsymbol{\mu}) \approx \mathbf{U}_{\boldsymbol{\mu}},$$

with the idea of replacing an operator that is otherwise computationally expensive to evaluate. As previously mentioned, this task can be successfully achieved by CNNs, as they allow to intrinsically account for underlying spatial correlations. However, the literature still lacks a comprehensive mathematical analysis and foundation motivating the remarkable performance shown by CNNs, and the role played by each hyperparameter in a CNN model remains unclear. In this work, we aim at addressing these critical points, showing rigorous estimates on the error $\mathcal{E} := \sup_{\boldsymbol{\mu}} \sup_{\mathbf{j}\in\{1,\dots,N\}^d} |u_{\boldsymbol{\mu}}(x_{\mathbf{j}}) - \Phi_{\mathbf{j}}(\boldsymbol{\mu})|$ generated when approximating the operator of interest by means of CNNs.

6.1.1 Review of the state-of-the-art

As we discussed at the beginning of the Thesis, neural networks were known to be universal approximators since Cybenko in 1989 [50]. However, the design of effective NN architectures able to preserve desired accuracy properties had not been in-depth investigated until recent years. A substantial improvement was achieved by Yarotsky in 2017, [202], where a rigorous mathematical meaning to structural DNN properties such as width and depth has been first provided. In particular, Yarotsky proved that any s-differentiable scalar-valued map $f : [0,1]^p \to \mathbb{R}$ can be approximated uniformly with error $0 < \varepsilon < 1/2$ by some ReLU DNN with $c\log(1/\varepsilon)$ layers and $c\varepsilon^{-p/s}\log(1/\varepsilon)$ active weights, where $c = c(p,s,f)$ is some constant that depends on the derivatives of f (see Theorem 0.3 in the introductory Chapter). This result was later extended to more general activation functions and different norms, see e.g. [78, 79, 182], and adapted to the case of CNNs exploiting some algebraic arguments that link dense and convolutional layers, see e.g. [206, 82].

However, all these results are limited to the approximation of scalar-valued maps and they are not suited for operator learning. To this end, it is worth to note the following aspect. Assume that we are interested in approximating a vector-valued map $f : [0,1]^p \to \mathbb{R}^n$,

$f(\boldsymbol{\mu}) = [f_1(\boldsymbol{\mu}), \dots, f_n(\boldsymbol{\mu})]$, with a DNN model Φ. Clearly, we could exploit the aforementioned results to approximate each f_i with some DNN ϕ_i, and then stack together the models to get $\Phi := [\phi_1, \dots, \phi_n]$. However, with this construction the number of active weights in Φ would grow linearly with n as $n \to +\infty$. In our context, where we deal with functional outputs and $n = N^d$ comes from having discretized $\Omega = (0,1)^d$ with a computational grid with N nodes per side, this would be rather undesirable.

Nevertheless, new approaches are now appearing in the literature, in a first attempt to employ DNNs for operator learning. Some of these, such as Neural Operators [109] and DeepONets [126], work with a continuous functional output space, while a second class of approaches relies on a discretization of the output space, see e.g. [112]. Here, we focus on the latter family of approaches.

Neural Operators provide a novel framework for building models between infinite dimensional spaces, and are essentially based on integral operators. Among them, those that have been mostly investigated are Fourier Neural Operators, for which several error estimates have been derived, see, e.g., [108]. Conversely, DeepONets are a class of models based on a separation of variables approach, which decouples the input parameter and the space variable at output. Error estimates for DeepONets are also available, see [115], and they are mostly settled on the aforementioned results in the scalar case and on those by [176] for high-dimensional inputs.

Besides these methods, Deep Learning approaches that discretize the functional output space are also available. This need usually arises, for instance, when dealing with parameter-dependent PDEs, whose solutions are usually computed through numerical discretization schemes like, e.g., the finite element method. In this case, the functional output space – usually given by a Sobolev space – is replaced by a finite-dimensional trial space (e.g., the space of polynomial finite elements of degree r built over a triangulation of the spatial domain Ω). Aside from our proposal in Chapter 5, Deep Learning approaches of this type were also proposed in [24] and [112]. The former relies on linear reduction methods to deal with the functional component at output, and it is able to recover mesh independence. The second one, instead, is purely based on DNNs. Both works provide error estimates, most of which are derived by exploiting the results in the scalar-case and projection arguments.

This flourishing literature indicates a growing interest at understanding the properties of DNN models and their potential in operator learning. However, at the best of our knowledge, no comprehensive study has yet been proposed for CNN models, despite these being extremely popular in practical applications. One reason might be that CNN architectures can be traced back to sparse versions of dense models, which led researchers to focus on deriving error bounds for DNNs, see e.g. [156]. Moreover, CNN models have been mostly studied for handling high-dimensional data at input and not at output, as in [82]. As a consequence, the available literature is left with a missing piece, which is to understand the approximation properties of convolutional layers when reconstructing functional signals. Here, we aim at addressing this issue.

6.1.2 Our contribution

Let $\boldsymbol{\mu} \to u_{\boldsymbol{\mu}}$ be some nonlinear operator whose output are functions $u_{\boldsymbol{\mu}} : [0,1]^d \to \mathbb{R}$ defined over the unit hypercube. We provide error bounds for the approximation of such an operator via a CNN model $\Phi : \mathbb{R}^p \to \left(\mathbb{R}^N\right)^d$. In particular, we characterize the model architecture in terms of the approximation error

$$\mathcal{E} := \sup_{\boldsymbol{\mu} \in \Theta} \sup_{\mathbf{j} \in \{1,\dots,N\}^d} |u_{\boldsymbol{\mu}}(x_{\mathbf{j}}) - \Phi_{\mathbf{j}}(\boldsymbol{\mu})|,$$

where $\Theta \subset \mathbb{R}^p$ is some parameter space and $\{x_{\mathbf{j}}\}_{\mathbf{j}} \subset [0,1]^d$ is a suitable $N \times \cdots \times N$ grid. By doing so, we also provide a clear interpretation to the model hyperparameters, including the number of dense and convolutional layers, the amount of active weights and the number of convolutional channels. In what follows, we limit ourselves to the 1-dimensional case, $d = 1$, even if the ideas at the core of our proofs can be extended to higher dimensions with little effort.

We report below our main result, Theorem 6.1, in which we characterize the approximation error in terms of the complexity of a DNN model comprised of a dense and a convolutional block. In what follows, we denote by $H^s(\Omega)$ the Sobolev space of s-times weakly-differentiable maps with square-integrable derivatives.

Theorem 6.1. *Let $\Omega := (0,1)$ and let $\{x_j\}_{j=1}^{N_h} \subset \Omega$ be a uniform grid with stepsize $h = 2^{-k}$. Let $\Theta \subset \mathbb{R}^p \ni \boldsymbol{\mu} \to u_{\boldsymbol{\mu}} \in H^s(\Omega)$ be a (nonlinear) operator, where Θ is a compact domain and $s \geq 1$. For some $r \geq 1$, assume that the operator is r-times continuously Fréchet differentiable. For any $0 < \varepsilon < 1/2$, there exists a deep neural network $\Phi : \mathbb{R}^p \to \mathbb{R}^{N_h}$ such that*

$$|u_{\boldsymbol{\mu}}(x_j) - \Phi_j(\boldsymbol{\mu})| < \varepsilon$$

uniformly for all $\boldsymbol{\mu} \in \Theta$ and all $j = 1, \dots, N_h$. Additionally, Φ can be defined as the composition of a fully connected network ϕ and a convolutional neural network Ψ, i.e. $\Phi = \Psi \circ \phi$, such that the overall architecture has at most

i) $C \log(1/\varepsilon)$ dense layers, with ReLU activation, and $C \log(1/h)$ convolutional layers,

ii) $C\varepsilon^{-2/(2s-1)} \left[\varepsilon^{-p/r} \log(1/\varepsilon) + \log(1/h)\right]$ active weights,

iii) $C\varepsilon^{-2/(2s-1)}$ channels in input and output,

where $C > 0$ is a constant dependent on Θ and on the operator $\boldsymbol{\mu} \to u_{\boldsymbol{\mu}}$, thus also on s, r, p.

In particular, the above result shows that:

(i) The number of dense layers depends logarithmically on the desired accuracy, while that of the convolutional layers depends logarithmically on the mesh resolution, i.e. on the number of discretization points.

(ii) The width of the dense block is related to the regularity of the operator itself, with smooth operators requiring less neurons.

(iii) The number of convolutional features depends on the regularity of the signals $u_{\boldsymbol{\mu}}$ at output.

Our proof of Theorem 6.1 is based on a link between CNNs and the Fourier transform, which we establish step-by-step through a sequence of technical Lemmas. For better readability, we have decided to postpone all the proofs and the technical results until the end of the Chapter.

6.2 Numerical validation

We present some numerical experiments that confirm the decay rates predicted in Theorem 6.1. We proceed as follows. First, we introduce the operator to be learned and we identify the smoothness indices, s and r, that appear in Theorem 6.1. Then, we fix a guess architecture $\Phi^{(1)}$ that serves as a starting point. Following the ideas of Theorem 6.1, we prescribe $\Phi^{(1)}$ as a DNN that is made of two blocks,

$$\Phi^{(1)} = \Psi^{(1)} \circ \phi^{(1)},$$

where $\phi^{(1)}$ is dense, while $\Psi^{(1)}$ is of convolutional type. More precisely:

- we let $\phi^{(1)}$ have L_1 hidden layers of constant width w_1, while we equip the output layer with $m_1(2^\ell + 5)$ neurons. In this way, the output of $\phi^{(i)}$ can be reshaped in the form of a $m_1 \times (2^\ell + 5)$ matrix, which allows the CNN block to interpret it as a discrete signal having m_1 features of length $2^\ell + 5$;

- we let $\Psi^{(1)} := R_2 \circ \tilde{\Psi}^{(1)} \circ R_1$, where the R_i are auxiliary reshape operations, whereas $\tilde{\Psi}^{(1)}$ contains the actual convolutional part. In particular, R_1 is used to reshape the output of $\phi^{(1)}$ as $m_1 \times (2^\ell + 5)$. Conversely, $\tilde{\Psi}^{(1)}$ is comprised of $k - \ell + 1$ transposed convolutional layers, where $k := \log_2(1/h)$ depends on the final grid resolution. All these layers have m_1 channels at input and output, grouped by m_1 (cf. Definition 6.1). The architecture of $\tilde{\Psi}^{(1)}$ is then further supplemented with a convolutional layer having a single output channel, and a terminal transposed convolution. All layers in $\tilde{\Psi}^{(1)}$ use a kernel of size 5 and a stride of 2, except for the last one, which resorts to the default stride of 1. With this set up, the output of $\tilde{\Psi}^{(1)} \circ R_1 \circ \phi^{(1)}$ is guaranteed to have shape $1 \times (2^{k+1} + 1)$. The final reshaping, R_2, is then used to flatten and half the output, leaving us with the correct dimension, that is, $2^k + 1$.

Table 6.1 reports in full detail the complete structure of the neural network architecture. We remark that the latter is itself parametrized by five hyperparameters, namely $L = L_1, w = w_1, m = m_1, k$ and ℓ. The first three allow us to tune the overall complexity of the model, and we shall exploit them to verify the estimates in Theorem 6.1. Conversely, the last two, k and ℓ, are problem dependent and we shall fix their value for each experiment alone. In fact, k is related to the grid discretization, while ℓ is used to define an intermediate level of resolution.

We initialize all the weights and biases of $\Phi^{(1)}$ randomly, following the approach introduced by He et al. in [83]. We then train $\Phi^{(1)}$ over a training set $\{\boldsymbol{\mu}_i, u_{\boldsymbol{\mu}_i}\}_{i=1}^{N_{\text{train}}}$ in such a way that the loss function below is minimized

$$\mathcal{L}(\Phi^{(1)}) := \frac{1}{N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} \left(h \sum_{j=1}^{N_h} |u_{\boldsymbol{\mu}_i}(x_j) - \Phi_j^{(1)}(\boldsymbol{\mu}_i)|^2 \right), \tag{6.1}$$

where $x_1, \dots, x_{N_h}$ is some dyadic partition of $(0,1)$ associated to a given grid resolution $h = 2^{-k}$. We then evaluate $\Phi^{(1)}$ over a test set of unseen instances $\{\boldsymbol{\mu}_i^{\text{test}}, u_{\boldsymbol{\mu}_i^{\text{test}}}\}_{i=1}^{N_{\text{test}}}$ in order to compute the empirical uniform error, given by

$$E(\Phi^{(1)}) = \max_{i,j} |u_{\boldsymbol{\mu}_i^{\text{test}}}(x_j) - \Phi_j^{(1)}(\boldsymbol{\mu}_i^{\text{test}})|. \tag{6.2}$$

Then, we exploit Theorem 6.1 in an attempt to define a second architecture, $\Phi^{(2)}$, that can be twice as accurate, i.e. such that $E(\Phi^{(2)}) \approx E(\Phi^{(1)})/2$. We do this as follows:

- we update the number of channels according to (iii) in Theorem 6.1. In particular, up to rounding operations, we let

$$m_2 := 2^{2/(2s-1)} m_1;$$

- we increase the with of the dense layers coherently with (ii) in Theorem 6.1, that is

$$w_2 := \sqrt{2^{p/r + 2/(2s-1)}} w_1,$$

  where the square root comes from the fact that a dense layer from $\mathbb{R}^w \to \mathbb{R}^w$ carries $\mathcal{O}(w^2)$ active weights;

- as suggested by (i) in Theorem 6.1, we also increase the number of dense hidden layers. In principle, the depth of the dense block should be increased by a constant factor $C \log(2)$. In practice, we let

$$L_2 = L_1 + l,$$

  where l is either 1 or 2. This is to ensure that the obtained architectures are still feasible to train, as very deep models may become hard and expensive to optimize.

Layer	Input	Output	Active weights
Dense	p	w	pw
Dense	w	w	w^2
... *Repeat last layer L-1 times* ...			
Dense	w	$(2^\ell+5)m$	$(2^\ell+5)wm$
Reshape	$(2^\ell+5)m$	$m\times(2^\ell+5)$	-
ConvTr$_{5,2}$	$m\times(2^\ell+5)$	$m\times(2^{\ell+1}+13)$	$5m$
... *Repeat last layer $k-\ell+2$ times* ...			
Conv$_{5,2}$	$m\times(2^{k+2}-3)$	$1\times(2^{k+1}-3)$	$5m$
ConvTr$_{5,1}$	$1\times(2^{k+1}-3)$	$1\times(2^{k+1}+1)$	5
Reshape	$1\times(2^{k+1}+1)$	$2^{k+1}+1$	0
Truncate	$2^{k+1}+1$	2^k+1	0

Table 6.1: Parametric architecture for the numerical experiments, Section 6.2. The dense block depends on the structural hyperparameters p (input dimension), w (width of the hidden layers), L (number of hidden layers-1), m and ℓ (intermediate output). Conversely, the design of the convolutional part depends on m (number of channels), ℓ and $k=\log_2(1/h)$ (depth of the convolutional block, upto a constant), where h is the stepsize of the discretization. Here, Conv$_{s,t}$ = Convolutional layer with kernel of size s and stride t; ConvTr$_{s,t}$ = Transposed convolutional layer with kernel of size s and stride t; Truncate = weightless layer that keeps only the central 2^k+1 components of the input. All learnable layers, except the last one, employ the 0.1-leakyReLU activation.

We then train $\Phi^{(2)}$ and iterate the above steps to generate $\Phi^{(3)}$, so that

$$E(\Phi^{(j)})\propto 2^{-j}.$$

We highlight that, according to Theorem 6.1, this procedure should be robust with respect to the space discretization. In other words, we expect to obtain similar results regardless of the number of grid points employed in the discretization. To assess whether this behavior is actually observed in practice, we repeat our analysis for different mesh step sizes $h=2^{-k}$ (when possible).

6.2.1 Benchmark example

To start, we consider the approximation of an operator that is defined analytically. More precisely, let $\Theta=[0,1]\times[0,1]\times[1,2]\subset\mathbb{R}^3$. For any fixed $\boldsymbol{\mu}=(\mu_1,\mu_2,\mu_3)\in\Theta$ let

$$u_{\boldsymbol{\mu}}(x)=\mu_3|x-\mu_1|^3\mathrm{e}^{-\mu_2 x}.$$

We are interested in learning the map $\boldsymbol{\mu}\to u_{\boldsymbol{\mu}}$. To this end, we note that $\{u_{\boldsymbol{\mu}}\}_{\boldsymbol{\mu}\in\Theta}\subset H^3(\Omega)\setminus H^4(\Omega)$. Also, the operator is at most twice differentiable with respect to $\boldsymbol{\mu}$, as its third derivative becomes discontinuous. According to the notation in Theorem 6.1, this results in $s=3$ and $r=2$. However, due to the boundness of the third derivative, we can actually apply Theorem 6.1 with an increased smoothness index, i.e. $r=3$ (see Remark 6.4.3 in the technical proofs Section).

For the space discretization, we consider three different mesh resolutions, $h=2^{-5},2^{-6},2^{-7}$, corresponding respectively to $N_h=33,65,129$ grid points. We train the networks by minimizing (6.1) via the so-called L-BFGS optimizer, where the training set consists of 500

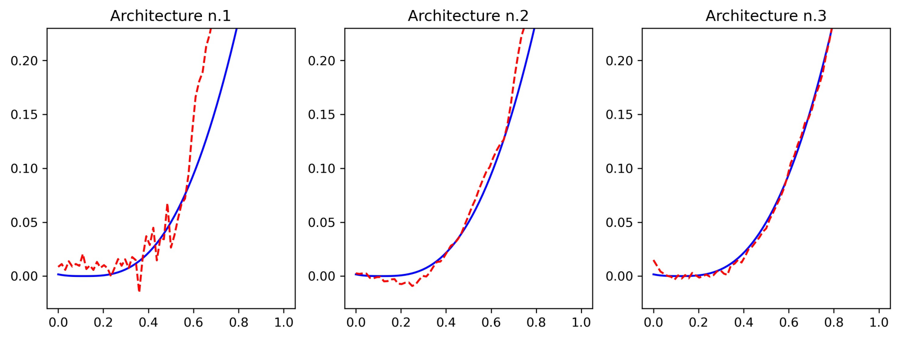


Figure 6.2: Benchmark example, Section 6.2.1. In blue, an instance $u_{\boldsymbol{\mu}}$ coming from the test set, here for $\boldsymbol{\mu} = [0.1125, 0.5409, 0.1255]$. The red dashed lines report the approximations proposed by the three DNNs, respectively $\Phi^{(1)}, \Phi^{(2)}, \Phi^{(3)}$. Grid resolution is $h = 2^{-6}$.

Model	ℓ	m_j	w_j	L_j	Active weights	$E(\Phi^{(j)})$
$\Phi^{(1)}$	3	5	1	1	223	0.884
$\Phi^{(2)}$	3	7	2	2	407	0.489
$\Phi^{(3)}$	3	9	3	3	653	0.177

Table 6.2: Architectures and corresponding errors for the Benchmark example, Section 6.2.1. Results are reported limitedly to the case of grid resolution $h = 2^{-7}$. The hyperparameters read as in Section 6.2, that is: m_j = maximum number of convolutional features in the CNN block, upto a multiplicative constant; w_j = number of neurons per dense layer; L_j = depth of the dense block. The errors $E(\Phi^{(j)})$ are computed as in Equation (6.2).

randomly sampled parameters instances. We do not use batching strategies and we set the learning rate to its default value of 1. To avoid possible biases introduced by the optimization, we initialize and train each architecture multiple times (here, five), only to keep the best out of all the training sessions. This is a common practice known as *ensemble training.* For our starting architecture, $\Phi^{(1)}$, we set

$$m_1 = 5,\ w_1 = 1,\ L_1 = 1.$$

In this case, we have $2^{2/(2s-1)} \approx 1.32$ and $\sqrt{2^{p/r+2/(2s-1)}} \approx 1.62$, since $p = 3$. In particular, our strategy for enriching the architectures can be stated as follows: to obtain a model that is twice as accurate, we increase the number of channels in the convolutional layers by nearly 30%, while we add about 60% new neurons to the dense layers. The theory also suggests to increase the depth of the dense block by some constant factor l. Here, we let $l = 1$. Finally, for this numerical experiment we choose a fixed coarse resolution of $13 = 2^{\ell} + 5$, that is, we let $\ell = 3$ in Table 6.1.

Results are in Table 6.2, Figures 6.2 and 6.3. The first picture compares the output of the three architectures with that of the operator, for an unseen value of the input parameter $\boldsymbol{\mu}$. The quality of the approximation clearly increases as we consider richer and richer models. In general, we see that the estimated signals are rougher compared to the ground truth. This, however, is most likely due to our use of the leaky ReLU activation (which we chose in order to be consistent with Theorem 6.1): other activations may lead to smoother results, with a possible benefit in terms of approximation properties, see e.g. [79], or training strategies, see e.g. [137]. We also note that the regions with lower regularity are the most difficult to capture, coherently with what we expected. Indeed, the architectures mostly struggle in capturing flat regions, which is understandable as these entail discontinuities in the higher-derivatives. Finally, Figure 6.3, reports the errors $E(\Phi^{(j)})$ in comparison with

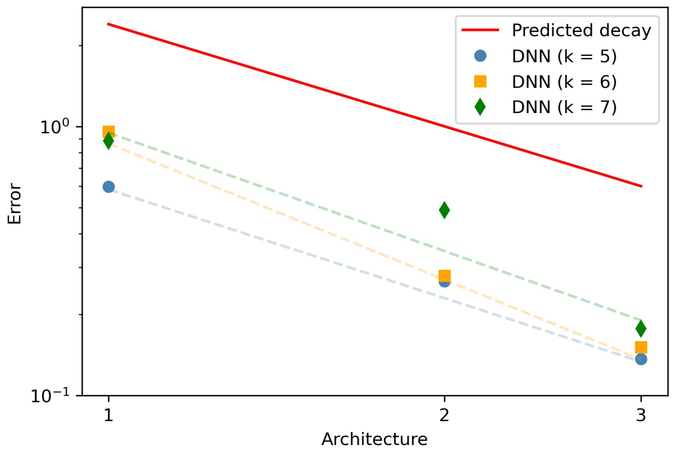


Figure 6.3: Numerical validation of Theorem 6.1 for the Benchmark example, Section 6.2.1. Axis are in loglog scale. The red line corresponds to the predicted decay rate, 2^{-j}, while the markers refer to the DNN models. Different markers correspond to different grid resolutions. The errors $E(\Phi^{(j)})$ obtained for the architectures $j = 1, 2, 3$, are computed as in Equation (6.2). Dashed lines are obtain through least square estimates.

the expected decay rate 2^{-j}. There, we see that the numerical results perfectly match the theory, regardless of the grid resolution that is considered.

6.2.2 Application to a parametrized time-dependent nonlinear PDE

We now consider a benchmark consisting of a one-dimensional coupled PDE-ODE nonlinear system

$$\begin{cases} \mu \dfrac{\partial u_{\boldsymbol{\mu}}}{\partial t} - \mu^2 \dfrac{\partial^2 u_{\boldsymbol{\mu}}}{\partial x^2} + R(u_{\boldsymbol{\mu}}) + w_{\boldsymbol{\mu}} = 0, & (x,t) \in \Omega \times (0,T) \\ \dfrac{dw_{\boldsymbol{\mu}}}{dt} + (2w_{\boldsymbol{\mu}} - 0.5u_{\boldsymbol{\mu}}) = 0, & (x,t) \in \Omega \times (0,T) \\ \dfrac{\partial u_{\boldsymbol{\mu}}}{\partial x}(0,t) = 50000t^3 e^{-15t}, & t \in (0,T) \\ \dfrac{\partial u_{\boldsymbol{\mu}}}{\partial x}(1,t) = 0, & t \in (0,T) \\ u_{\boldsymbol{\mu}}(x,0) = 0, \ w_{\boldsymbol{\mu}}(x,0) = 0, & x \in \Omega, \end{cases} \tag{6.3}$$

where $R(u_{\boldsymbol{\mu}}) := u_{\boldsymbol{\mu}}(u_{\boldsymbol{\mu}} - 0.1)(u_{\boldsymbol{\mu}} - 1)$, while $\Omega = (0,1)$ and $T = 2$. The above consists in a parametrized version of the monodomain equation coupled with the FitzHugh-Nagumo cellular model, describing the excitation-relaxation of the cell membrane in the cardiac tissue [64, 145]. System (6.3) has been discretized in space through linear finite elements, by considering $N_h = 2^k$, with $k \in \mathbb{N}$, grid points, and using a one-step, semi-implicit, first order scheme for time discretization with time-step $\Delta t = 5 \times 10^{-3}$; see, e.g., [152] for further details. The solution of the former problem consists in a parameter-dependent traveling wave, which exhibits sharper and sharper fronts as the parameter μ gets smaller. The numerical transmembrane potential solution $u_{\boldsymbol{\mu}}$ represent the ground truth data in the experiments reported in the following.

Here, we consider the map $(\mu, t) \to u_{\boldsymbol{\mu}}(\cdot, t)$ as our operator of interest. In particular, the two dimensional vector parameter $\boldsymbol{\mu} := (\mu, t)$ consists of the scalar parameter μ and the time variable t. We let $\boldsymbol{\mu}$ vary in the (time-extended) parameter space $\Theta := \Theta_0 \times [0, T]$, where $\Theta_0 := 5 \cdot [10^{-3}, 10^{-2}]$. In this case, it is not straightforward to identify the smoothness indices s and r. The numerical simulations show that the solutions $u_{\boldsymbol{\mu}}$ to (6.3) tend to have sharp gradients for certain values of the scalar parameter μ. In light of this, we let $s = 1$; if the

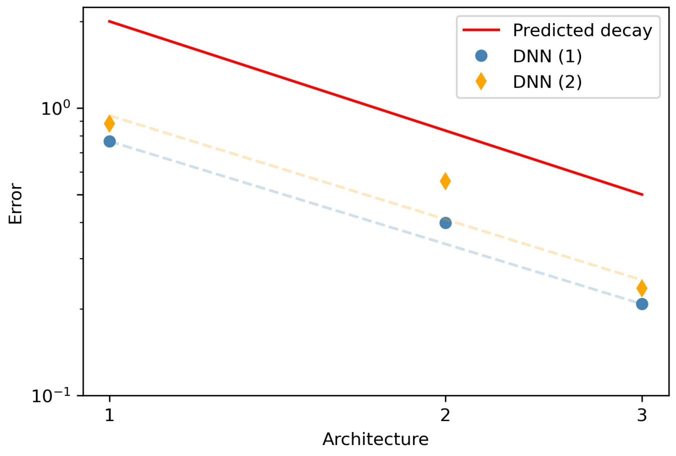


Figure 6.4: Numerical validation of Theorem 6.1 for the coupled FitzHugh-Nagumo problem, Section 6.2.2. Axis read in loglog scale. The red line corresponds to the predicted decay rate, 2^{-J}, while the blue markers report the errors $E(\Phi^{(j)})$ obtained for the architectures $j = 1, 2, 3$. Different markers correspond to different choices of the initial guess architecture $\Phi^{(1)}$, respectively. Errors are computed accordingly to Equation 6.2.

solutions are actually smoother, then we expect the errors to decay faster than the predicted rate. Conversely, we make the assumption that $r = +\infty$, i.e. that the parameter-to-solution map is infinitely differentiable. We remark that the constant C appearing in Theorem 6.1 actually depends on r. To this end, we make the further assumption that $C = C(r)$ is bounded with respect to r. This is a rather restrictive assumption, but we expect the latter to hold for analytic operators with fast decaying coefficients: indeed, in this case, one could adapt the proof of Theorem 6.1 by replacing the result due to Yarotsky with a stronger one, such as, e.g., Theorem 3.9 in [176].

As a starting point, we build our reference architecture, $\Phi^{(1)}$, by considering the following structural hyperparameters

$$m_1 = 1,\ w_1 = 5,\ L_1 = 1.$$

In this case, we have $2^{2/(2s-1)} = 2^{p/r+2/(2s-1)} = 4$. In particular, in order to half the test error, Theorem 6.1 suggests to quadruplicate the number channels in the convolutional layers, and to double that of the neurons in the dense layers. Regarding the depth of the dense block, instead, we increase it by a constant factor of $l = 1$ when moving from a simpler architecture to a more complex one. In this case, we do not assess the model performance for varying resolution levels as we stick to the same grid employed by the Finite Element solver. Instead, to collect more data, we repeat the same analysis for a different guess architecture, namely

$$m_1 = 1,\ w_1 = 2,\ L_1 = 2.$$

To collect the training and test sets, we proceed as follows. We sample $N_{train} = 20$ equally spaced values for the scalar parameter $\mu \in \Theta_0$, and we consider their midpoints to obtain $N_{test} = 19$ test instances. For each $\mu \in \Theta_0$ fixed, we then extract uniformly $N_t = 25$ time snapshots from the global trajectory defined over the interval $[0, T]$. Once again, we train the DNN models using the L-BFGS optimizer (no batching, learning rate $= 1$).

Results are reported in Figures 6.4 and 6.5. As for the benchmark example, we see that the DNN models become more and more expressive as we move from $\Phi^{(1)}$ to $\Phi^{(3)}$. Furthermore, the error trend, reported in Figure 6.4, is in agreement with the estimates presented in Theorem 6.1 regardless of the initial guess for the architecture. Note, once again, that here we only consider one resolution level, as we employ the same step size h adopted by the Finite Element solver.

Model	m_j	w_j	L_j	Active weights	$E(\Phi^{(j)})$
$\Phi^{(1)}$	1	5	1	225	0.765
$\Phi^{(2)}$	4	10	2	1’705	0.398
$\Phi^{(3)}$	16	20	3	13’085	0.208

Table 6.3: Architectures and corresponding errors for the coupled FitzHugh-Nagumo problem, Section 6.2.2. Results are reported limitedly to one of the initial guess architectures. Hyperparameters read as in Section 6.2 and Table 6.2. Errors are computed as in Equation (6.2).

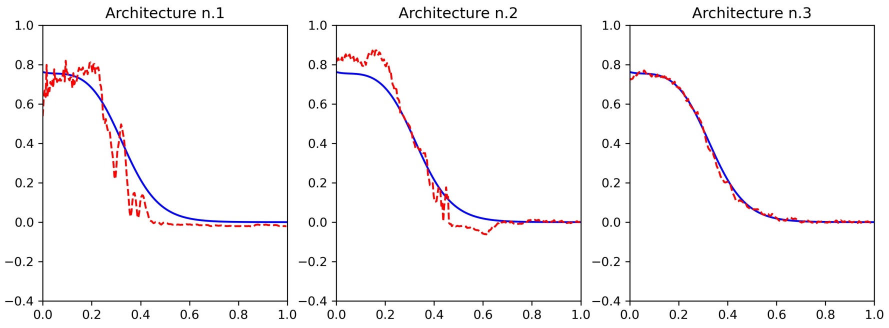


Figure 6.5: Learning the parameter-to-solution operator of a parametrized time-dependent nonlinear PDE, Section 6.2.2. In blue, a snapshot $u_{\boldsymbol{\mu}}(\cdot, t)$ coming from the test set, here for $\mu = 0.0488$ and $t = 0.7250$. The red dashed lines correspond to the approximations proposed by the three DNN models, respectively $\Phi^{(1)}, \Phi^{(2)}, \Phi^{(3)}$.

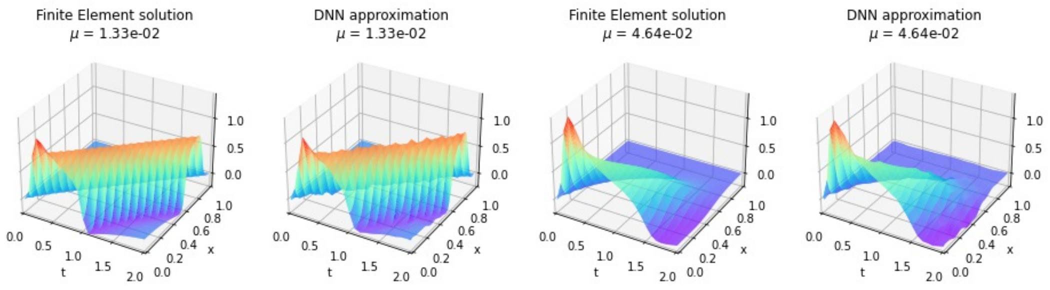


Figure 6.6: Learning the parameter-to-solution operator of a parametrized time-dependent nonlinear PDE, Section 6.2.2. Comparison between Finite Element solutions and DNN approximations for different μ. The first and the third plot report the spacetime surface $[0, T] \times \Omega \to \mathbb{R}$ representing the Finite Element simulation, thus $(t, x) \to u_{\mu}(x, t)$. Conversely, the second and the fourth picture show the corresponding DNN approximation over the same spatial grid, $(t, x_j) \to \Phi_j(t, \mu)$. Here, Φ is constructed considering the third architecture generated during the iterative augmentation process in Section 6.2, starting from the second guess architecture in Section 6.2.2.

Since we included time as an additional parameter, the plots in Figure 6.5 fix both the scalar parameter μ and the time instant t. However, we recall that according to Equation (6.2) the model was evaluated in terms of worst-case errors. In particular, the quality of the approximation is guaranteed over the whole time interval $[0, T]$ and for any choice of the scalar parameter $\mu \in \Theta_0$. Figure 6.6, shows the overall dynamics of the solution for two different choices of μ, with a comparison between Finite Element solutions and DNN approximations. Despite containing a few numerical artifacts, we see that the DNN model fully captures the general behavior of the solutions, both in the hyperbolic and diffusive case ($\mu = 1.33 \cdot 10^{-2}$ and $\mu = 4.64 \cdot 10^{-2}$ respectively). Of note, the spurious oscillations in the DNN approximation are in perfect agreement with the errors reported in Table 6.3. Accordingly to Theorem 6.1, these can be removed by considering larger architectures and, possibly, more training data.

6.3 Conclusions

In this Chapter, we have established and verified theoretical error bounds for the approximation of nonlinear operators by means of CNNs. Our results shed a light on the role played by convolutional layers and their hyperparameters, such as input-output channels, depth and others. In particular, they show how operator learning problems can be decoupled in two parts: on the one hand, the difficulty in characterizing the dependence with respect to the input parameters; on the other hand, the issue in having to reconstruct complex space-dependent outputs. The presented research is original and timely. Indeed, at the best of our knowledge, all the available results on DNNs and operator learning do not address the peculiar properties of CNNs, instead they consider classic fully connected architectures. Conversely, those works that focus on CNN models are typically not framed in the context of operator learning.

Our analysis is limited to the 1-dimensional case, $d = 1$, that is when the output of the operator are functions defined over an interval. However, we note that the main ideas underlying our proofs can be extended to higher dimensions with little effort. The critical points are Lemmas 6.1, 6.3 and 6.4 (see the next Section). To generalize the first two, one needs to define suitable convolutional layers that are able to advance along different dimensions separately, which can be carried out via 2D and 3D convolutions whenever $d = 2, 3$. Conversely, Lemma 6.4 has to be adapted in a proper way, since it becomes trickier to turn generic maps $f : [0, 1]^d \to \mathbb{R}$ onto periodic functions. Furthermore, as the spatial dimension d plays an important role in Sobolev inequalities, it may be convenient to replace the output space $H^s(\Omega)$ with other functional spaces, such as $W^{s,\infty}(\Omega)$ or $\mathcal{C}^s(\overline{\Omega})$, when addressing the case $d > 1$.

Nevertheless, we believe that our results motivate the recent success of CNNs, especially in areas such as Reduced Order Modeling of PDEs. This is because, as shown in Theorem 6.1, smooth outputs are those that are better approximated by CNNs: as solutions to partial differential equations often enjoy regularity properties, this makes them an appealing area of application for the proposed analysis. In general, these considerations further promote the practical use of CNNs, as well as their theoretical study from a purely mathematical point of view.

6.4 Technical proofs

In this Section, we report the proof of Theorem 6.1 together with the Lemmas propedeutical to it. In what follows, we make use of the embedding $\mathbb{C} \hookrightarrow \mathbb{R}^4$,

$$\mathbf{z} \to [\mathrm{Re}(\mathbf{z}), \mathrm{Im}(\mathbf{z}), \mathrm{Re}(\mathbf{z}), \mathrm{Im}(\mathbf{z})],$$

to represent complex numbers. This will come in handy when trying to mimic the algebra of complex numbers using neural networks. With this convention, we also let $\mathbb{C}^n \hookrightarrow \mathbb{R}^{4\times n}$ in the obvious way. In particular, when defining CNN architectures of the form $\phi : \mathbb{R}^{4\times n} \to \mathbb{R}^{4\times m}$, we shall write $\phi : \mathbb{C}^n \to \mathbb{C}^m$. However, this is only a matter of notation: in practice, all the networks considered from now on will never deal with complex values (neither at input or output) but only with the equivalent representation in $\mathbb{R}^4$.

6.4.1 Interpolation of the discrete Fourier transform

Convolution operations are intimately connected to the Fourier transform via the so-called Convolution Theorem, see e.g. [97]. Here, we further investigate this connection by deriving some preliminary results that will serve as building blocks for Theorem 6.1. The idea can be stated as follows. Given any dyadic partition of the unit interval,

$$\{x_j\}_{j=1}^{N_h} := \{(j-1)2^{-k}\}_{j=1}^{N_h},$$

where $N_h := 2^k + 1$, and any positive integer m, we construct a CNN model $\mathcal{S}_m$ that interpolates the (discrete) map

$$[\mathbf{z}_{-m}, \cdots, \mathbf{z}_m] \to \left[\sum_{k=-m}^{m} \mathbf{z}_k e^{2\pi \mathbf{i} k x_0}, \dots, \sum_{k=-m}^{m} \mathbf{z}_k e^{2\pi \mathbf{i} k x_{N_h}}\right]$$

associating the coefficient $\mathbf{z}_k \in \mathbb{C}$, $k = -m, \dots, m$, to the truncated Fourier transform at the points x_j, $j = 1, \dots, N_h$, with $\mathbf{i}$ the imaginary unit. The construction of $\mathcal{S}_m$ is detailed step-by-step, starting at Lemma 6.1 and concluding with Lemma 6.3. The proofs are constructive, as they explicitly describe how to implement $\mathcal{S}_m$. In particular, we are able to characterize the complexity of $\mathcal{S}_m$ in terms of those specific features that are typical of CNNs, such as depth, kernel size, stride, dilation, padding, and number of input-output channels. For instance, we shall see in Lemma 6.3 that the depth of $\mathcal{S}_m$ grows logarithmically with the grid resolution, while the active weights grow linearly with m. These observations will play a key role when deriving the upper bounds in Lemma 6.4 and Theorem 6.1.

Lemma 6.1. *For any $k \in \mathbb{N}_+$ and any $\mathbf{z} \in \mathbb{C}$ there exists a convolutional neural network $\phi_{\mathbf{z}}^k : \mathbb{C}^{2^{k-1}} \to \mathbb{C}^{2^k}$ such that*

i) it is linear (no activation at any level),

ii) it only employs 1D convolutional and reshaping operations,

iii) it has an architecture of at most six layers,

iv) the input and the output of its convolutional layers have at most 8 channels,

v) the kernels of the convolutional layers have size at most equal to 2,

and such that

$$\phi_{\mathbf{z}}^k([\mathbf{w}_1, \dots, \mathbf{w}_{2^{k-1}}]) = [\mathbf{w}_1, \mathbf{z}\mathbf{w}_1, \dots, \mathbf{w}_{2^{k-1}}, \mathbf{z}\mathbf{w}_{2^{k-1}}]$$

for all $\mathbf{w}_1, \dots, \mathbf{w}_{2^{k-1}} \in \mathbb{C}$.

Proof. Let $n = 2^{k-1}$ be the (complex) input dimension. Let f_1 be a 1D transposed convolutional layer with the following specifics. The layer has four channels at input and four channels at output. It uses a 2-sized window that acts with a stride of 2. The layer has no bias and its weight matrix $\mathbf{W}_1 \in \mathbb{R}^{4\times4\times2}$, which is obtained by stacking together the convolutional kernels, is zero at all but six entries. These are given by the relations below

$$\begin{bmatrix} \mathbf{W}_1^{1,1,1} & \mathbf{W}_1^{1,1,2} \\ \mathbf{W}_1^{2,2,1} & \mathbf{W}_1^{2,2,2} \\ \mathbf{W}_1^{3,3,1} & \mathbf{W}_1^{3,3,2} \\ \mathbf{W}_1^{4,4,1} & \mathbf{W}_1^{4,4,2} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ \mathrm{Re}(\mathbf{z}) & \mathrm{Im}(\mathbf{z}) \\ -\mathrm{Im}(\mathbf{z}) & \mathrm{Re}(\mathbf{z}) \end{bmatrix}. \tag{6.4}$$

Note that, here, we have also listed some of the zero entries in $\mathbf{W}_1$: this is to facilitate the reader in understanding the purpose of f_1. The first block in (6.4) is used to mimic the action of the identity matrix. Conversely, the second block encodes a 2×2 matrix representation of the complex number $\mathbf{z}$. The idea is that these two blocks should provide a way of computing the map $\mathbf{w} \to [\mathbf{w}, \mathbf{zw}]$. However, for these computations to be actually carried out, we also need a further layer that performs a suitable summation of the outputs given by f_1. To this end, we define the second layer, f_2, as a 1D convolution that maps 4-channeled inputs onto 2-channeled outputs. The latter uses a 1-sized window that acts with a stride of 1. The layer has no bias and its weight matrix $\mathbf{W}_1 \in \mathbb{R}^{2\times4\times1}$ contains either zeros or ones. The positive entries are

$$\mathbf{W}_2^{1,1,1},\ \mathbf{W}_2^{1,2,1},\ \mathbf{W}_2^{2,3,1},\ \mathbf{W}_2^{2,4,1} = 1.$$

Then, $f_2 \circ f_1 : \mathbb{R}^{4\times n} \to \mathbb{R}^{2\times 2n}$, and, upto some basic calculations, we have

$$(f_2 \circ f_1)([\mathbf{w}_1, \dots, \mathbf{w}_n]) =$$
$$\begin{bmatrix} \mathrm{Re}(\mathbf{w}_1), & \mathrm{Im}(\mathbf{w}_1), & \dots, & \mathrm{Re}(\mathbf{w}_n), & \mathrm{Im}(\mathbf{w}_n) \\ \mathrm{Re}(\mathbf{zw}_1), & \mathrm{Im}(\mathbf{zw}_1), & \dots, & \mathrm{Re}(\mathbf{zw}_n), & \mathrm{Im}(\mathbf{zw}_n) \end{bmatrix}.$$

In practice, the desired output of $\phi_{\mathbf{z}}^k$ is already there, but we need to adjust the output dimension in order to match our convention for complex numbers.

To this end, we start by introducing a reshape operation $R_1 : \mathbb{R}^{2\times 2n} \to \mathbb{R}^{1\times 4n}$ that flattens the whole output. Then, we add a third convolutional layer, f_3, whose purpose is to double the entries in input. More precisely, we define f_3 has a 1D convolution that has 1 channel at input and 4 at output. The layer uses a 2-sized kernel that acts with a stride of 2. Once again, the layer introduces no bias and has a weight matrix $\mathbf{W}_3 \in \mathbb{R}^{4\times1\times2}$ given by

$$\mathbf{W}_3 = [[[1,0]],\ [[0,1]],\ [[1,0]],\ [[0,1]]].$$

With the notation adopted to represent complex numbers, the current action of $f_3 \circ R_1 \circ f_2 \circ f_1$ becomes

$$[\mathbf{w}_1, \dots, \mathbf{w}_n] \to [\mathbf{w}_1, \dots, \mathbf{w}_n, \mathbf{zw}_1, \dots, \mathbf{zw}_n].$$

Let us now act further on the output to sort the entries in the desired order. To do so, we introduce a 1D convolutional layer, f_4, that has a dilation factor of 2^k (this is because we want to group $\mathbf{w}_1$ with $\mathbf{zw}_1$, which is 2^k entries faraway, and so on). We let f_4 go from 4 to 8 channels, and employ a kernel of size 2 with unit stride. Once again, f_4 does not have a bias term, while its weight matrix satisfies

$$\mathbf{W}_4^{i,j,k} = \begin{cases} 1 & \text{if } i = j + 4(k-1) \\ 0 & \text{otherwise.} \end{cases}$$

At this point we have,

$$f_4 \circ f_3 \circ R_1 \circ f_2 \circ f_1 : [\mathbf{w}_1, \dots, \mathbf{w}_n] \to \begin{bmatrix} \mathbf{w}_1, & \dots, & \mathbf{w}_n \\ \mathbf{zw}_1, & \dots, & \mathbf{zw}_n \end{bmatrix},$$

and we only need to add a final reshaping operation, R_2. We define the latter as follows. First, it transposes the input by mapping $\mathbb{R}^{8\times n} \to \mathbb{R}^{n\times 8}$. Then, it performs the reshaping $\mathbb{R}^{n\times 8} \to \mathbb{R}^{2n\times 4}$, where entries are read by rows, and finally it transposes back the input so that it ends up in $\mathbb{R}^{4\times 2n} \cong \mathbb{C}^{2^k}$. Finally, letting $\phi_{\mathbf{z}}^k := R_2 \circ f_4 \circ f_3 \circ R_1 \circ f_2 \circ f_1$ concludes the proof. $\square$

Lemma 6.2. *Let $k \in \mathbb{N}_+$ and $h = 2^{-k}$. Let $\{x_1, \dots, x_{N_h}\}$ be an uniform partition of $[0,1]$ with stepsize h. For any $\omega \in \mathbb{R}$ there exists a convolutional neural network $\mathcal{F}_\omega : \mathbb{C} \to \mathbb{C}^{N_h - 1}$ such that* *i)* $\mathcal{F}_\omega$ *is linear (no activation at any level),*

ii) $\mathcal{F}_\omega$ *has at most depth* $C\log(1/h)$,

iii) $\mathcal{F}_\omega$ *has at most* $C\log(1/h)$ *active weights,*

iv) $\mathcal{F}_\omega(\mathbf{w}) = [\mathbf{w}e^{\mathbf{i}\omega x_1},\dots,\mathbf{w}e^{\mathbf{i}\omega x_{N_h-1}}]$.

where $C>0$ *is a constant independent on* h *and* ω*. Furthermore, up to reshape operations,* $\mathcal{F}_\omega$ *only uses 1D convolutional layers that have at most 8 channels at both input and output. Moreover, the kernel size of all convolutional layers in* $\mathcal{F}_\omega$ *is at most 2.*

Proof. Let $\mathbf{z}\in\mathbb{C}$. For all $j=1,\dots,k$, define the CNNs $\phi^j_{\mathbf{z}^{2^{j-1}}}$ as in Lemma 1. For the sake of simplicity assume that $k\geq 2$. Then it is straightforward to check that $\phi^1_{\mathbf{z}}:\mathbf{w}\to[\mathbf{w},\mathbf{wz}]$, while $\phi^2_{\mathbf{z}^2}\circ\phi^1_{\mathbf{z}}:\mathbf{w}\to[\mathbf{w},\mathbf{wz}^2,\mathbf{wz},\mathbf{wz}^3]$ and so on. In particular,

$$\phi^k_{\mathbf{z}^{2^{k-1}}}\circ\dots\circ\phi^1_{\mathbf{z}}:\mathbf{w}\to\pi_k\left([\mathbf{w},\mathbf{wz},\mathbf{wz}^2,\dots,\mathbf{wz}^{2^k-1}]\right)$$

where $\pi_k:\mathbb{C}^{N_h-1}\to\mathbb{C}^{N_h-1}$ is some (invertible) map that acts as a permutation over the entries. We now claim that this permutation can be nullified through the composition of three suitable reshape operations: that is, there exist three reshape layers $R^{(k)}_1,R^{(k)}_2,R^{(k)}_3$ such that

$$\pi_k^{-1}=R^{(k)}_3\circ R^{(k)}_2\circ R^{(k)}_1. \tag{6.5}$$

Before proving it, we note that (6.5) would immediately yield the desired conclusion. In fact, if we let $\mathbf{z}:=e^{\mathbf{i}\omega h}$, then (6.5) allows us to define $\mathcal{F}_\omega:=\pi_k^{-1}\circ\phi^k_{\mathbf{z}^{2^{k-1}}}\circ\dots\circ\phi^1_{\mathbf{z}}$, which results in the map

$$\mathcal{F}_\omega:\mathbf{w}\to[\mathbf{w}e^{\mathbf{i}\omega 0h},\mathbf{w}e^{\mathbf{i}\omega 1h},\dots,\mathbf{w}e^{\mathbf{i}\omega(N_h-2)h}].$$

Therefore, it is sufficient for us to prove that (6.5) holds. To this end, we define the three maps as

$$R^{(k)}_1:\mathbb{C}^{2^k}\to\mathbb{C}^{\overbrace{2\times\dots\times 2}^{k\text{ times}}},$$

where the output entries are filled by rows,

$$R^{(k)}_2:\mathbb{C}^{2\times\dots\times2}\to\mathbb{C}^{2\times\dots\times2},$$

which acts as a full transposition of the indexes, that is

$$R^{(k)}_2(\mathbf{Q}_{i_1,\dots,i_k})=\mathbf{Q}_{i_k,\dots,i_1},$$

and finally

$$R^{(k)}_3:\mathbb{C}^{2\times\dots\times2}\to\mathbb{C}^{2^k},$$

that flattens back the input. With this setup, we shall now prove (6.5) following an argument by induction over k. To start, let $k=2$. We choose to skip the trivial case $k=1$ to provide the reader with a more insightful computation that anticipates the ideas used later for the inductive step. In this case, we note that $\pi_k([\mathbf{0},\mathbf{1},\mathbf{2},\mathbf{3}])=[\mathbf{0},\mathbf{2},\mathbf{1},\mathbf{3}]$, where we write $\mathbf{k}=k+0\mathbf{i}\in\mathbb{C}$ to embed real numbers in $\mathbb{C}$. The reshape layers act on the latter vector as

$$R^{(k)}_3\circ R^{(k)}_2\circ R^{(k)}_1([\mathbf{0},\mathbf{2},\mathbf{1},\mathbf{3}])=$$
$$=R^{(k)}_3\circ R^{(k)}_2\left(\begin{bmatrix}\mathbf{0}&\mathbf{2}\\\mathbf{1}&\mathbf{3}\end{bmatrix}\right)=R^{(k)}_3\left(\begin{bmatrix}\mathbf{0}&\mathbf{1}\\\mathbf{2}&\mathbf{3}\end{bmatrix}\right)=$$
$$=[\mathbf{0},\mathbf{1},\mathbf{2},\mathbf{3}].$$

Since π_k is a permutation, the above directly implies (6.5). To conclude, we shall now prove the inductive step: assuming that (6.5) holds for k, we show that the statement is also true for $k+1$. For any $j \in \mathbb{N}$, let

$$\mathbf{E}_j := \pi_j\left(\left[\mathbf{0}, \dots, \mathbf{2}^j - \mathbf{1}\right]\right).$$

We split the vector $\mathbf{e}_{k+1}$ into halves by introducing the following notation

$$\mathbf{E}_{k+1} = [\mathbf{A}_1, \dots, \mathbf{A}_{2^k}, \mathbf{B}_1, \dots, \mathbf{B}_{2^k}],$$

so that $\mathbf{E}_{k+1}$ is the concatenation of the two (complex) vectors $\mathbf{A}, \mathbf{B} \in \mathbb{C}^{2^k}$. Following the construction in Lemma 6.1, it is not hard to see that

$$\mathbf{A} = 2\mathbf{E}_k, \quad \mathbf{B} = 2\mathbf{E}_k + 1, \tag{6.6}$$

where, with little abuse of notation, we intend $[\mathbf{v}_1, \dots, \mathbf{v}_n] + 1 := [\mathbf{v}_1 + 1, \dots, \mathbf{v}_n + 1]$. Let now

$$\mathbf{Z} := R_1^{(k+1)}(\mathbf{e}_{k+1}), \qquad \tilde{\mathbf{Z}} := R_2^{k+1}(\mathbf{Z}).$$

By definition, we have

$$\tilde{\mathbf{Z}}_{j_1,\dots,j_k,1} = \mathbf{Z}_{1,j_k,\dots,j_1} = \mathbf{A}_{(j_k-1)2^{1-1}+\dots+(j_1-1)2^{k-1}},$$

$$\tilde{\mathbf{Z}}_{j_1,\dots,j_k,2} = \mathbf{Z}_{2,j_k,\dots,j_1} = \mathbf{B}_{(j_k-1)2^{1-1}+\dots+(j_1-1)2^{k-1}}.$$

We note that, the subtensor of $\tilde{\mathbf{Z}}$ obtained by fixing the last index equal to 1 is equivalent to $R_2^{(k)} \circ R_1^{(k)}(\mathbf{A})$, and similarly for $\mathbf{B}$. In particular, if we let

$$\hat{\mathbf{A}} := R_3^{(k)} \circ R_2^{(k)} \circ R_1^{(k)}(\mathbf{A}), \quad \hat{\mathbf{B}} := R_3^{(k)} \circ R_2^{(k)} \circ R_1^{(k)}(\mathbf{B}),$$

then the action of the final layer, $R_3^{(k+1)}$, results in

$$R_3^{(k+1)}(\tilde{\mathbf{Z}}) = [\hat{\mathbf{A}}_1, \hat{\mathbf{B}}_1, \dots, \hat{\mathbf{A}}_{2^k}, \hat{\mathbf{B}}_{2^k}].$$

Finally, by recalling (6.6) and by applying our inductive hypothesis, we have

$$\hat{\mathbf{A}} = [\mathbf{0}, \mathbf{2}, \dots, \mathbf{2^{k+1}} - \mathbf{2}], \quad \hat{\mathbf{B}} = [\mathbf{1}, \mathbf{3}, \dots, \mathbf{2^{k+1}} - \mathbf{1}]$$

$$\implies R_3^{(k+1)}(\tilde{\mathbf{Z}}) = [\mathbf{0}, \mathbf{1}, \dots, \mathbf{2^{k+1}} - \mathbf{1}] = \pi_{k+1}^{-1}(\mathbf{E}_{k+1}),$$

which proves our original claim in (6.5). □

Lemma 6.3. *Let $k \in \mathbb{N}_+$ and $h = 2^{-k}$. Let $\{x_1, \dots, x_{N_h}\}$ be a uniform partition of $[0,1]$ with stepsize h. For any positive integer m, there exists a convolutional neural network $\mathcal{S}_m : \mathbb{C}^{2m+1} \to \mathbb{C}^{N_h}$ such that*

i) $\mathcal{S}_m$ is linear (no activation at any level),

ii) $\mathcal{S}_m$ has at most depth $C\log(1/h)$,

iii) $\mathcal{S}_m$ has at most $Cm\log(1/h)$ active weights,

iv) $\mathcal{S}_m$ uses convolutional layers with at most Cm channels,

v) for any complex vector $\mathbf{Z} = [\mathbf{z}_{-m}, \dots, \mathbf{z}_m] \in \mathbb{C}^{2m+1}$ one has

$$\mathcal{S}_m(\mathbf{Z})_i = \sum_{k=-m}^{m} \mathbf{z}_k e^{2\pi \mathbf{i} k x_i},$$

for all $i = 1, \dots, N_h$, where $\mathcal{S}_m(\mathbf{Z})_i$ is the ith component of the output vector $\mathcal{S}_m(\mathbf{Z})$.

Here, $C > 0$ is a universal constant independent on h and m. Furthermore, up to reshape operations, $\mathcal{S}_m$ only uses 1D convolutional layers whose kernel size does not exceed 2.

Proof. Fix any $k \in \{-m, \dots, m\}$. Let $\mathcal{F}_{(k)}$ be the CNN in Lemma 6.2 when $\omega = 2\pi k$. We note that, as k varies, the structure of $\mathcal{F}_{(-m)}, \dots \mathcal{F}_{(m)}$ does not change: these architectures have the same depth and they employ convolutional layers with the same specifics. Also, the reshaping operations entailed by the networks occur at the same locations. Therefore, we can stack all these models on top of each other to obtain a global CNN $\tilde{\mathcal{S}}_m$ such that

$$\tilde{\mathcal{S}}_m(\mathbf{Z}) = \left[\mathcal{F}_{(-m)}(\mathbf{z}_{-m}), \dots, \mathcal{F}_{(m)}(\mathbf{z}_m)\right]$$

where $\mathbf{Z} = [\mathbf{z}_{-m}, \dots, \mathbf{z}_m] \in \mathbb{C}^{2m+1}$ is a generic input vector. This can be done as follows. To stack $2m+1$ convolutional layers with c_{in} channels at input and c_{out} channels at output each, we define a single CNN layer with $(2m+1)c_{in}$ channels at input and $(2m+1)c_{out}$ at output. Then, to avoid the introduction of redundant kernels, we constrain the new layer to group its kernels in $(2m+1)$ subsets. This ensures the wished behavior, i.e. that we actually stack the outputs of the $2m+1$ original layers as if they work in parallel (thus each seeing only the part of interest of the input). Similarly, reshaping and transpositions can be easily stacked together. For instance, stacking $(2m+1)$ transpositions of the form $\phi : \mathbb{R}^{a\times b} \to \mathbb{R}^{b\times a}$ results in a map from $\mathbb{R}^{(2m+1)\times a\times b}$ to $\mathbb{R}^{(2m+1)\times b\times a}$. Since $\tilde{\mathcal{S}}_m$ takes values in $\mathbb{C}^{(2m+1)\times(N_h-1)}$, our next purpose is to append a further layer L such that

$$(L \circ \tilde{\mathcal{S}}_m)(\mathbf{Z}) = \sum_{k=-m}^{m} \mathcal{F}_{(k)}(\mathbf{z}_k) \in \mathbb{C}^{N_h-1}.$$

It is easy to see that L can be obtained with a convolutional layer having $(2m+1)$ channels at input and 1 at output, no stride or dilation, and a kernel of size 1 whose weight is constantly equal to 1. Finally, we note that for all $k \in \{-m, \dots, m\}$ we have

$$\mathrm{e}^{2\pi \mathbf{i} k x_1} = \mathrm{e}^{2\pi \mathbf{i} k 0} = 1 = \mathrm{e}^{2\pi \mathbf{i} k} = \mathrm{e}^{2\pi \mathbf{i} k x_{N_h}}$$

due to periodicity. Therefore, we may simply define $\mathcal{S}_m := A \circ L \circ \tilde{\mathcal{S}}_m$, where A has the only purpose of augmenting the input vector by appending a copy of its first component, that is

$$A([\mathbf{w}_1, \dots, \mathbf{w}_{N_h-1}]) = [\mathbf{w}_1, \dots, \mathbf{w}_{N_h-1}, \mathbf{w}_1].$$

This can be seen as a form of reshaping or padding. By construction, $\mathcal{S}_m$ satisfies (v). Similarly, (i) and (iv) hold. In fact, each of the $\mathcal{F}_{(k)}$ has length $C\log(1/h)$, where C is a common constant. Since we stacked them in parallel to get $\tilde{\mathcal{S}}_m$, our final model has depth $C\log(1/h) + 2 = \tilde{C}\log(1/h)$. Also, the CNNs $\mathcal{F}_{(k)}$ featured at most 8 channels, thus $\mathcal{S}_m$ uses no more than $(2m+1)8 = \tilde{C}m$ channels at input-output. Property (iii) follows similarly by recalling that we grouped the CNNs kernels in order to properly stack the architectures. □

6.4.2 Signals reconstruction

Our next goal is to show that, for any desired accuracy, there exists a single CNN architecture that is able to provide an approximation of any function belonging to a given smoothness class. More precisely, let $s \geq 1$ be a smoothness index, and fix some $m \in \mathbb{N}$. Let also $\{x_j\}_{j=1}^{N_h}$ be some dyadic grid defined over $\Omega = (0,1)$. We build a CNN model $\Psi : \mathbb{C}^{2m+1} \to \mathbb{R}^{N_h}$ such that

$$\forall f \in H^s(\Omega)\;\; \exists \mathbf{Z}_f \in \mathbb{C}^{2m+1} \;\text{ such that }\; \sup_{j=1,\dots,N_h} |f(x_j) - \Psi_j(\mathbf{Z}_f)| < Cm^{1/2-s}\|f\|_H^s(\Omega),$$

where $C = C(s) > 0$ is some constant and Ψ_j is the jth component of the CNN output. The above states that any smooth function f can be well approximated by Ψ, provided that the model is fed with a suitable input vector. As for Lemma 6.3, we characterize the network complexity in terms of depth, channels and active weights. Furthermore, we show that the map $f \to \mathbf{Z}_f$ can be realized by some continuous linear operator that depends, at most, on s. Before stating this rigorously in Lemma 6.4, it is worth to remark that this result concerns the approximation of any functional output in $H^s(\Omega)$. In particular, although the proof is based on classical estimates coming from the literature of Fourier series, no periodicity is required.

Lemma 6.4. *Let $k \in \mathbb{N}_+$ and $h = 2^{-k}$. Let $\{x_j\}_{j=1}^{N_h}$ be a uniform partition of $\Omega := (0,1)$ with stepsize h, so that $N_h = 2^k + 1$. For any positive integer m and a universal constant C independent on m and h, there exists a linear convolutional neural network $\Psi : \mathbb{C}^{2m+1} \to \mathbb{R}^{N_h}$ with*

i) at most $C\log(1/h)$ layers,

ii) at most $Cm\log(1/h)$ active weights,

iii) at most Cm channels in input and output, with kernels coming in $2m+1$ groups,

such that for any $s \geq 1$ and all $f \in H^s(\Omega)$ one has

$$\sup_{j=1,\dots,N_h} |f(x_j) - \Psi_j(Tf)| \leq cm^{1/2-s}\|f\|_{H^s(\Omega)}.$$

Here, $c = c(s) > 0$ and $T : H^s(\Omega) \to \mathbb{C}^{2m+1}$ are a positive constant and a continuous linear operator that depend on s, respectively.

Proof. Let us denote by $\mathbb{T}$ the 1-dimensional torus, $\mathbb{T} := \mathbb{R}/\mathbb{Z}$, so that the spaces $\mathcal{C}^k(\mathbb{T})$ refer to those functions that are k-times differentiable on the torus, namely

$$\mathcal{C}^k(\mathbb{T}) = \left\{f \in \mathcal{C}^k[0,1],\ f^{(j)}(0) = f^{(j)}(1)\ \ \forall j = 1,\dots,k\right\}.$$

We start by defining an operator $T_0 : H^s(\Omega) \to H^s(\Omega)$ that perturbs the signal at input to match suitable boundary conditions. To this end, we recall that there exist polynomials $p_0,\dots,p_{s-1}$ and $q_0,\dots,q_{s-1}$, of degree $2s-1$, such that

$$p_j^{(k)}(0) = \delta_{j,k}, \quad p_j^{(k)}(1) = 0,$$

$$q_j^{(k)}(0) = 0, \quad q_j^{(k)}(1) = \delta_{j,k},$$

for $j,k \in \{0,\dots,s-1\}$, see e.g. [186]. Then, the linear operator

$$\mathbf{v} \to P\mathbf{v} := \sum_{j=0}^{s-1} v_j(p_j - q_j)$$

maps any input vector $\mathbf{v} \in \mathbb{R}^{s-1}$ into a smooth polynomial with given boundary values. By recalling that $H^s(\Omega) \hookrightarrow \mathcal{C}^{s-1}(\Omega)$ thanks to classical Sobolev inequalities, we are then allowed to define

$$T_0 : f \to f + P[f^{(0)}(1) - f^{(0)}(0),\dots,f^{(s-1)}(1) - f^{(s-1)}(0)]$$

so that $T_0 : H^s(\Omega) \to H^s(\Omega)$. We now introduce the following notation for "periodicized signals". For any $f \in H^s(\Omega)$ we let $\tilde{f}$ be defined as

$$\tilde{f}(x) = \begin{cases}(T_0 f)(2x) & 0 \leq x \leq 1/2 \\ f(2x-1) & 1/2 < x \leq 1\end{cases}.$$

It is straightforward to see that $\tilde{f} \in H^s(\Omega) \cap \mathcal{C}^{s-1}(\mathbb{T})$. For instance,

$$\tilde{f}(0) = f(0) + \sum_{j=0}^{s-1}[f^{(j)}(1) - f^{(j)}(0)]\cdot[p_j(0) - q_j(0)]$$
$$= f(0) + [f(1) - f(0)]p_0(0) = f(1) = \tilde{f}(1),$$

while

$$\tilde{f}(1/2) = f(1) + \sum_{j=0}^{s-1}[f^{(j)}(1) - f^{(j)}(0)]\cdot[p_j(1) - q_j(1)]$$
$$= f(1) - [f(1) - f(0)]q_0(1) = f(0) = \lim_{x\to\frac{1}{2}^+}\tilde{f}(x).$$

Similar calculations hold for the derivatives as well. Furthermore, the mapping $f \to \tilde{f}$ is linear and continuous, in the sense that for some constant $C > 0$, depending only on s, we have

$$\|\tilde{f}\|_{H^s(\Omega)} \leq C\|f\|_{H^s(\Omega)} \tag{6.7}$$

for all $f \in H^s(\Omega)$. With this construction, for any positive integer m, let $S_m\tilde{f}$ be the m truncated Fourier series of the function $\tilde{f}$,

$$\left(S_m\tilde{f}\right)(x) = \sum_{j=-m}^{m} c_{\tilde{f}}^{k} e^{2\pi \mathbf{i} kx}$$

where

$$c_{\tilde{f}}^{k} := \int_{\Omega} \tilde{f}(x) e^{-2\pi \mathbf{i} kx} dx.$$

Since $\tilde{f} \in \mathcal{C}^{s-1}(\mathbb{T})$ and its s-derivative is in $L^2(\mathbb{T})$, by exploiting classical estimates of Fourier analysis, we have the error bound

$$\|\tilde{f} - S_m\tilde{f}\|_{L^\infty(\Omega)} \leq \sqrt{\frac{2}{2s-1}} m^{1/2-s} \|\tilde{f}\|_{H^s(\Omega)}. \tag{6.8}$$

Let now $T : H^s(\Omega) \to \mathbb{C}^{2m+1}$ be defined as

$$T : f \to \left[c_{\tilde{f}}^{-m}, \dots, c_{\tilde{f}}^{m}\right],$$

so that T maps each signal into the Fourier coefficients of its periodic alias. Let $\{y_1, \dots, y_{2N_h-1}\}$ be a uniform partition of (0,1) that is twice as fine as the original one $\{x_1, \dots, x_{N_h}\}$, that is $y_{j+1} - y_j = h/2$. With this partition as a reference, let then $\mathcal{S}_m$ be the CNN in Lemma 6.3. By definition, we have

$$\left(S_m\tilde{f}\right)(y_i) = \mathcal{S}_m(Tf)_i.$$

Finally, let $\Psi := E \circ R \circ \mathcal{S}_m$, where

- R is a reshape truncation layer,

  $$R(\mathbf{w}_1, \dots, \mathbf{w}_{2N_h-1}) = [\mathbf{w}_{N_h}, \dots, \mathbf{w}_{2N_h-1}],$$

  that we use to remove the undesired output. Note in fact that, the signal $\tilde{f}$ over $(1/2, 1)$ is practically f over $(0, 1)$. Thus, in light of (6.8), we are only interested in the second half of the output.

- $E : \mathbb{C}^{N_h} \to \mathbb{R}^{N_h}$ is the embedding that only keeps the real part of the input. This can also be seen as a reshape layer with a truncation at the end. Since f, and thus $\tilde{f}$, are real valued, so are $S_m\tilde{f}$ and $\mathcal{S}_m(Tf)$. Therefore, we are not losing any information.

Finally, let $j \in \{1, \dots, N_h\}$. We have

$$\begin{aligned}|f(x_j) - \Psi_j(Tf)| &= |f(x_j) - \mathcal{S}_m(Tf)_{N_h-1+j}| = \\ &= |\tilde{f}(1/2 + x_j/2) - \mathcal{S}_m(Tf)_{N_h-1+j}| = \\ &= \left|\tilde{f}(y_{N_h-1+j}) - \left(S_m\tilde{f}\right)(y_{N_h-1+j})\right|.\end{aligned}$$

Thus, by putting together (6.8) and (6.7) we get

$$|f(x_j) - \Psi_j(Tf)| \leq \dots \leq Cm^{1/2-s}\|\tilde{f}\|_{H^s(\Omega)} \leq \tilde{C}m^{1/2-s}\|f\|_{H^s(\Omega)}.$$

□

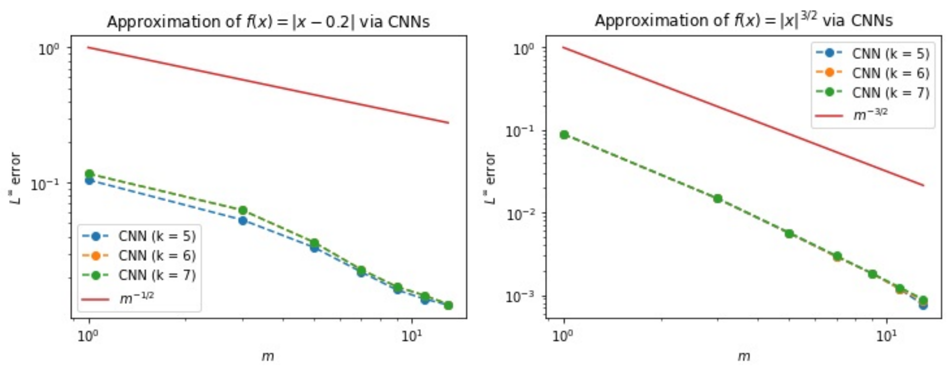


Figure 6.7: Numerical validation of the upper bounds in Lemma 6.4. The two panels show the results obtained for signals of different smoothness, respectively $H^1(0,1)$ on the left and $H^2(0,1)$ on the right. The results are reported for different grid resolutions, the mesh stepsize being $h = 2^{-k}$.

We remark that, as for the results in the previous Section, the proof is constructive. The pictures in Figure 6.7 show the approximation rates obtained by the actual implementation of Ψ (and T) along the lines detailed in the proof. Note that we do not train the network, as we directly initialize Ψ with the wished weights and biases. The left panel in Figure 6.7 shows the results obtained for a mildly smooth signal, $f(x) = |x - 1/5|$. In this case we have $s = 1$, and the L^∞ error between the desired output, f, and the CNN approximation, is shown to decay at the expected rate, that is $1/\sqrt{m}$. This is also true regardless of the grid resolution, coherently with Lemma 6.4. Indeed, we obtained nearly the same results for $N_h = 33, 65, 129$. Finally, the right panel in Figure 6.7 refers to a smoother case, $f(x) = |x|^{3/2}$, where $s = 2$. Here we can remark, once again, the expected behavior.

6.4.3 Proof of Theorem 6.1

We are now ready to prove Theorem 6.1. For better readability, we report both the Theorem and its proof below.

Theorem 6.1. *Let $\Omega := (0,1)$ and let $\{x_j\}_{j=1}^{N_h} \subset \Omega$ be a uniform grid with stepsize $h = 2^{-k}$. Let $\Theta \subset \mathbb{R}^p \ni \boldsymbol{\mu} \to u_{\boldsymbol{\mu}} \in H^s(\Omega)$ be a (nonlinear) operator, where Θ is a compact domain and $s \geq 1$. For some $r \geq 1$, assume that the operator is r-times continuously Fréchet differentiable. For any $0 < \varepsilon < 1/2$, there exists a deep neural network $\Phi : \mathbb{R}^p \to \mathbb{R}^{N_h}$ such that*

$$|u_{\boldsymbol{\mu}}(x_j) - \Phi_j(\boldsymbol{\mu})| < \varepsilon$$

uniformly for all $\boldsymbol{\mu} \in \Theta$ and all $j = 1, \dots, N_h$. Additionally, Φ can be defined as the composition of a fully connected network ϕ and a convolutional neural network Ψ, i.e. $\Phi = \Psi \circ \phi$, such that the overall architecture has at most

i) $C\log(1/\varepsilon)$ dense layers, with ReLU activation, and $C\log(1/h)$ convolutional layers,

ii) $C\varepsilon^{-2/(2s-1)}\left[\varepsilon^{-p/r}\log(1/\varepsilon) + \log(1/h)\right]$ active weights,

iii) $C\varepsilon^{-2/(2s-1)}$ channels in input and output,

where $C > 0$ is some constant dependent on Θ and on the operator $\boldsymbol{\mu} \to u_{\boldsymbol{\mu}}$, thus also on s, r, p.

Proof. Let $\varepsilon > 0$ and let $c = c(s) > 0$ be the constant in Theorem 6.4. We take advantage of the compactness of Θ and the continuity of the operator to define

$$M := \max_{\boldsymbol{\mu} \in \Theta} \|u_{\boldsymbol{\mu}}\|_{H^s(\Omega)} < +\infty.$$

Let now Ψ be the CNN in Lemma 6.4, where we fix $m = \lceil (\varepsilon/2)^{-2/(2s-1)} Mc \rceil$. Then,

$$|u_{\boldsymbol{\mu}}(x_j) - \Psi_j(Tu_{\boldsymbol{\mu}})| < \varepsilon/2 \tag{6.9}$$

for all $j = 1, \dots, N_h$ and $\boldsymbol{\mu} \in \Theta$, where $T : H^s(\Omega) \to \mathbb{C}^{2m+1} \cong \mathbb{R}^{4m+2}$ is some continuous linear operator. We now note that, by composition, the map

$$\boldsymbol{\mu} \to Tu_{\boldsymbol{\mu}}$$

is an element of the Sobolev space $W^{r,\infty}(\Theta; \mathbb{R}^{4m+1})$. In particular, by Theorem 1 in [202], there exists a ReLU DNN $\phi : \mathbb{R}^p \to \mathbb{R}^{4m+2}$ with $C \log(m/\varepsilon)$ hidden layers and $Cm\varepsilon^{-p/r} \log(m/\varepsilon)$ active weights, such that

$$\sup_{\boldsymbol{\mu} \in \Theta} \|Tu_{\boldsymbol{\mu}} - \phi(\boldsymbol{\mu})\|_1 < \varepsilon/2, \tag{6.10}$$

where $\|\cdot\|_1$ is the ℓ_1 norm over $\mathbb{R}^{4m+2}$, while $C > 0$ is a constant that depends on r, p, Θ, s and the operator $\boldsymbol{\mu} \to u_{\boldsymbol{\mu}}$. The dependence on s comes from the Lipschitz constant of T, which may inflate the magnitude of the partial derivatives of $\boldsymbol{\mu} \to Tu_{\boldsymbol{\mu}}$.

Let now consider the composition $\Phi := \Psi \circ \phi$. It is easy to see that, up to replacing the value of the constant C, this DNN architecture satisfies the requirements claimed in the Theorem. In fact, we note that, since $0 < \varepsilon < 1/2$,

$$\log(m/\varepsilon) = \log(m) + \log(1/\varepsilon) = C' \log(1/\varepsilon) + C'' \log(2^{2/(2s-1)} Mc) \le C''' \log(1/\varepsilon).$$

Furthermore, for any $\boldsymbol{\mu} \in \Theta$ and $j = 1, \dots, N_h$ we have the desired bound. In fact, by (6.9),

$$\begin{aligned} |u_{\boldsymbol{\mu}}(x_j) - \Phi_j(\boldsymbol{\mu})| \le& \\ &\le |u_{\boldsymbol{\mu}}(x_j) - \Psi_j(Tu_{\boldsymbol{\mu}})| + |\Phi_j(\boldsymbol{\mu}) - \Psi_j(Tu_{\boldsymbol{\mu}})| < \\ &< \frac{\varepsilon}{2} + |\Psi_j(\phi(\boldsymbol{\mu})) - \Psi_j(Tu_{\boldsymbol{\mu}})|. \end{aligned} \tag{6.11}$$

Now, we note that $|\Psi_j(\mathbf{a}) - \Psi_j(\mathbf{b})| \le \|\mathbf{a} - \mathbf{b}\|_1$. In fact, in Lemma 6.4, Ψ was defined as $E \circ R \circ \mathcal{S}_m$, where E and R were reshape layers, while $\mathcal{S}_m$ was as in Lemma 6.3. In particular,

$$|\Psi_j(\mathbf{a}) - \Psi_j(\mathbf{b})| \le \sup_{x \in [0,1]} \left| \sum_{k=-m}^{m} a_k e^{2\pi \mathrm{i} kx} - \sum_{k=-m}^{m} b_k e^{2\pi \mathrm{i} kx} \right| \le \sum_{k=-m}^{m} |a_k - b_k|.$$

The above, together with (6.10), finally allows us to continue (6.11) as

$$|u_{\boldsymbol{\mu}}(x_j) - \Phi_j(\boldsymbol{\mu})| \le \dots < \frac{\varepsilon}{2} + \|\phi(\boldsymbol{\mu}) - Tu_{\boldsymbol{\mu}}\|_1 < \varepsilon.$$

□

Remark. The hypothesis of Frechét differentiability in Theorem 6.1 can be relaxed. Indeed, for the error bounds to hold, it is sufficient that the map $\boldsymbol{\mu} \to Tu_{\boldsymbol{\mu}}$ is in the Sobolev space $W^{r,+\infty}(\Theta, \mathbb{R}^{4m+1})$. For instance, one may require the operator $\boldsymbol{\mu} \to u_{\boldsymbol{\mu}}$ to be r-times Frechét differentiable, with the first $r-1$ derivatives being continuous and the last one being (essentially) bounded.

7 | Beyond convolutions: Mesh-Informed Neural Networks

In this seventh Chapter, we present a novel class of neural networks architectures, called Mesh-Informed Neural Networks, an original idea developed during my PhD. These models enable in the most natural way the development of Deep Learning based ROMs for problems that feature general geometries, overcoming the limitations posed by convolutional architectures. After the presentation of the approach and its numerical validation, we conclude the Chapter with a connection to oxygen transfer models, eventually explaining in full detail the construction of the surrogate model employed back in Section 4.3.

In Chapter 5, we developed a Deep Learning approach for Reduced Order Modeling of parametrized PDEs, DL-ROM, discussing its potential applications to the study of oxygen transfer models. However, the proposed methodology is mostly grounded on the use of convolutional autoencoders, which may limit the applicability of DL-ROMs to certain geometries. Our purpose for this Chapter is to overcome this limitation by developing and validating a novel class of neural network architectures that can handle general non-convex domains. We call them *Mesh-Informed Neural Networks* (MINNs).

The rest of the Chapter is organized as follows. In Section 7.1, we set some notation and formally introduce our new class of architectures from a theoretical point of view. There, we also discuss their practical implementation and comment on the parallelism between MINNs and other emerging approaches such as Graph Neural Networks. Then, in Section 7.2, we assess the approximation power of MINNs in the context of Operator Learning. To do this, we compare the use of MINNs with that of classical fully connected architectures, testing their performance over a broad set of numerical experiments, with applications to nonlinear operators and nonlinear PDEs, and ultimately validating their scientific value. Finally, in Section 7.3, we revert our attention back to the case of oxygen microcirculation in biological tissues, reconnecting to the example treated at the end of Chapter 4.

7.1 Mesh-informed architectures

As we extensively discussed in Chapters 5 and 6, Deep Neural Networks (DNNs) have recently become a valuable tool for learning nonlinear operators in high dimensional spaces. In particular, architectures such as Convolutional Neural Networks (CNNs) have shown a remarkable power when addressing this kind of problems. When the operator under study concerns functions defined over discrete hypercubes, CNNs are a natural alternative to classical fully connected architectures, as they can achieve the same expressivity with far less learnable parameters (i.e., weights and biases). However, CNNs cannot be applied directly to data defined over general domains, which constitutes a major limitation for PDE applications. So how should we extend this idea for it to work?

Clearly, this is not a new question: many researchers have been working on this problem, and new alternative architectures, such as Graph Neural Networks [172], have been developed. However, most of these approaches were motivated by applications unrelated to operator learning. In particular, within this context, more natural and efficient constructions may exist. Here, we shall present our own proposal, a novel type of architecture called Mesh-Informed Neural Network.

To derive its construction, let us reconsider once again CNNs and their successes. Among all the properties that characterize CNNs, one fundamental feature is arguably their capacity to operate at different resolutions. To achieve data compression, CNNs send their input through hidden states of decreasing resolution; conversely, when reconstructing functional signals, CNN models start from data at low resolution and proceed with multiple subsequent refinements. We have also seen throughout the previous Chapters, both in practical experiments, as in Chapter 5, and in theoretical proofs, as in Chapter 6.

When considering the numerical resolution of PDE models, we find a similar paradigm in the so-called multigrid methods, where multiple meshes are used to solve different scales of the same spatial domain. Inspired by this analogy, we propose a new type of architecture that can handle general non-convex domains by operating at different resolution levels. To do so, we shall equip the components of the architecture with a geometrical meaning, turning, for instance, classical neurons into mesh vertices. We detail our idea below, in a much more rigorous way.

We are given a bounded domain $\Omega \subset \mathbb{R}^d$, not necessarily convex, and two meshes having respectively stepsizes $h, h' > 0$ and vertices

$$\{\mathbf{x}_j\}_{j=1}^{N_h},\ \{\mathbf{x}'_i\}_{i=1}^{N_{h'}} \subset \overline{\Omega}.$$

The two meshes can be completely different and they can be either structured or unstructured. To each mesh we associate the corresponding space of piecewise-linear Lagrange polynomials, namely $V_h, V_{h'} \subset L^2(\Omega)$. Our purpose is to introduce a suitable notion of *mesh-informed layer* $L : V_h \to V_{h'}$ that exploits the apriori existence of Ω. In analogy to the case of dense layers, see Definition 0.1, L should have N_h neurons at input and $N_{h'}$ neurons at output, since $V_h \cong \mathbb{R}^{N_h}$ and $V_{h'} \cong \mathbb{R}^{N_{h'}}$. However, as we already discussed in Chapter 6, thinking of the state spaces as either comprised of functions or vectors is fundamentally different: while we can describe the objects in V_h as regular, smooth or noisy, these notions have no meaning in $\mathbb{R}^{N_h}$, and similarly for $V_{h'}$ and $\mathbb{R}^{N_{h'}}$. Furthermore, in the case of PDE applications, we are typically not interested in all the elements of V_h and $V_{h'}$, rather we focus on those that present spatial correlations coherent with the underlying physics. Starting from these considerations, we build a novel layer architecture that can meet our specific needs. In order to provide a rigorous definition, and directly extend the idea to higher order finite element spaces, we first introduce some preliminary notation. For the sake of simplicity, we will restrict to simplicial finite elements.

Definition 7.1. (Admissible mesh) *Let $\Omega \subset \mathbb{R}^d$ be a bounded domain. Let $\mathcal{M} = \{K_i\}_{i\in\mathcal{I}}$ be a collection of d-simplices in Ω, that is $K \subset \overline{\Omega}$ for each $K \in \mathcal{M}$. For each element $K \in \mathcal{M}$, define the quantities*

$$h_K := \operatorname{diam}(K), \qquad R_K := \sup\left\{\operatorname{diam}(S) \mid S \text{ is a ball contained in } K\right\},$$

where $\operatorname{diam}(\cdot)$ *is the set diameter. We say that $\mathcal{M}$ is an admissible mesh of stepsize $h > 0$ over Ω if the following conditions hold.*

1. *The elements are exhaustive, that is*

$$\operatorname{dist}\left(\overline{\Omega}, \bigcup_{K\in\mathcal{M}} K\right) \le h$$

where $\operatorname{dist}(A,B) = \sup_{x\in A}\inf_{y\in B}|x-y|$ *is the distance between A and B.*

2. *Any two distinct elements $K, K' \in \mathcal{M}$ have disjoint interiors. Also, their intersection is either empty or results in a common face of dimension $s < d$.*

4. *The elements are non degenerate and their maximum diameter equals h, i.e.*

$$\min_{K\in\mathcal{M}} R_K > 0 \quad \text{and} \quad \max_{K\in\mathcal{M}} h_K = h.$$

In that case, the quantity

$$\sigma = \min_{K\in\mathcal{M}} \frac{h_K}{R_K} < +\infty,$$

is said to be the aspect-ratio of the mesh.

Definition 7.2. (Function to node operator) *Let $\Omega \subset \mathbb{R}^d$ be a bounded domain and let $\mathcal{M} = \{K_i\}_{i\in\mathcal{I}}$ be a mesh of stepsize $h > 0$ defined over Ω. For any positive integer q, we write $X_h^q(\mathcal{M})$ for the finite element space of piecewise-polynomials of degree at most q, that is*

$$X_h^q(\mathcal{M}) := \{v \in \mathcal{C}(\overline{\Omega}) \text{ s.t. } v_{|K} \text{ is a polynomial of degree at most } q \;\; \forall K \in \mathcal{M}\}.$$

Let $N_h = \dim(X_h^q(\mathcal{M}))$. We say that a collection of nodes $\{\mathbf{x}_i\}_{i=1}^{N_h} \subset \Omega$ and a sequence of functions $\{\varphi_i\}_{i=1}^{N_h} \subset X_h^q(\mathcal{M})$ define a Lagrangian basis of $X_h^q(\mathcal{M})$ if

$$\varphi_j(\mathbf{x}_i) = \delta_{i,j} \qquad i, j = 1, \dots, N_h.$$

We write $\Pi_{h,q}(\mathcal{M}) : X_h^q(\mathcal{M}) \to \mathbb{R}^{N_h}$ for the function-to-nodes operator,

$$\Pi_{h,q}(\mathcal{M}) : \;\; v \to [v(\mathbf{x}_1), \dots, v(\mathbf{x}_{N_h})],$$

whose inverse is

$$\Pi_{h,q}^{-1}(\mathcal{M}) : \;\; \mathbf{c} \to \sum_{i=1}^{N_h} c_i \varphi_i.$$

We now have all we need to introduce our concept of mesh-informed layer.

Definition 7.3. (Mesh-informed layer) *Let $\Omega \subset \mathbb{R}^d$ be a bounded domain and $d : \Omega \times \Omega \to [0, +\infty)$ a given distance function. Let $\mathcal{M}$ and $\mathcal{M}'$ be two meshes of stepsizes h and h', respectively. Let $V_h = X_h^q(\mathcal{M})$ and $V_{h'} = X_{h'}^{q'}(\mathcal{M})$ be the input and output spaces, respectively. Denote by $\{\mathbf{x}_j\}_{j=1}^{N_h}$ and $\{\mathbf{x}'_i\}_{i=1}^{N_{h'}}$ the nodes associated to a Lagrangian basis of V_h and $V_{h'}$ respectively. A mesh-informed layer with activation function $\rho : \mathbb{R} \to \mathbb{R}$ and support $r > 0$ is a map $L : V_h \to V_{h'}$ of the form*

$$L = \Pi_{h',q'}^{-1}(\mathcal{M}') \circ \tilde{L} \circ \Pi_{h,q}(\mathcal{M})$$

where $\tilde{L} : \mathbb{R}^{N_h} \to \mathbb{R}^{N'_h}$ is a layer with activation ρ whose weight matrix $\mathbf{W}$ satisfies the additional sparsity constraint below,

$$d(\mathbf{x}_j, \mathbf{x}'_i) > r \implies \mathbf{W}_{i,j} = 0.$$

The distance function d in Definition 7.3 can be any metric over Ω. For instance, one may choose to consider the Euclidean distance, $d(\mathbf{x}, \mathbf{x}') := |\mathbf{x} - \mathbf{x}'|$. However, if the geometry of Ω becomes particularly involved, better choices of d might be available, such as the *geodesic distance* . The latter quantifies the distance of two points $\mathbf{x}, \mathbf{x}' \in \Omega$ by measuring the length of the shortest path within Ω that starts at $\mathbf{x}$ and ends at $\mathbf{x}'$, namely

$$d(\mathbf{x}, \mathbf{x}') := \inf \left\{ \int_0^1 |\gamma'(t)| dt, \text{ with } \gamma \in \mathcal{C}([0,1], \mathbb{R}^d),\; \gamma([0,1]) \subseteq \Omega, \right. \\ \left. \gamma(0) = \mathbf{x},\; \gamma(1) = \mathbf{x}' \right\}$$

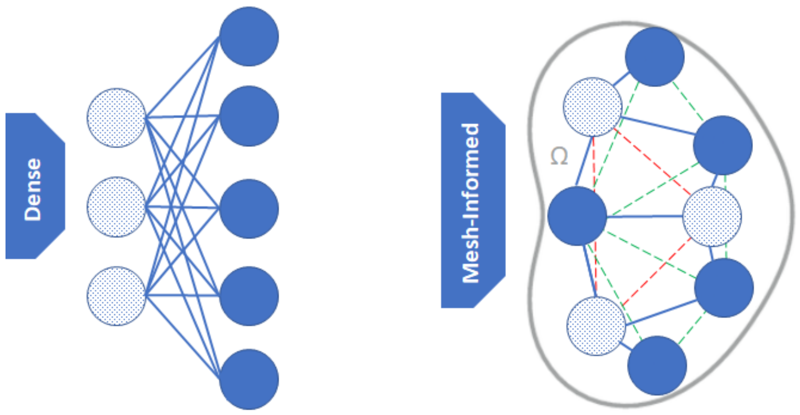


Figure 7.1: Comparison of a dense layer (see the preliminary Chapter, Definition 0.1) and a mesh-informed layer (cf. Definition 7.3). The dense model features 3 neurons at input and 5 at output. All the neurons communicate: consequently the weight matrix of the layer has 15 active entries. In the mesh-informed counterpart, neurons become vertices of two meshes (resp. in green and red) defined over the same spatial domain Ω. Only nearby neurons are allowed to communicate. This results in a sparse model with only 9 active weights.

It is worth pointing out that, as a matter of fact, the projections $\Pi_{h,q}(\mathcal{M})$ and $\Pi^{-1}_{h',q'}(\mathcal{M}')$ have the sole purpose of making Definition 7.3 rigorous. What actually defines the mesh-informed layer L are the sparsity patterns imposed to $\tilde{L}$. In fact, the idea is that these constraints should help the layer in producing outputs that are more coherent with the underlying spatial domain (cf. Figure 7.1). In light of this intrinsic duality between L and $\tilde{L}$, we will refer to the weights and biases of L as to those that are formally defined in $\tilde{L}$. Also, for better readability, from now on we will use the notation

$$L : V_h \xrightarrow{r} V_{h'},$$

to emphasize that L is a mesh-informed layer with support r. We note that dense layers can be recovered by letting $r \ge \sup\{d(\mathbf{x},\mathbf{x}') \mid \mathbf{x},\mathbf{x}' \in \Omega\}$, while lighter architecture are obtained for smaller values of r. The following result provides an explicit upper bound on the number of nonzero entries in a mesh-informed layer. For the sake of simplicity, we restrict to the case in which d is the Euclidean distance.

Theorem 7.1. *Let $\Omega \subset \mathbb{R}^d$ be a bounded domain and d the Euclidean distance. Let $\mathcal{M}$ and $\mathcal{M}'$ be two meshes having respectively stepsizes h, h' and aspect-ratios σ, σ'. Let*

$$h_{\min} := \min_{K\in\mathcal{M}} h_K, \quad h'_{\min} := \min_{K'\in\mathcal{M}'} h_{K'}$$

be the smallest diameters within the two meshes respectively. Let $L : V_h \xrightarrow{r} V_{h'}$ be a mesh-informed layer of support $r > 0$, where $V_h := X^q_h(\mathcal{M})$ and $V_{h'} := X^{q'}_{h'}(\mathcal{M}')$. Then,

$$\|\mathbf{W}\|_0 \le C\left(\sigma\sigma'\frac{r}{h_{\min}h'_{\min}}\right)^d$$

where $\|\mathbf{W}\|_0$ is the number of nonzero entries in the weight matrix of the layer L, while $C = C(\Omega, d, q, q') > 0$ is a constant depending only on Ω, d, q and q'.

Proof. Let $N_h := \dim(V_h)$, $N_{h'} := \dim(V_{h'})$ and let $\{\mathbf{x}_j\}_{j=1}^{N_h}, \{\mathbf{x}'_i\}_{i=1}^{N_{h'}}$ be the Lagrangian nodes in the two meshes respectively. Let $\omega := |B(\mathbf{0},1)|$ be the volume of the unit ball in $\mathbb{R}^d$. Since $\min_{K'} R_{K'} \ge h'_{\min}/\sigma'$, the volume of an element in the output mesh is at least

$$v_{\min}(h') := (h'_{\min}/\sigma')^d\omega.$$

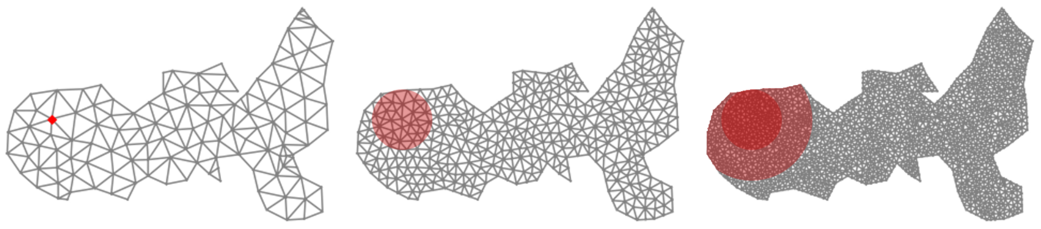


Figure 7.2: MINNs operate at different resolution levels to enforce local properties. Here, three meshes of the Elba island, in Italy, represent three different hidden states in the pipeline of a suitable MINN architecture, where neurons are identified with mesh vertices. Due to the sparsity constraint (Definition 7.3), the red neuron only fires information to the nearby neurons within the circle (second mesh). In turn, these can only communicate with the neurons in the third mesh that are sufficiently close (largest red circle). Even though the supports are limited, the composition of mesh-informed layers eventually lets information spread all over the domain.

It follows that, for any $\mathbf{x} \in \Omega$, the ball $B(\mathbf{x}, r)$ can contain at most

$$n_{\mathrm{e}}(r, h') := \frac{\omega r^d}{v_{\min}(h')} = \left(\frac{\sigma' r}{h'_{\min}}\right)^d$$

elements of the output mesh. Therefore, the number of indices i such that $|\mathbf{x}'_i - \mathbf{x}_j| \le r$ is at most $n_{\mathrm{e}}(r,h')c(d,q')$, where $c(d,q') := (d+q')!/(q'!d!)$ bounds the number of degrees of freedom within each element. Finally,

$$\|\mathbf{W}\|_0 \le N_h n_{\mathrm{e}}(r,h')c(d,q') \le$$

$$\le \frac{c(d,q)|\Omega|}{v_{\min}(h)} n_{\mathrm{e}}(r,h')c(d,q') = \frac{c(d,q)c(d,q')|\Omega|}{\omega} \cdot \left(\frac{\sigma\sigma' r}{h_{\min} h'_{\min}}\right)^d$$

□

Starting from here, we define MINNs by composition, with a possible interchange of dense and mesh-informed layers. Consider for instance the case in which we want to define a Mesh-Informed Neural Network $\Phi : \mathbb{R}^p \to V_h \cong \mathbb{R}^{N_h}$ that maps a low-dimensional input, say $p \ll N_h$, to some functional output. Then, using our notation, one possible architecture could be

$$\begin{tikzcd} \Phi : \mathbb{R}^p \arrow[r] & V_{4h} \arrow[r, "r = 0.5"] \arrow[d, phantom, "\cong"] & V_{2h} \arrow[r, "r = 0.25"] \arrow[d, phantom, "\cong"] & V_h \arrow[d, phantom, "\cong"] \\ & \mathbb{R}^{N_{4h}} & \mathbb{R}^{N_{2h}} & \mathbb{R}^{N_h} \end{tikzcd} \quad , \tag{7.1}$$

The above scheme defines a DNN of depth $l = 2$, as it is composed of 3 layers. The first layer is dense (Definition 0.1) and has the purpose of preprocessing the input while mapping the data onto a coarse mesh (stepsize $4h$). Then, the remaining two layers perform local transformations in order to return the desired output. Note that the three meshes need not to satisfy any hierarchy, see e.g. Figure 7.2. Also, the corresponding finite element spaces need not to share the same polynomial degree. Clearly, (7.1) can be generalized by employing any number of layers, as well as any sequence of stepsizes $h_1, \dots, h_n$ and supports $r_1, \dots, r_{n-1}$. The choice of the hyperparameters remains problem specific, but a good rule of thumb is to decrease the supports as the mesh becomes finer, so that the network complexity is kept under control (cf. Theorem 7.1).

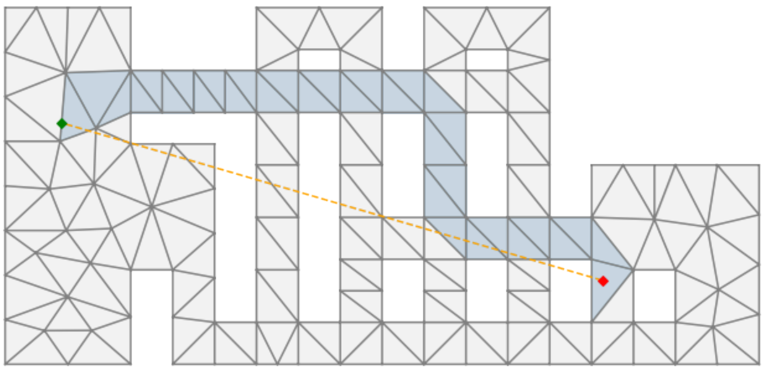


Figure 7.3: Approximation of the geodesic distance $\mathscr{d}(\mathbf{x},\mathbf{y})$ as the shortest path across elements (blue triangles). In general, the latter is different from the Euclidean distance between the two points (orange dashed line).

7.1.1 Implementation details

For simplicity, let us first focus on the case in which distances are evaluated according to the Euclidean metric, $\mathscr{d}(\mathbf{x},\mathbf{x}') = |\mathbf{x}-\mathbf{x}'|$. In this case, the practical implementation of mesh-informed layers is straightforward and can be done as follows. Given $\Omega \subset \mathbb{R}^d$, $h, h' > 0$, let $\mathbf{X} \in \mathbb{R}^{N_h \times d}$ and $\mathbf{X}' \in \mathbb{R}^{N_{h'} \times d}$ be the matrices containing the degrees of freedom associated to the chosen finite element spaces, that is $\mathbf{X}_{j,\cdot} := [X_{j,1},\dots,X_{j,d}]$ are the coordinates of the jth node in the input mesh, and similarly for $\mathbf{X}'$. In order to build a mesh-informed layer of support $r > 0$, we first compute all the pairwise distances $D_{i,j} := |\mathbf{X}_{j,\cdot} - \mathbf{X}'_{i,\cdot}|^2$ among the nodes in the two meshes. This can be done efficiently using tensor algebra, e.g.

$$D = \sum_{l=1}^{d} (e_{N_{h'}} \otimes \mathbf{X}_{\cdot,l} - \mathbf{X}'_{\cdot,l} \otimes e_{N_h})^{\circ 2}$$

where $\mathbf{X}_{\cdot,l}$ is the lth column of $\mathbf{X}$, $e_k := [1,\dots,1] \in \mathbb{R}^k$, $\otimes$ is the tensor product and $\circ 2$ is the Hadamard power. We then extract the indices $\{(i_k, j_k)\}_{k=1}^{\text{dof}}$ for which $D_{i_k,j_k} \le r^2$ and initialize a weight vector $\mathbf{w} \in \mathbb{R}^{\text{dof}}$. This allows us to declare the weight matrix $\mathbf{W}$ in sparse format by letting the nonzero entries be equal to w_k at position (i_k, j_k), so that $\|\mathbf{W}\|_0 = \text{dof}$. Preliminary to the training phase, we fill the values of $\mathbf{w}$ by considering an adaptation of the so-called He initialization. More precisely, we sample independently the values $w_1,\dots,w_{\text{dof}}$ from a normal distribution having mean zero and variance $1/\|\mathbf{W}\|_0$.

The above reasoning can be easily adapted to the general case, provided that one is able to compute efficiently all the pairwise distances $\mathscr{d}(\mathbf{X}_{j,\cdot}, \mathbf{X}'_{i,\cdot})$. Of course, the actual implementation will then depend on the specific choice of $\mathscr{d}$. Since the case of geodesic distances can be of particular interest in certain applications, we shall briefly discuss it below. In this setting, the main difficulty arises from the fact that, in general, we are required to computed distances between points of different meshes. Additionally, if we consider Finite Element spaces of degree $q > 1$, not all the Lagrangian nodes will be placed over the mesh vertices, meaning that we cannot exploit the graph structure of the mesh to compute shortest paths.

To overcome these drawbacks, we propose the introduction of an auxiliary coarse mesh $\mathcal{M}_0 := \{K_i\}_{i=1}^m$, whose sole purpose is to capture the geometry of the domain. We use this mesh to build another graph, $\mathscr{G}$, which describes the location of the elements K_i. More precisely, let $\mathbf{c}_i$ be the centroid of the element K_i. We let $\mathscr{G}$ be the weighted graph having vertices $\{\mathbf{c}_i\}_{i=1}^m$, where we link $\mathbf{c}_i$ with $\mathbf{c}_j$ if and only if the elements K_i and K_j are adjacent. Then, to weight the edges, we use the Euclidean distance between the centroids. Once $\mathscr{G}$ is constructed, we use Dijkstra's algorithm to compute all the shortest paths along the graph. This leaves us with an estimated geodesic distance $g_{i,j}$ for each pair of centroids $(\mathbf{c}_i, \mathbf{c}_j)$, which we can precompute and store in a suitable matrix. Then, we approximate the geodesic distance of *any* two points $\mathbf{x}, \mathbf{x}' \in \Omega$ as

$$d(\mathbf{x}, \mathbf{x}') \approx g_{i(\mathbf{x}), i(\mathbf{x}')},$$

where $i : \Omega \to \{1, \dots, m\}$ maps each point to one of the elements that contains that point (see Figure 7.3). In other words,

$$i(\mathbf{x}) := \min\{i \mid \mathbf{x} \in K_i\}.$$

If the auxiliary mesh does not fill completely Ω, then one may also use the relaxed version below

$$i(\mathbf{x}) := \min\{i \mid \text{dist}(\mathbf{x}, K_i) = \min_{j=1,\dots,m} \text{dist}(\mathbf{x}, K_j)\}.$$

In general, evaluating the index function i can be done in $\mathcal{O}(m)$ time. In particular, going back to our original problem, we can approximate all the pairwise distances between the nodes of the input and the output spaces in $\mathcal{O}(mN_h + mN_n' + m^2)$ time. In fact, we can run Dijkstra's algorithm once and for all with a computational cost of $\mathcal{O}(m^2)$. Then, we just need to evaluate the index function i for all the Lagrangian nodes $\{\mathbf{x}\}_{i=1}^{N_h}$ and $\{\mathbf{x}'\}_{i=1}^{N_{h'}}$, which takes respectively $\mathcal{O}(mN_h)$ and $\mathcal{O}(mN_{h'})$. In particular, the total computational cost for building the mesh-informed layer can be substantially smaller than the one obtained with a brute force approach, which would be $\mathcal{O}(N_h N_{h'})$.

7.1.2 How MINNs relate to existing literature

It is worth to comment on the differences and similarities that MINNs share with other Deep Learning approaches. We discuss them below.

Relationship to CNNs and GNNs

Mesh-Informed architectures can work at different resolutions simultaneously, in a way that is very similar to CNNs. However, their construction comes with multiple advantages. First of all, Definition 7.3 adapts to any geometry, while convolutional layers typically operate on square or cubic input-output. Furthermore, convolutional layers use weight sharing, meaning that all parts of the domain are treated in the same way. This may not be an optimal choice in some applications, such as those involving PDEs, as we may want to differentiate our behavior over Ω (for instance near of far away from the boundaries).

Conversely, MINNs share with GNNs the ability to handle general geometries. As a matter of fact, we mention that these architectures have been recently applied to mesh-based data, see e.g. [62, 76, 158, 201]. With respect to GNNs, the main advantage of MINNs lies in their capacity to work at different fidelity levels. This fact, which essentially comes from the presence of an underlying spatial domain, has at least two advantages: it favors resolution independence and it increases the interpretability of hidden layers (now the number of neurons is not arbitrary but comes from the chosen discretization). In this sense, MINNs exploit meshing strategies as auxiliary tools and they appear to be a natural choice for learning discretized functional outputs.

Relationship to DeepONets and Neural Operators

Recently, some new DNN models have been proposed for operator learning. One of these are DeepONets [126], a mesh-free approach that is based on an explicit decoupling between the input and the space variable. More precisely, DeepONets consider a representation of the following form

$$(\mathcal{G}_h f)(\mathbf{x}) \approx \Psi(f) \cdot \phi(\mathbf{x}),$$

where $\cdot$ is the dot product, $\Psi : V_h \to \mathbb{R}^m$ is the so-called trunk-net, and $\phi : \Omega \to \mathbb{R}^m$ is the branch-net. DeepONets have been shown capable of learning nonlinear operators and are now being extended to include apriori physical knowledge, see e.g. [196]. We consider MINNs and DeepONets as two fundamentally different approaches that answer different questions. DeepONets were originally designed to process input data coming from sensors and, being mesh-free, they are particularly suited for those applications where the output is only partially known or observed. In contrast, MINNs are rooted on the presence of a high fidelity model $\mathcal{G}_h$ and are thus better suited for tasks such as reduced order modeling. Another difference lies in the fact that DeepONets include explicitly the dependence on the space variable $\mathbf{x}$. This can then make it harder to process general domains and include additional information such as boundary data. Conversely, MINNs can easily handle this kind of issues thanks to their global perspective.

In this sense, MINNs are much closer to the so-called Neural Operators, a novel class of DNN models recently proposed by Kovachki et al. [109]. Neural Operators are an extension of classical DNNs that was developed to operate between infinite dimensional spaces. These models are ultimately based on Hilbert-Schmidt operators, meaning that their linear part, that is ignoring activations and biases, is of the form

$$W : f \to \int_{\Omega} k(\cdot, \mathbf{x}) f(\mathbf{x}) d\mathbf{x} \tag{7.2}$$

where $k : \Omega \times \Omega \to \mathbb{R}$ is some *kernel* function that has to be identified during the training phase. Clearly, the actual implementation of Neural Operators is carried out in a discrete setting and integrals are replaced with suitable quadrature formulas.

The construction of Neural Operators is of full generality, to the point that other approaches such as DeepONets [126] can be seen as a special case. Similarly, we note that MINNs also form a special class of Neural Operators. Indeed, a rough Monte Carlo-type estimate of (7.2) would yield

$$(Lf)(\mathbf{x}_i') = \int_{\Omega} k(\mathbf{x}_i', \mathbf{x}) f(\mathbf{x}) d\mathbf{x} \approx \frac{|\Omega|}{N_h} \sum_j k(\mathbf{x}_i', \mathbf{x}_j) f(\mathbf{x}_j).$$

If we let $\{\mathbf{x}_j\}_j$ and $\{\mathbf{x}_i'\}_i$ be the nodes in the two meshes, then the constraint in Definition 7.3 becomes equivalent to the requirement that k is supported somewhere near the diagonal, that is $\text{supp}(k) \subseteq \{(\mathbf{x}, \mathbf{x} + \varepsilon)$ with $|\varepsilon| \le r\}$.

We believe that these parallels are extremely valuable, as they indicate the existence of a growing scientific community with common goals and interests.

7.2 Numerical experiments

We provide empirical evidence that the sparsity introduced by MINNs can be of great help in learning maps that involve functional data, such as nonlinear operators, showing a reduced computational cost and better generalization capabilities. We first detail the setting of each experiment alone, specifying the corresponding operator to be learned and the adopted MINN architecture. Then, at the end of the current Section, we discuss the numerical results.

Throughout all our experiments, we adopt a standardized approach for designing and training the networks. In general, we always employ the 0.1-leakyReLU activation for all the hidden layers, while we do not use any activation at the output. To build mesh-informed layers we adopt the Euclidean distance as a metric function d. Additionally, every time a mesh-informed architecture is introduced, we also consider its dense counterpart, obtained without imposing the sparsity constraints. Both networks are then trained following the same criteria, so that a fair comparison can be made. As loss function, we always consider the mean square error computed with respect to the L^2 norm, that is

$$\mathbb{E}_{f \sim \mathbb{P}} \|\mathcal{G}_h f - \Phi(f)\|^2_{L^2(\Omega)}$$

where $\mathcal{G}_h$ is the (discretized) operator to be learned and Φ is the DNN model. Here, $\mathbb{P}$ is some given probability distribution over the input space Θ, which we allow to be either finite or infinite dimensional.

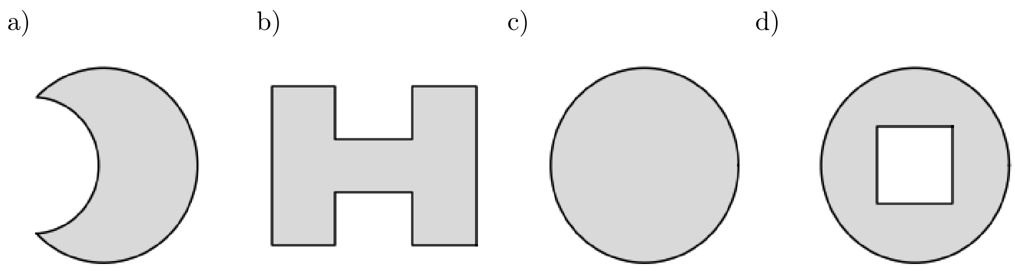


Figure 7.4: Domains considered for the numerical experiments in Section 7.2. Panel a): upto boundaries, Ω is the difference of two circles, $B(\mathbf{0},1)$ and $B(\mathbf{x}_0,0.7)$, where $\mathbf{x}_0=(-0.75,0)$. Panel b): A polygonal domain obtained by removing the rectangles $[-0.75,0.75]\times[0.5,1.5]$ and $[-0.75,0.75]\times[-1.5,-0.5]$ from $(-2,2)\times(-1.5,1.5)$. Panel c): the unit circle $B(\mathbf{0},1)$. Panel d): Ω is obtained by removing a square, namely $[-0.4,0.4]^2$, from the unit circle $B(\mathbf{0},1)$.

The optimization of the loss function is performed via the L-BFGS optimizer, with learning rate always equal to 1 and no batching. What may change from case to case are the network architecture, the number of epochs, and the size of the training set. After training, we compare mesh-informed and dense architectures by evaluating their performance on 500 unseen samples (test set), which we use to compute an unbiased estimate of the relative error below,

$$\mathbb{E}_{f\sim\mathbb{P}}\left[\frac{\|\mathcal{G}_h f-\Phi(f)\|_{L^2(\Omega)}}{\|\mathcal{G}_h f\|_{L^2(\Omega)}}\right]. \tag{7.3}$$

All the code was written in Python 3, mainly relying on Pytorch for the implementation of DNNs. Instead, we employed the FEniCS library for defining meshes and, when needed, solving PDEs.

7.2.1 Description of the benchmark operators

Learning a parametrized family of functions

Let Ω be the domain defined as in Figure 7.4a. For our first experiment, we consider a variation of a classical problem concerning the calculation of the so-called *signed distance function* of Ω. This kind of functions are often encountered in areas such as computer vision [155] and real-time rendering [3]. In particular, we consider the following operator,

$$\mathcal{G}:\Theta\subset\mathbb{R}^3\to L^2(\Omega)$$

$$\mathcal{G}:\boldsymbol{\mu}\to u_{\boldsymbol{\mu}}(\mathbf{x}):=\min_{\substack{\mathbf{y}\in\partial\Omega,\\ y_2>\mu_1}}|\mathbf{y}-\mathbf{A}_{\mu_3}\mathbf{x}|e^{x_1\mu_2}$$

where $\boldsymbol{\mu}=(\mu_1,\mu_2,\mu_3)$ is a finite dimensional vector, and $\mathbf{A}_{\mu_3}=\text{diag}(1,\mu_3)$. In practice, the value of $u_{\boldsymbol{\mu}}(\mathbf{x})$ corresponds to the (weighted) distance between the dilated point $\mathbf{A}_{\mu_3}\mathbf{x}$ and the truncated boundary $\partial\Omega\cap\{\mathbf{y}:\ y_2>\mu_1\}$.

As input space we consider $\Theta:=[0,1]\times[-1,1]\times[1,2]$, endowed with the uniform probability distribution. Since the input is finite-dimensional, we can think of $\mathcal{G}$ as to the parametrization of a 3-dimensional hypersurface in $L^2(\Omega)$. We discretize Ω using P1 triangular finite elements with mesh stepsize $h=0.02$, resulting in the high-fidelity space $V_h:\cong\mathbb{R}^{13577}$. To learn the discretized operator $\mathcal{G}_h$, we employ the following MINN architecture

$$\mathbb{R}^3\to\mathbb{R}^{100}\to V_{9h}\xrightarrow{r=0.4}V_{3h}\xrightarrow{0.2}V_h.$$

The corresponding dense counterpart, which servers as benchmark, is obtained by removing the sparsity constraints (equivalently, by letting the supports go to infinity). We train the networks on 50 samples and for a total of 50 epochs.

Learning a local nonlinear operator

As a second experiment, we learn a nonlinear operator that is local with respect to the input. Let Ω be as in Figure 7.4b. We consider the *infinitesimal area* operator $\mathcal{G} : H^1(\Omega) \to L^2(\Omega)$,

$$\mathcal{G} : u \to \sqrt{1 + |\nabla u|^2}.$$

Note in fact that, if we associate to each $u \in H^1(\Omega)$ the cartesian surface

$$S_u := \{(x_1, x_2, u(x_1, x_2)\} \subset \mathbb{R}^3,$$

then $\mathcal{G}u$ yields a measure of the area that is locally spanned by that surface, in the sense that

$$\int_{S_u} \phi(\mathbf{s}) d\mathbf{s} = \int_{\Omega} \phi(u(\mathbf{x})) (\mathcal{G}u)(\mathbf{x}) d\mathbf{x}$$

for any continuous map $\phi : S_u \to \mathbb{R}$. Over the input space $\Theta := H^1(\Omega)$ we consider the probability distribution $\mathbb{P}$ induced by a Gaussian process with mean zero and covariance kernel

$$\text{Cov}(\mathbf{x}, \mathbf{y}) = \frac{1}{|\Omega|} \exp\left(-\frac{1}{2}|\mathbf{x} - \mathbf{y}|^2\right).$$

We discretize Ω using a triangular mesh of stepsize $h = 0.045$ and P1 finite elements, which results in a total of $N_h = 11266$ vertices. To sample from the Gaussian process we truncate its Karhunen-Loeve expansion at $k = 100$. Conversely, the output is computed numerically by exploiting the high-fidelity mesh as a reference.

To learn $\mathcal{G}_h$ we use the MINN architecture below,

$$V_h \xrightarrow{r=0.15} V_{3h} \xrightarrow{r=0.3} V_{3h} \xrightarrow{r=0.15} V_h,$$

which we train for 50 epochs over 500 snapshots.

Learning a nonlocal nonlinear operator

Since MINNs are based on local operations, it is of interest to assess whether they can also learn nonlocal operators. To this end, we consider the problem of learning the so-called Hardy-Littlewood Maximal Operator $\mathcal{G} : L^2(\Omega) \to L^2(\Omega)$,

$$(\mathcal{G}f)(\mathbf{x}) := \sup_{r>0} \fint_{|\mathbf{y}-\mathbf{x}|<r} |f(\mathbf{y})| d\mathbf{y}$$

which is known to be a continuous nonlinear operator from $L^2(\Omega)$ onto itself [154]. Here, we let $\Omega := B(\mathbf{0}, 1) \subset \mathbb{R}^2$ be the unit circle, see Figure 7.4c. Over the input space $\Theta := L^2(\Omega)$ we consider the probability distribution $\mathbb{P}$ induced by a Gaussian process with mean zero and covariance kernel

$$\text{Cov}(\mathbf{x}, \mathbf{y}) = \exp\left(-|\mathbf{x} - \mathbf{y}|^2\right).$$

As a high-fidelity reference, we consider a discretization of Ω via P1 triangular finite elements of stepsize $h = 0.033$, resulting in a state space V_h with $N_h = 7253$ degrees of freedom. As for the previous experiment, we sample from $\mathbb{P}$ by considering a truncated Karhunen-Loeve expansion of the Gaussian process (100 modes). Conversely, the true output $u \to \mathcal{G}_h(u)$ is computed approximately by replacing the suprema over r with a maxima across 50 equispaced radii in $[h, 2]$. To learn $\mathcal{G}_h$ we use a MINN of depth 2 with a dense layer in the middle,

$$V_h \xrightarrow{r=0.25} V_{2h} \to V_{2h} \xrightarrow{r=0.25} V_h.$$

The idea is that nonlocality can still be enforced through the use of fully connected blocks, but this are only inserted at the lower fidelity levels to reduce the computational cost. We train the architectures over 500 samples and for a total of 50 epochs.

Operator	N_h	Architecture	Test error	Gen. error
Low-dimensional manifold	13'577	Mesh-Informed Dense	4.14% 4.78%	3.08% 4.07%
Local area operator	11'266	Mesh-Informed Dense	1.49% 3.89%	1.36% 2.54%
H-L maximal operator	7'253	Mesh-Informed Dense	3.65% 6.70%	1.49% 3.09%
Nonlinear PDE solver	5'987	Mesh-Informed Dense	4.29% 18.53%	3.03% 6.56%

Table 7.1: Comparison of Mesh-Informed Neural Networks and Fully Connected DNNs for the test cases in Section 7.2. N_h = number of vertices in the (finest) mesh. Gen. error = Generalization error, defined as the gap between training and test errors. All reported errors are intended with respect to the L^2-norm, see Equation (7.3).

Learning the solution operator of a nonlinear PDE

For our final experiment, we consider the case of a parameter dependent PDE, which is a framework of particular interest in the literature of Reduced Order Modeling. In fact, learning the solution operator of a PDE model by means of neural networks allows one to replace the original numerical solver with a much cheaper and efficient surrogate, which enables expensive multi-query tasks such as PDE constrained optimal control, Uncertainty Quantification or Bayesian Inversion.

Here, we consider a steady version of the so-called porous media equation, defined as follows

$$-\nabla \cdot \left(|u|^2 \nabla u\right) + u = f.$$

The PDE is understood in the domain Ω defined in Figure 7.4d, and it is complemented with homogeneous Neumann boundary conditions. We define $\mathcal{G}$ to be the data-to-solution operator that maps $f \to u$. This time, we endow the input space with the push-forward distribution $\#\mathbb{P}$ induced by the square map $f \to f^2$, where $\mathbb{P}$ is the probability distribution associated to a Gaussian random field with mean zero and covariance kernel

$$\operatorname{Cov}(\mathbf{x}, \mathbf{y}) = \frac{1}{1 + |\mathbf{x} - \mathbf{y}|^2}.$$

To sample from the latter distribution we exploit a truncated Karhunen-Loeve expansion of the random field. We set the truncation index to $k = 20$ as that is sufficient to fully capture the volatility of the field. For the high-fidelity discretization, we consider a mesh of stepsize $h = 0.03$ and P1 finite elements, resulting in $N_h = 5987$. Finally, we employ the MINN architecture below,

$$V_h \xrightarrow{r=0.4} V_{4h} \to V_{2h} \xrightarrow{r=0.2} V_h,$$

which we train over 500 snapshots and for a total of 100 epochs. Note that, as in our third experiment, we employ a dense block at the center of the architecture. This is because the solution operator to a boundary value problem is typically nonlocal (consider for instance the so-called Green formula for the Poisson equation).

7.2.2 Numerical results

Table 7.1 reports the numerical results obtained across the four experiments. In general, MINNs perform better with respect to their dense counterpart, with relative errors that

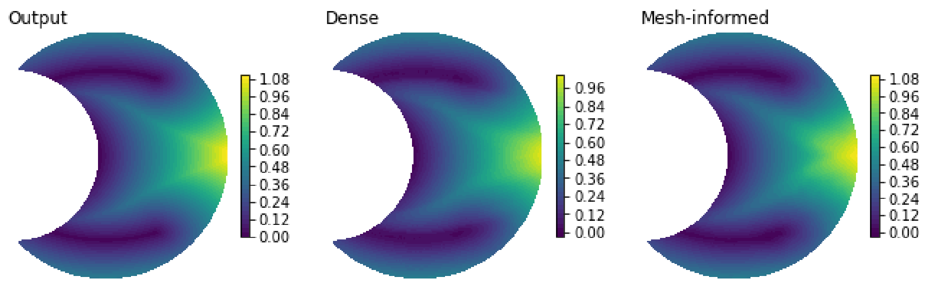


Figure 7.5: Comparison of DNNs and MINNs when learning a low-dimensional manifold $\boldsymbol{\mu} \to u_{\boldsymbol{\mu}} \in L^2(\Omega)$, cf. Section 7.2.1. The reported results correspond to the approximations obtained on an unseen input value $\boldsymbol{\mu}^* = [0.42, 0.04, 1.45]$.

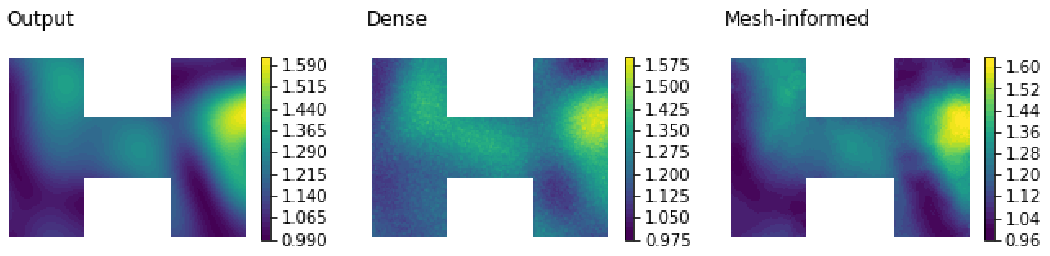


Figure 7.6: Comparison of DNNs and MINNs when learning the local operator $u \to \sqrt{1 + |\nabla u|^2}$, cf. Section 7.2.1. The reported results correspond to the approximations obtained for an input instance outside of the training set.

are always below 5%. As the operator to be learned becomes more and more involved, fully connected DNNs begin to struggle, eventually reaching an error of 18% in the PDE example. In contrast, MINNs are able to keep up and maintain their performance. This is also due to the fact that, having less parameters, MINNs are unlikely to overfit, instead they can generalize well even in poor data regimes (cf. last column of Table 7.1). Consider for instance the first experiment, which counted as little as 50 training samples. There, the dense model returns an error of 4.78%, of which 4.07% is due to the generalization gap. This means that the DNN model actually surpassed the MINN performance over the training set, as their training errors are respectively of $4.78\% - 4.07\% = 0.71\%$ and $4.14\% - 3.08\% = 1.06\%$. However, the smaller generalization gap allows the sparse architecture to perform better on unseen inputs.

Figures 7.5 to 7.8 reports some examples of approximation on unseen input values. There, we note that dense models tend to have noisy outputs (Figures 7.6 and 7.8) and often miscalculate the range of values spanned by the output (Figures 7.5 and 7.7). Conversely, MINNs always manage to capture the main features present in the actual ground truth.

Nonetheless, Mesh-Informed Neural Networks also allow for a significant reduction in the computational cost, as reported in Table 7.2. In general, MINNs are ten to a hundred times lighter with respect to fully connected DNNs. While this is not particularly relevant once the architecture is trained (the most heavy DNN weights as little as 124 Megabytes), it makes a huge difference during the training phase. In fact, additional resources are required to optimize a DNN model, as one needs to keep track of all the operations and gradients in order to perform the so-called *backpropagation* step. This poses a significant limitation to the use of dense architectures, as the entailed computational cost can easily exceed the capacity of modern GPUs. For instance, in our experiments, fully connected DNNs required more than 2 GB of memory during training, while, depending on the operator to be learned, 10 to 250 MB were sufficient for MINNs. Clearly, one could also alleviate the computational burden by exploiting cheaper optimization routines, such as first order optimizers and batching strategies, however this typically prevents the network from actually reaching the global

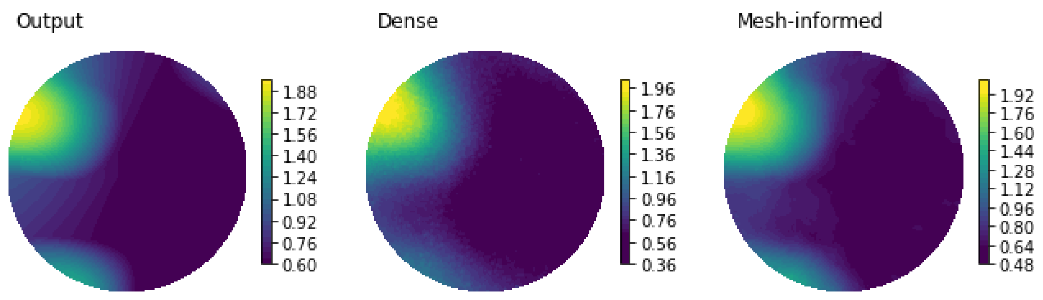


Figure 7.7: Comparison of DNNs and MINNs when learning the Hardy-Littlewood Maximal Operator, cf. Section 7.2.1. The pictures correspond to the results obtained for an unseen input instance.

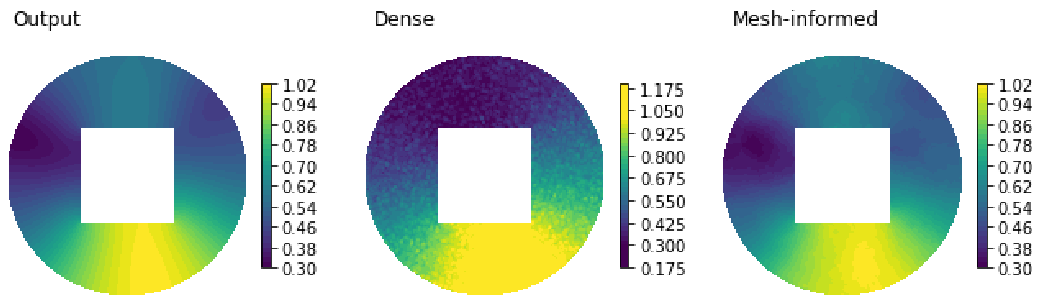


Figure 7.8: Comparison of DNNs and MINNs when learning the solution operator $f \to u$ of a nonlinear PDE, cf. Section 7.2.1. The reported results correspond to the approximations obtained for an input instance f outside of the training set.

Architecture	dof	Training speed	Memory usage	
			(static)	(training)
Mesh-Informed	0.3M	31.5 s/ep	1.3 Mb	0.01 Gb
Dense	21.7M	123.5 s/ep	86.9 Mb	3.49 Gb
Mesh-Informed	0.3M	60.5 s/ep	1.0 Mb	0.04 Gb
Dense	31.0M	195.3 s/ep	124.1 Mb	4.96 Gb
Mesh-Informed	1.0M	33.9 s/ep	3.9 Mb	0.16 Gb
Dense	12.8M	83.0 s/ep	51.2 Mb	2.05 Gb
Mesh-Informed	1.4M	70.0 s/ep	5.4 Mb	0.22 Gb
Dense	12.5M	130.7 s/ep	50.0 Mb	2.00 Gb

Table 7.2: Comparison of Mesh-Informed Neural Networks and Fully Connected DNNs in terms of their computational cost. dof = degrees of freedom, i.e. number of parameters to be optimized during the training phase. Memory usage (static) = bytes required to store the architecture. Memory usage (optimization) = bytes required to run a single epoch of the training phase. s/ep = seconds per epoch, M = millions, Mb = Megabytes, Gb = Gigabytes.

minimum of the loss function. In fact, we recall that the optimization of a DNN architecture is, in general, a non-convex and ill-posed problem. Of note, despite being 10 to 100 times lighter, MINNs are only 2 to 4 times faster during training. We believe that these results can be improved, possibly by optimizing the code used to implement MINNs.

7.3 Application to oxygen transfer models

We finally consider an application to Uncertainty Quantification (UQ) in the context of oxygen transfer models. UQ is an essential aspect of computational science and engineering, which often involves expensive numerical and statistical routines. In this Section, we provide an example on how MINNs can alleviate these costs by serving as model surrogates in the computational pipeline. In particular, starting from a generalization of the case study analyzed in Chapter 5, we address a problem of quantifying how the morphology of the vascular network affects the distribution of oxygen in biological tissues.

Model description

As we already discussed back in Chapter 4, oxygen is a fundamental constituent of most biological processes. In humans, oxygen is delivered by the circulatory system from the lungs to the rest of the body. At the small scales, cells receive oxygen from the vascular network of capillaries that spread all over the body. An efficient oxygen transfer is fundamental to ensure a healthy micro-environment of any biological tissue, and abnormal values in oxygen concentration are often associated to pathological scenarios. In particular, low oxygen supply, the so-called hypoxia, plays an important role in the development and treatment of tumors. It has been shown that hypoxic tissue opposes a resistance to chemotherapy and radiotherapy [35, 160]. These issues are caused by perturbed properties of the tumor blood vessels in terms of morphology and phenotype. Here, we aim at developing a methodology to assess the role of vascular morphology on tissue hypoxia. More precisely, we wish to address the following question: how does the topology of the vascular network relate to the size of the tissue under hypoxia? We answer this question in the simplified setting that we describe below.

Within an idealized setting, we consider a portion of a vascularized tissue $\Omega := \{\mathbf{x} \in \mathbb{R}^2 : |x| < 1\}$. Let $\Lambda \subset \overline{\Omega}$ be a graph representing the vascular network of capillaries (cf. Figure 7.9) and let $u : \Omega \to [0,1]$ be the oxygen concentration in the tissue, normalized to the unit value. We model the oxygen transfer from the network to the tissue with the following equations,

$$\begin{cases} -\alpha\Delta u + u = (1-u)\hat{\delta}_\Lambda & \text{in } \Omega \\ -\alpha\nabla u \cdot \mathbf{n} = \beta u & \text{on } \partial\Omega \end{cases} \tag{7.4}$$

where $\alpha = 0.1$ and $\beta = 0.01$ are respectively a fixed diffusion and resistance coefficient, while $\hat{\delta}_\Lambda$ is the unique singular measure for which

$$\int_\Omega v(\mathbf{x})\hat{\delta}_\Lambda(d\mathbf{x}) = \frac{1}{|\Lambda|}\int_\Lambda v(\mathbf{s})d\mathbf{s}$$

for all $v \in \mathcal{C}(\overline{\Omega})$. Here, we denote by $|\Lambda| := \int_\Lambda 1 d\mathbf{s}$ the total length of the vascular graph. The first equation in (7.4) describes the diffusion and consumption of oxygen, balanced accordingly to the amount released from the vascular network on the right hand side. Finally, the model is closed using resistance boundary conditions of Robin type. We understand (7.4) in the weak sense, meaning that define $u = u_\Lambda$ as the unique solution to the problem below

$$\int_\Omega \alpha\nabla u(\mathbf{x}) \cdot \nabla v(\mathbf{x})d\mathbf{x} + \int_\Omega u(\mathbf{x})v(\mathbf{x})d\mathbf{x} + \int_{\partial\Omega} \beta u(\mathbf{s})v(\mathbf{s})d\mathbf{s} = \\ = \frac{1}{|\Lambda|}\int_\Lambda (1-u(\mathbf{s}))v(\mathbf{s})d\mathbf{s} \tag{7.5}$$

where the above is to be satisfied for all $v \in \mathcal{C}^\infty(\Omega)$.

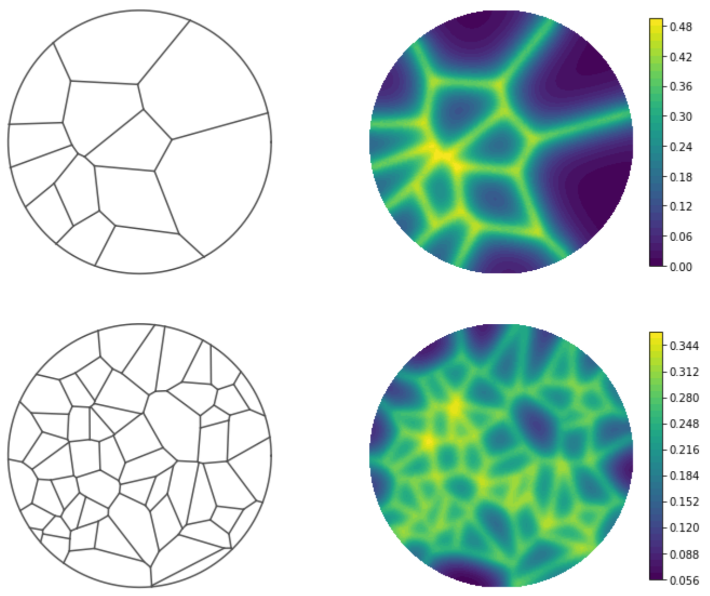


Figure 7.9: Forward UQ problem (Section 7.3). Topology of the microvascular network Λ (left) and corresponding oxygen distribution in the tissue $u = u_\Lambda$ (right). The top and the bottom rows corresponds respectively to a poorly and a highly vascularized tissue (resp. $\lambda = 1$ and $\lambda = 3$). Globally, the two networks provide the same amount of oxygen (cf. Equation 7.5), but their topology significantly affects the values of u in the tissue. In the first case (top row), nearly 31% of the tissue has an oxygen level below the threshold value $u_* := 0.1$. Conversely, only 3% of the tissue reports a low oxygen concentration in the second example.

Uncertainty quantification setting

As we mentioned previously, we are interested in the relationship between Λ and u. To this end, we introduce the parameter space

$$\Theta := \{\Lambda \subset \overline{\Omega} : \ \Lambda \text{ is the union of finitely many segments}\}$$

which consists of all vascular networks. Note that, due to the normalizing factor $1/|\Lambda|$ in (7.5), all the vascular networks actually provide the same global amount of oxygen. However, as we will see later on, only those vascular graphs that are sufficiently spread across the domain can ensure a proper oxygen supply to the whole tissue (cf. Figure 7.9). In other words, we explore the influence of the distribution of the network, rather than its source density, on the oxygen level.

The next subsection is devoted to prescribing a suitable discretization of (7.5) to work with, and to introduce a class of probability measures $\{\mathbb{P}_\lambda\}_\lambda$ defined over Θ. The idea is the following. We will use a macro-scale parameter λ to describe the general perfusion of the tissue. Higher values of λ will correspond to a highly vascularized tissue. This means that the topology of the vascular network will still be uncertain, but the corresponding probability distribution $\mathbb{P}_\lambda$ will favor dense graphs. Conversely, lower values of λ will describe scenarios where capillaries are more sparse (see Figure 7.9, top vs bottom row). This will then bring us to consider the family of random variables

$$Q_\lambda := \frac{1}{|\Omega|}|\{u_\Lambda < 0.1\}| \text{ with } \Lambda \sim \mathbb{P}_\lambda,$$

that measure the portion of the tissue under the oxygen threshold 0.1, which we take as the value under which hypoxia takes place. Our interest will be to estimate the probability density function of each Q_λ and capture their overall behavior. While these tasks can be achieved using classical Monte Carlo, the computational cost is enormous as it implies solving equation (7.5) repeatedly. To alleviate this burden, we will replace the original PDE solver with a suitable MINN architecture trained to learn the map $\Lambda \to u_\Lambda$.

Discretization and implementation details

For the random generation of vascular networks we exploit Voronoi diagrams [6]. Let $\mathcal{P} := \{P \subset \Omega \mid P \text{ finite}\}$ be the collection of all points tuples in Ω. To any $P \in \mathcal{P}$, we associate the vascular graph $\Lambda(P)$ defined by the edges of the Voronoi cells generated by P. In this way, we obtain a correspondence $\mathcal{P} \to \Theta$ given by $P \to \Lambda(P)$, that we can exploit to prescribe probability measures over Θ. To this end, let $\lambda > 0$ and let X_λ be a Poisson point process over Ω having a uniform intensity of 10λ. We denote by $\tilde{\mathbb{P}}_\lambda$ the probability measure induced by X_λ over $\mathcal{P}$. Then, we define $\mathbb{P}_\lambda := \#\tilde{\mathbb{P}}_\lambda$ as the push-forward measure obtained via the action $P \to \Lambda(P)$. This ensures the wished behavior: higher values of λ tend to generate more points in the domain and, consequently, denser graphs.

We now proceed to discretize the variational problem. As a first step, we note that the vascular graph Λ is not given in terms of a parametrization, which makes it harder to compute integrals of the form $\int_\Lambda v(\mathbf{s})d\mathbf{s}$. As an alternative, we resort to the smoothed approximation that we introduced back in Chapter 5, Section 5.5, i.e.

$$\int_\Lambda v(\mathbf{s})d\mathbf{s} \approx \int_\Omega v(\mathbf{x})\sigma_\Lambda(\mathbf{x})d\mathbf{x}, \tag{7.6}$$

where

$$\sigma_\Lambda := \frac{1}{\epsilon^2}\max\{\epsilon - \text{dist}(\mathbf{x},\Lambda), 0\}.$$

We recall that, as shown in the appendix of Chapter 5, the right-hand side of equation (7.6) converges to the left hand-side as $\epsilon \downarrow 0^+$ for each fixed $v \in \mathcal{C}(\overline{\Omega})$. Here, we let $\epsilon = 0.05$. Then, our operator of interest becomes $\mathcal{G} : \sigma_\Lambda \to u_\Lambda$, and we can proceed with our usual discretization via P1 Finite Elements. To this end, we discretize the domain using a triangular mesh of stepsize $h = 0.03$, which results in $N_h = 7253$ degrees of freedom. Then, we allow λ to vary uniformly in $\{1, 2, \dots, 10\}$ and we generate a total of 4500 training snapshots accordingly to the probability distributions introduced previously. We exploit these snapshots to train the MINN model below,

$$V_h \xrightarrow{r=0.1} V_{3h} \to V_{3h} \xrightarrow{r=0.1} V_h,$$

where the architecture has been defined in analogy to the one employed for the nonlinear PDE in Section 7.2.1. The network is trained for a total of 50 epochs and using the same criteria presented in Section 7.2.

Results

Once trained, the Mesh-Informed Neural Network reported an average L^2-error of 3.89%, with errors below 5% for 454 out of 500 test instances. Figure 7.10 shows the approximation for three unseen vascular networks of increasing density. We considered these results satisfactory and we proceeded to sample a total of 100'000 solutions using our DNN model. More precisely, we considered 100 equally spaced values of λ in [1,10], and for each of those we sampled 1'000 independent solutions. From there, we obtained an i.i.d. sample of size 1'000 for each of the Q_{λ_i}, where $\lambda_i = \{1 + i/11\}_{i=0}^{99}$. Results are in Figure 7.11.

The left panel of Figure 7.11 shows the pointwise approximation of the map $\lambda \to \mathbb{E}[Q_\lambda]$, together with the inter-quantile and inter-decile bands of Q_λ. Given the large amount of data generated, spurious oscillations are most likely due to the numerical errors introduced by the MINN model, rather than from statistical noise. Coherently with the physical interpretation of λ, we see that the probability of low oxygenation decreases with the vascular density. Interestingly, although the total intensity of the source term is normalized to the

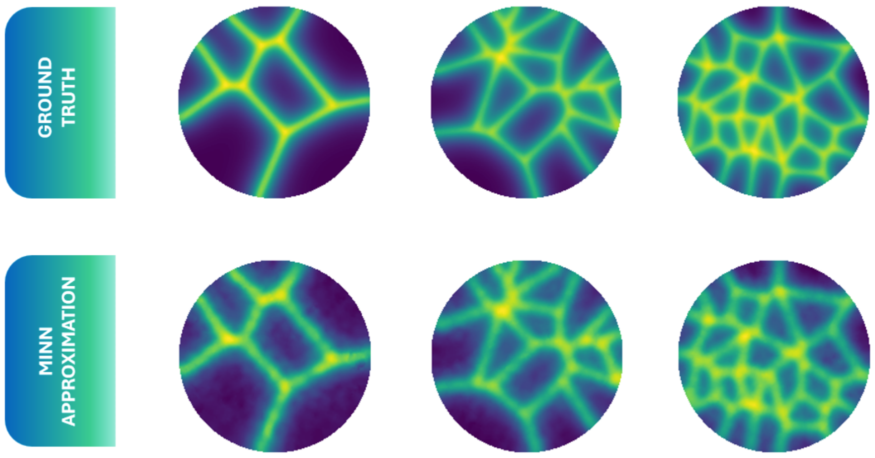


Figure 7.10: Ground truth samples (top row) and corresponding MINN approximations (bottom) for three network topologies not included in the training set.

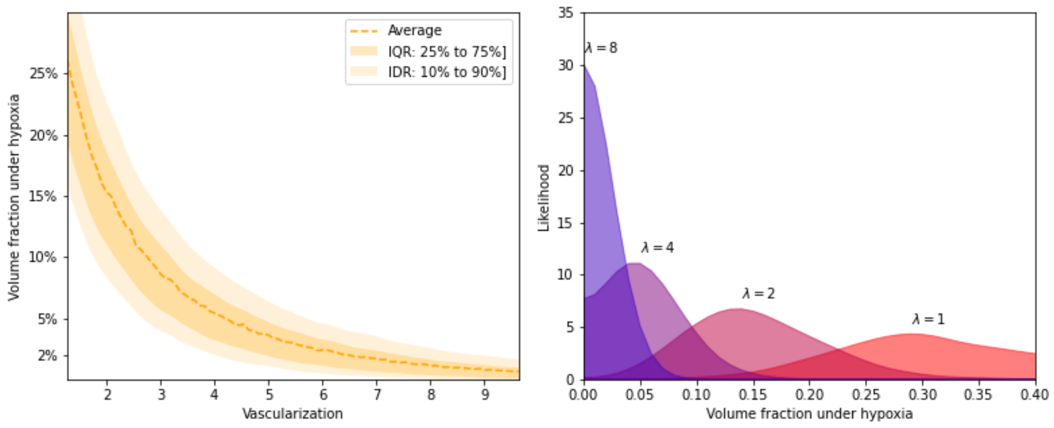


Figure 7.11: Results for the UQ problem in Section 7.3. Left panel: average volume fraction exposed to low oxygen supply, $\mathbb{E}[Q_\lambda]$, as a function of the vascularization level λ, enriched with inter-quantile (IQR) and inter-decile (IDR) bands. Right panel: probability distribution of Q_λ for different values of λ. Colors fade from red to purple as λ grows.

same level in any configuration, the networks with higher gaps between neighboring edges are prone to spots of low oxygen concentration. Not only, the decay appears to be exponential. Further investigations seem to confirm this intuition, as we obtain an R^2-coefficient of 0.987 when trying to relate λ and $\log \mathbb{E}[Q_\lambda]$ via linear regression. The quantile bands, instead, allows us to notice how the uncertainty in the quantity interest is larger for smaller values of λ. In particular, we can see how even poorly vascularized tissues can present a healthy environment under suitable rare circumstances. Conversely, the behavior of Q_λ tends toward determinism as λ increases. This same phenomenon is also well illustrated by the right panel of Figure 7.11. The latter shows how the probability distribution of Q_λ changes according to λ. To account for the fact that the densities are compactly supported, as $0 \leq Q_\lambda \leq 1$, we have resorted to diffusion methods for their estimation, see [30]. Once again, we see that the densities are more spread out when λ is near 1, while they shrink towards zero as λ increases. This is coherent with the physical intuition, and we would expect the density of Q_λ to converge to a Dirac delta as $\lambda \to +\infty$.

In real scenarios where the physical complexity of a vascularized tissue is appropriately described as in [160], this analysis would be computationally viable only with the employment of the MINN model as a surrogate for the numerical solver. In the case presented here, both the full order model and the surrogate model are computationally inexpensive. However, the former required around 2 minutes to generate 1'000 PDE solutions. Conversely, the trained DNN model was able to provide the same number of solutions in as little as 3 milliseconds, corresponding to a speed up factor of approximately 40'000. For multiphysics models where a simulation of a single point in the parameter space could cost hours of wall computational time, such gain could enable approaches that would be otherwise unreasonable. Even in the present simplified setting, such a boost also makes up for the computational effort required to train the network. In fact: (i) collecting the training snapshots took 575.96 seconds, (ii) training the MINN model required 926.40 seconds, (iii) generating the 100'000 new solutions took 0.3 seconds. In contrast, the numerical solver would generate at most ≈ 13'000 solutions within the same time frame. These considerations support the interest in further developing model order reduction techniques based on deep neural networks that are robust for general spatial domains, such as MINNs. Indeed, we are currently developing model reduction techniques applied to realistic models of the vascular microenvironment that leverage on the DL-ROM framework, combined with the efficiency of MINNs. However, this is still an ongoing work that goes beyond the scope of the Thesis.

7.3.1 Application to radiotherapy and TCP models

We conclude this Section by extending the above reasoning to TCP models, with a more direct application to radiotherapy optimization. In fact, as we discussed back in Chapter 4, there is a deep connection between the perfusion of oxygen and the sensitivity to radiation in biological tissues.

Even though system (7.4) describes an ideal dimensionless setting, we can still provide qualitative insights about this biological mechanism through the use of Equations (4.4) and (4.3). More precisely, let us assume that the domain under study describes a tumoral region, with cancerous cells populating the tissue with a uniform density N. Given a radiation dose D, the probability of tumor eradication is

$$\text{TCP}(u, D) = \exp\left(-N\int_{\Omega} e^{-\alpha(u(\mathbf{x}))D-\beta(u(\mathbf{x}))D^2}d\mathbf{x}\right),$$

where we recall that

$$\alpha(u) = \frac{(a_1 + a_2\ell)\,u + (a_3 + a_4\ell)\eta}{u+\eta}, \qquad \beta(u) = \left(\frac{b_1 u + b_2\eta}{u+\eta}\right)^2.$$

We set the several constants as follows: $a_1 = 0.2$, $a_2 = 6\cdot 10^{-3}$, $a_3 = 0.04$, $a_4 = 3.1\cdot 10^{-3}$, $b_1 = 0.64$, $b_2 = 0.14$, $\ell = 2$, $\eta = 6.5\cdot 10^{-2}$, $N = 50$, $D = 10$.

Since our MINN model provided an approximation of the overall solution field u, we can directly study this new quantity of interest without any additional training. This flexibility is undoubtedly a great advantage with respect to a black-box input-output approach, which would require two separate models for Q_λ and $\text{TCP}(\cdot, D)$.

We report the results in Figure 7.12, in the same fashion of our previous analysis. Coherently with what we would expect, the TCP shows a reverse behavior with respect to Q_λ. As λ increases, the probability of tumor eradication saturates exponentially fast to 1 (left panel in Figure 7.12). In general, the probability distributions describing the uncertainity in the TCP are very different as we move from sparse to dense vascular networks. Interestingly, for $\lambda = 1$, most of the likelihood is concentrated toward zero, highlighting the fact that complete elimination of the tumor is a rarity under those circumstances; however, even a slight increase in the vascularization results in a significant change. Indeed, for $\lambda = 1.45$ we see that the probability distribution is spread all over the domain in a seemingly uniform way, indicating that tumor eradication and survival are almost equally likely. Clearly, all these considerations are qualitative as we would need to consider much accurate oxygen transfer models to provide useful clinical insights. Also, since tumors are typically hypoxic, future analysis should be restricted values of λ below a suitable threshold.

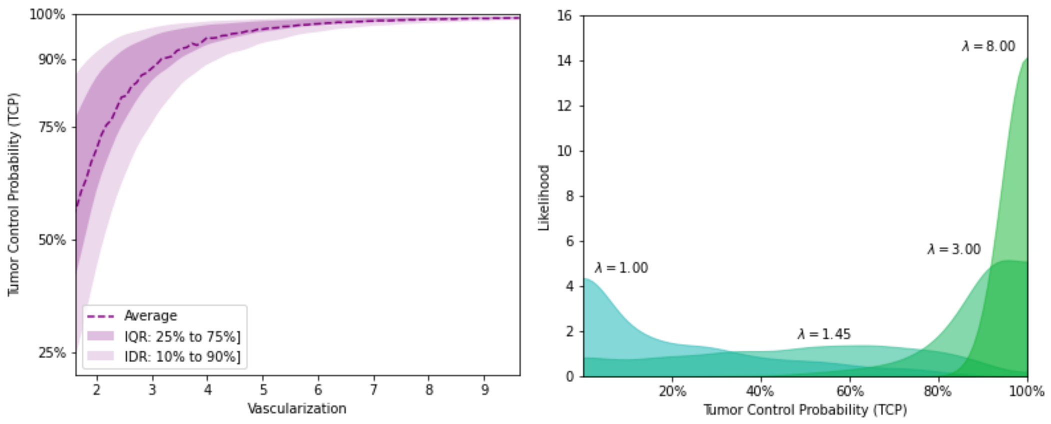


Figure 7.12: Analysis of the relation between tissue vascularization and TCP at fixed radiation dose. Left panel: average TCP as a function of the vascularization level λ, enriched with inter-quantile (IQR) and inter-decile (IDR) bands. Right panel: probability distribution of $\text{TCP}(u, D)$ for different values of λ. Colors fade from blue to green as λ grows.

7.4 Conclusions

In this Chapter, we have introduced Mesh-Informed Neural Networks (MINNs), a novel class of sparse DNN models that can be used to learn general operators between infinite dimensional spaces. The approach is based on an apriori pruning strategy that is obtained by embedding the hidden states into discrete functional spaces of different fidelities. Despite being very easy to implement, MINNs show remarkable advantages with respect to dense architectures, such as a massive reduction in the computational costs and an increased ability to generalize over unseen samples. This is coherent with the results available in the pruning literature [26], even though the setting differs from the one considered thereby.

We have tested MINNs over a large variety of scenarios, going from low dimensional manifolds to parameter dependent PDEs, showing that these architectures can learn non-linear operators for general shapes of the underlying spatial domain. This opens a wide new range of research directions that we wish to investigate further in future works. For instance, one could test the use of MINNs in more sophisticated Deep Learning based Reduced Order Models for PDEs (DL-ROMs), such as those in [65, 70, 71, 118]. In addition, considering how MINNs are actually built, it may be interesting to see whether one can take advantage of multi-fidelity strategies during the training phase, as in [80].

Finally, to showcase the potential advantages provided by these approaches to precision medicine, we have reported a case study concerning uncertainty quantification in radiotherapy treatments. There, thanks to the efficiency of MINNs, we were able to obtain novel insights about the role of microvascular networks in oxygen transfer and tumor eradication. While limited to a simplified setting, these results encourage us to further investigate and transfer these ideas to more complex and accurate physical models.

Conclusion

In this Thesis, we addressed the use of Machine Learning techniques for precision medicine, with particular emphasis on applications to personalized treatment planning of radiotherapy. In the first part of the dissertation, we developed a new approach to polygenic risk scoring, which, leveraging on Deep Learning and Data Mining algorithms, is able to construct interaction-aware predictive biomarkers. Then, in the second part we exploited Deep Learning models to build efficient model surrogates that can replace the expensive simulation of physical models in the complex workflow of biomarker discovery.

The methodologies developed across the several Chapters are extremely diverse, but they share common tools and objectives. Most importantly, they provide a comprehensive framework for precision medicine, rooted on the fruitful combination of data-driven and physics based approaches. We believe that the advantages provided by such cooperation are well illustrated by our studies concerning radiotherapy treatments. There, the coalescence of data-driven and physics based approaches allowed us to tackle the problem from two different perspectives. On the one hand, the personalization of the treatment through NTCP models, which, by including clinical and genetic variables, aims at the prevention of long term complications associated to radiation. On the other hand, instead, the study of TCP models through the numerical simulation of biophysical systems, with the goal of improving the efficiency of radiotherapy in eliminating cancer cells.

This dichotomy, however, is not specific of this clinical setting. In fact, it is often the case that biological mechanisms can be understood both through mathematical models and statistical data. Researchers, in biology and mathematics, have long worked on the rigorous description of living systems, providing us innovative ways for studying healthy and pathological scenarios. In the meantime, medical doctors have collected incredible amounts of data about their patients, ranging from anamnesis to genetic data, which has pushed statisticians and data scientists to further develop new approaches for their analysis. In this sense, the fusion of data and physics provides a new paradigm for precision medicine: an ambitious objective that can only been achieved through suitable tools, such as Machine Learning.

Ongoing research and future directions

Throughout the Thesis, we have presented several methodologies, highlighting their potential but also discussing their limitations and considering possible improvements. Future research in this field may involve both the amelioration of the proposed approaches, as well as the development of novel tools and the exploration of new clinical applications.

Currently, we are working towards these directions along multiple perspectives. For what concerns genome-based approaches, we are now investigating the applicability of the DSAEE and *hi*PRS algorithms (Chapters 2 and 3, respectively) to a different clinical cohort, consisting of breast cancer patients. Here, the subject of study is once again the occurrence of late radiotherapy toxicity, which in this case can result in long term complications associated, e.g., to breast oedema, skin induration and telangiectasia. However, the research is still ongoing as we are slowly acquiring more data thanks to our collaboration in the RADPrecise project. Within this same framework, it would also be interesting to investigate the use of other omic-data, such as transcriptomic data associated to mRNA and micro-arrays. However, this would require a clever adaptation of our approaches, especially *hi*PRS, as these are not categorical but rather continuous data.

Another research direction, instead, concerns the generalization of these approaches to time-to-event outcomes (survival analysis). In fact, in some cases, it is more natural to ask *when* an event will occur, rather than it if ever will. This poses significant challenges as the adaptation of our PRS algorithm to such contexts is not straightforward. At the moment, we are exploring extensions of our approach based on time-to-event correlation measures, such as Kendall's tau index, with promising results.

For what concerns the study of physics based models for precision medicine, we are also working on several fronts. On the one hand, we are considering the application of our methodologies to more complex and accurate physical models, such as the microvascular oxygen transfer model presented in Chapter 4. On the other hand, we are exploring suitable generalizations and extensions of the DL-ROM approach (Chapter 5), in order to tackle time-dependent problems and handle unbounded parameter spaces, with some preliminary insights [69]. Another frontier of research is instead related to the direct integration of physical knowledge, a topic now explored by many, e.g. [41, 164, 166, 196], which is expected to reduce data requirements and improve parameter extrapolation.

Finally, we are also working on completely new approaches that combine the efficiency of deep neural networks with the properties of certified linear methods. In particular, we are exploring the use of Deep Learning for the development of adaptive-basis surrogates, in the same spirit of other works found in the literature, e.g. [5].

Adapting the words of the italian writer Alessandro Manzoni:

"We hope you enjoyed reading our work.
But, in case you found it heavy or tiresome,
please trust us: that was not intentional."

:

Bibliography

[1] Abraham, G., Kowalczyk, A., Zobel, J., and Inouye, M. "Performance and robustness of penalized and unpenalized methods for genetic prediction of complex human disease". In: *Genetic epidemiology* 37.2 (2013), pages 184–195.

[2] Adams, R. A. and Fournier, J. J. *Sobolev spaces*. Elsevier, 2003.

[3] Akenine-Moller, T., Haines, E., and Hoffman, N. *Real-time rendering*. AK Peters/crc Press, 2019.

[4] Alman, J. and Williams, V. V. "A refined laser method and faster matrix multiplication". In: *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM. 2021, pages 522–539.

[5] Amsallem, D. and Farhat, C. "Interpolation method for adapting reduced-order models and application to aeroelasticity". In: *AIAA journal* 46.7 (2008), pages 1803–1813.

[6] Aurenhammer, F. "Voronoi diagrams—a survey of a fundamental geometric data structure". In: *ACM Computing Surveys (CSUR)* 23.3 (1991), pages 345–405.

[7] Ayers, K. L. and Cordell, H. J. "SNP selection in genome-wide and candidate gene studies via penalized logistic regression". In: *Genetic epidemiology* 34.8 (2010), pages 879–891.

[8] Babuška, I., Nobile, F., and Tempone, R. "A stochastic collocation method for elliptic partial differential equations with random input data". In: *SIAM Journal on Numerical Analysis* 45.3 (2007), pages 1005–1034.

[9] Bachmayr, M. and Cohen, A. "Kolmogorov widths and low-rank approximations of parametric elliptic PDEs". In: *Mathematics of Computation* 86.304 (2017), pages 701–724.

[10] Badré, A., Zhang, L., Muchero, W., Reynolds, J. C., and Pan, C. "Deep neural network improves the estimation of polygenic risk scores for breast cancer". In: *Journal of Human Genetics* 66.4 (2021), pages 359–369.

[11] Baeyens, A., Thierens, H., Claes, K., Poppe, B., Messiaen, L., De Ridder, L., and Vral, A. "Chromosomal radiosensitivity in breast cancer patients with a known or putative genetic predisposition". In: *British journal of cancer* 87.12 (2002), pages 1379–1385.

[12] Balcer-Kubiczek, E. "Apoptosis in radiation therapy: a double-edged sword". In: *Experimental oncology* (2012).

[13] Barrault, M., Maday, Y., Nguyen, N. C., and Patera, A. T. "An 'empirical interpolation'method: application to efficient reduced-basis discretization of partial differential equations". In: *Comptes Rendus Mathematique* 339.9 (2004), pages 667–672.

[14] Bateson, W. and Mendel, G. *Mendel's principles of heredity*. Courier Corporation, 2013.

[15] Behravan, H., Hartikainen, J. M., Tengström, M., Kosma, V.-.-M., and Mannermaa, A. "Predicting breast cancer risk using interacting genetic and demographic factors and machine learning". In: *Scientific Reports* 10.1 (2020), pages 1–16.

[16] Behravan, H., Hartikainen, J. M., Tengström, M., Pylkäs, K., Winqvist, R., Kosma, V.-.-M., and Mannermaa, A. “Machine learning identifies interacting genetic variants contributing to breast cancer risk: A case study in Finnish cases and controls”. In: *Scientific reports* 8.1 (2018), pages 1–13.

[17] Benfenati, A., Causin, P., Oberti, R., and Stefanello, G. “Unsupervised deep learning techniques for powdery mildew recognition based on multispectral imaging”. In: *arXiv preprint arXiv:2112.11242* (2021).

[18] Benyamini, Y. and Lindenstrauss, J. *Geometric nonlinear functional analysis*. Volume 48. American Mathematical Soc., 1998.

[19] Berg, J. J., Harpak, A., Sinnott-Armstrong, N., Joergensen, A. M., Mostafavi, H., Field, Y., Boyle, E. A., et al. “Reduced signal for polygenic adaptation of height in UK Biobank”. In: *Elife* 8 (2019), e39725.

[20] Berlanga, C. and Flores-Ramos, M. “Different gender response to serotonergic and noradrenergic antidepressants. A comparative study of the efficacy of citalopram and reboxetine”. In: *Journal of affective disorders* 95.1-3 (2006), pages 119–123.

[21] Bernabeu, M. O., Köry, J., Grogan, J. A., Markelc, B., Beardo, A., d’Avezac, M., Enjalbert, R., et al. “Abnormal morphology biases hematocrit distribution in tumor vasculature and contributes to heterogeneity in tissue oxygenation”. In: *Proceedings of the National Academy of Sciences* 117.45 (2020), pages 27811–27819.

[22] Bhaskaran, S. and Swaminathan, M. “Polyploidy and radiosensitivity in wheat and barley”. In: *Genetica* 32.1 (1962), pages 200–246.

[23] Bhattacharjee, S. and Matouš, K. “A nonlinear manifold-based reduced order model for multiscale analysis of heterogeneous hyperelastic materials”. In: *Journal of Computational Physics* 313 (2016), pages 635–653.

[24] Bhattacharya, K., Hosseini, B., Kovachki, N. B., and Stuart, A. M. “Model reduction and neural networks for parametric PDEs”. In: *arXiv preprint arXiv:2005.03180* (2020).

[25] Binev, P., Cohen, A., Dahmen, W., DeVore, R., Petrova, G., and Wojtaszczyk, P. “Convergence rates for greedy algorithms in reduced basis methods”. In: *SIAM journal on mathematical analysis* 43.3 (2011), pages 1457–1472.

[26] Blalock, D., Gonzalez Ortiz, J. J., Frankle, J., and Guttag, J. “What is the state of neural network pruning?” In: *Proceedings of machine learning and systems* 2 (2020), pages 129–146.

[27] Bolcskei, H., Grohs, P., Kutyniok, G., and Petersen, P. “Optimal approximation with sparsely connected deep neural networks”. In: *SIAM Journal on Mathematics of Data Science* 1.1 (2019), pages 8–45.

[28] Borsuk, K. “Drei Sätze über die n-dimensionale euklidische Sphäre”. In: *Fundamenta Mathematicae* 20.1 (1933), pages 177–190.

[29] Borsuk, K. and Ulam, S. “On symmetric products of topological spaces”. In: *Bulletin of the American Mathematical Society* 37.12 (1931), pages 875–882.

[30] Botev, Z. I., Grotowski, J. F., and Kroese, D. P. “Kernel density estimation via diffusion”. In: *The annals of Statistics* 38.5 (2010), pages 2916–2957.

[31] Bresolin, A., Garibaldi, E., Faiella, A., Cante, D., Vavassori, V., Waskiewicz, J. M., Girelli, G., et al. “Predictors of 2-year incidence of patient-reported urinary incontinence after post-prostatectomy radiotherapy: Evidence of dose and fractionation effects”. In: *Frontiers in oncology* 10 (2020), page 1207.

[32] Bui-Thanh, T., Burstedde, C., Ghattas, O., Martin, J., Stadler, G., and Wilcox, L. C. “Extreme-scale UQ for Bayesian inverse problems governed by PDEs”. In: *SC’12: Proceedings of the international conference on high performance computing, networking, storage and analysis*. IEEE. 2012, pages 1–11.

[33] Carreau, A., Hafny-Rahbi, B. E., Matejuk, A., Grillon, C., and Kieda, C. “Why is the partial oxygen pressure of human tissues a crucial parameter? Small molecules and hypoxia”. In: *Journal of cellular and molecular medicine* 15.6 (2011), pages 1239–1253.

[34] Casas, E. “L^2 estimates for the finite element method for the Dirichlet problem with singular data”. In: *Numerische Mathematik* 47.4 (1985), pages 627–632.

[35] Cattaneo, L. and Zunino, P. “A computational model of drug delivery through microcirculation to compare different tumor treatments”. In: *International journal for numerical methods in biomedical engineering* 30.11 (2014), pages 1347–1371.

[36] Cecile, A., Janssens, J., and Joyner, M. J. “Polygenic risk scores that predict common diseases using millions of single nucleotide polymorphisms: is more, better?” In: *Clinical chemistry* 65.5 (2019), pages 609–611.

[37] Chasioti, D., Yan, J., Nho, K., and Saykin, A. J. “Progress in polygenic composite scores in Alzheimer’s and other complex diseases”. In: *Trends in Genetics* 35.5 (2019), pages 371–382.

[38] Che, R. and Motsinger-Reif, A. “Evaluation of genetic risk score models in the presence of interaction and linkage disequilibrium”. In: *Frontiers in genetics* 4 (2013), page 138.

[39] Chen, J., Sathe, S., Aggarwal, C., and Turaga, D. “Outlier detection with autoencoder ensembles”. In: *Proceedings of the 2017 SIAM international conference on data mining*. SIAM. 2017, pages 90–98.

[40] Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. “Neural ordinary differential equations”. In: *Advances in neural information processing systems* 31 (2018).

[41] Chen, W., Wang, Q., Hesthaven, J. S., and Zhang, C. “Physics-informed machine learning for reduced-order modeling of nonlinear problems”. In: *Journal of computational physics* 446 (2021), page 110666.

[42] Chicco, M. “Principio di massimo forte per sottosoluzioni di equazioni ellittiche di tipo variazionale.” In: *Bollettino dell’Unione Matematica Italiana* 22.3 (1967), pages 368–372.

[43] Choi, S. W., Mak, T. S.-H., and O’Reilly, P. F. “Tutorial: a guide to performing polygenic risk score analyses”. In: *Nature Protocols* 15.9 (2020), pages 2759–2772.

[44] Clarke, G. M., Anderson, C. A., Pettersson, F. H., Cardon, L. R., Morris, A. P., and Zondervan, K. T. “Basic statistical analysis in genetic case-control studies”. In: *Nature protocols* 6.2 (2011), pages 121–133.

[45] Cohen, A., Devore, R., Petrova, G., and Wojtaszczyk, P. “Optimal stable nonlinear approximation”. In: *Foundations of Computational Mathematics* 22.3 (2022), pages 607–648.

[46] Collins, R. L., Hu, T., Wejse, C., Sirugo, G., Williams, S. M., and Moore, J. H. “Multifactor dimensionality reduction reveals a three-locus epistatic interaction associated with susceptibility to pulmonary tuberculosis”. In: *BioData mining* 6.1 (2013), pages 1–5.

[47] Consortium, E. et al. “A roadmap for precision medicine in the epilepsies”. In: *The Lancet Neurology* 14.12 (2015), pages 1219–1228.

[48] Cordell, H. J. “Epistasis: what it means, what it doesn’t mean, and statistical methods to detect it in humans”. In: *Human molecular genetics* 11.20 (2002), pages 2463–2468.

[49] Cox, C. B., Moore, P. D., and Ladle, R. J. *Biogeography: an ecological and evolutionary approach*. John Wiley & Sons, 2016.

[50] Cybenko, G. “Approximation by superpositions of a sigmoidal function”. In: *Mathematics of control, signals and systems* 2.4 (1989), pages 303–314.

[51] D'angelo, C. and Quarteroni, A. "On the coupling of 1d and 3d diffusion-reaction equations: application to tissue perfusion problems". In: *Mathematical Models and Methods in Applied Sciences* 18.08 (2008), pages 1481–1504.

[52] Daubechies, I., DeVore, R., Foucart, S., Hanin, B., and Petrova, G. "Nonlinear Approximation and (Deep) ReLU Networks". In: *Constructive Approximation* 55.1 (2022), pages 127–172.

[53] DeVore, R., Hanin, B., and Petrova, G. "Neural network approximation". In: *Acta Numerica* 30 (2021), pages 327–444.

[54] DeVore, R. A., Howard, R., and Micchelli, C. "Optimal nonlinear approximation". In: *Manuscripta mathematica* 63.4 (1989), pages 469–478.

[55] Dosovitskiy, A., Tobias Springenberg, J., and Brox, T. "Learning to generate chairs with convolutional neural networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2015, pages 1538–1546.

[56] Druţu, C. and Kapovich, M. *Geometric group theory.* Volume 63. American Mathematical Soc., 2018.

[57] Dugundji, J. "An extension of Tietze's theorem." In: *Pacific Journal of Mathematics* 1.3 (1951), pages 353–367.

[58] Engelking, R. *Dimension theory.* Volume 19. North-Holland Publishing Company Amsterdam, 1978.

[59] Escott-Price, V., Myers, A. J., Huentelman, M., and Hardy, J. "Polygenic risk score analysis of pathologically confirmed Alzheimer disease". In: *Annals of neurology* 82.2 (2017), pages 311–314.

[60] Evans, L. C. *Partial differential equations.* Volume 19. American Mathematical Soc., 2010.

[61] Fachal, L., Gómez-Caamaño, A., Barnett, G. C., Peleteiro, P., Carballo, A. M., Calvo-Crespo, P., Kerns, S. L., et al. "A three-stage genome-wide association study identifies a susceptibility locus for late radiotherapy toxicity at 2q24. 1". In: *Nature genetics* 46.8 (2014), pages 891–894.

[62] Feng, Y., Feng, Y., You, H., Zhao, X., and Gao, Y. "Meshnet: Mesh neural network for 3d shape representation". In: *Proceedings of the AAAI Conference on Artificial Intelligence.* Volume 33. 2019, pages 8279–8286.

[63] Fink, J. P. and Rheinboldt, W. C. "Solution manifolds and submanifolds of parametrized equations and their discretization errors". In: *Numerische Mathematik* 45.3 (1984), pages 323–343.

[64] FitzHugh, R. "Impulses and physiological states in theoretical models of nerve membrane". In: *Biophysical Journal* 1.6 (1961), pages 455–466.

[65] Franco, N. R., Manzoni, A., and Zunino, P. "A deep learning approach to reduced order modelling of parameter dependent partial differential equations". In: *Mathematics of Computation* 92.340 (2023), pages 483–524.

[66] Franco, N. R., Fresca, S., Manzoni, A., and Zunino, P. "Approximation bounds for convolutional neural networks in operator learning". In: *Neural Networks* 161 (2023), pages 129–141.

[67] Franco, N. R., Manzoni, A., and Zunino, P. "Learning Operators with Mesh-Informed Neural Networks". In: *arXiv preprint arXiv:2203.11648* (2022).

[68] Franco, N. R., Massi, M. C., Ieva, F., Manzoni, A., Paganoni, A. M., Zunino, P., Veldeman, L., et al. "Development of a method for generating SNP interaction-aware polygenic risk scores for radiotherapy toxicity". In: *Radiotherapy and Oncology* 159 (2021), pages 241–248.

[69] Fraulin, D. “Deep Learning-based reduced order models for PDEs: multi-fidelity strategies for transfer learning”. In: *Master thesis in Mathematical Engineering, Politecnico di Milano* (2022), supervised by Andrea Manzoni and Nicola Rares Franco.

[70] Fresca, S., Dede, L., and Manzoni, A. “A comprehensive deep learning-based approach to reduced order modeling of nonlinear time-dependent parametrized PDEs”. In: *Journal of Scientific Computing* 87.2 (2021), pages 1–36.

[71] Fresca, S. and Manzoni, A. “POD-DL-ROM: enhancing deep learning-based reduced order models for nonlinear parametrized PDEs by proper orthogonal decomposition”. In: *Computer Methods in Applied Mechanics and Engineering* 388 (2022), page 114181.

[72] Geist, M., Petersen, P., Raslan, M., Schneider, R., and Kutyniok, G. “Numerical solution of the parametric diffusion equation by deep neural networks”. In: *Journal of Scientific Computing* 88.1 (2021), pages 1–37.

[73] Goldberg, Y. “A primer on neural network models for natural language processing”. In: *Journal of Artificial Intelligence Research* 57 (2016), pages 345–420.

[74] Greif, C. and Urban, K. “Decay of the Kolmogorov N-width for wave problems”. In: *Applied Mathematics Letters* 96 (2019), pages 216–222.

[75] Grogan, J. A., Markelc, B., Connor, A. J., Muschel, R. J., Pitt-Francis, J. M., Maini, P. K., and Byrne, H. M. “Predicting the influence of microvascular structure on tumor response to radiotherapy”. In: *IEEE Transactions on Biomedical Engineering* 64.3 (2016), pages 504–511.

[76] Gruber, A., Gunzburger, M., Ju, L., and Wang, Z. “A comparison of neural network architectures for data-driven reduced-order modeling”. In: *Computer Methods in Applied Mechanics and Engineering* 393 (2022), page 114764.

[77] Guerrero, R. F., Scarpino, S. V., Rodrigues, J. V., Hartl, D. L., and Ogbunugafor, C. B. “Proteostasis environment shapes higher-order epistasis operating on antibiotic resistance”. In: *Genetics* 212.2 (2019), pages 565–575.

[78] Gühring, I., Kutyniok, G., and Petersen, P. “Error bounds for approximations with deep ReLU neural networks in W s, p norms”. In: *Analysis and Applications* 18.05 (2020), pages 803–859.

[79] Gühring, I. and Raslan, M. “Approximation rates for neural networks with encodable weights in smoothness spaces”. In: *Neural Networks* 134 (2021), pages 107–130.

[80] Guo, M. and Hesthaven, J. S. “Reduced order modeling for nonlinear structural analysis using Gaussian process regression”. In: *Computer methods in applied mechanics and engineering* 341 (2018), pages 807–826.

[81] Habier, D., Fernando, R. L., and Garrick, D. J. “Genomic BLUP Decoded: A Look into the Black Box of Genomic Prediction”. In: *Genetics* 194 (2013), pages 597–607.

[82] He, J., Li, L., and Xu, J. “Approximation Properties of Deep ReLU CNNs”. In: *arXiv preprint arXiv:2109.00190* (2021).

[83] He, K., Zhang, X., Ren, S., and Sun, J. “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification”. In: *Proceedings of the IEEE international conference on computer vision.* 2015, pages 1026–1034.

[84] Hinton, G. E. and Salakhutdinov, R. R. “Reducing the dimensionality of data with neural networks”. In: *science* 313.5786 (2006), pages 504–507.

[85] Ho, D. S. W., Schierding, W., Wake, M., Saffery, R., and O’Sullivan, J. “Machine learning SNP based prediction for precision medicine”. In: *Frontiers in genetics* 10 (2019), page 267.

[86] Hocking, J. G. and Young, G. S. *Topology.* Courier Corporation, 1988.

[87] Hoerl, A. E. and Kennard, R. W. “Ridge regression: Biased estimation for nonorthogonal problems”. In: *Technometrics* 12.1 (1970), pages 55–67.

[88] Hoffmann, H., Wald, A., and Nguyen, T. T. N. "Parameter identification for elliptic boundary value problems: an abstract framework and applications". In: *Inverse Problems* 38.7 (2022), page 075005.

[89] Hüls, A. and Czamara, D. "Methodological challenges in constructing DNA methylation risk scores". In: *Epigenetics* 15.1-2 (2020), pages 1–11.

[90] Ij, H. "Statistics versus machine learning". In: *Nat Methods* 15.4 (2018), page 233.

[91] Jacod, J. and Protter, P. *Probability essentials.* Springer Science & Business Media, 2004.

[92] Janssens, A. C. J. "Validity of polygenic risk scores: are we measuring what we think we are?" In: *Human molecular genetics* 28.R2 (2019), R143–R150.

[93] Jiang, J., Chen, Y., and Narayan, A. "A goal-oriented reduced basis methods-accelerated generalized polynomial chaos algorithm". In: *SIAM/ASA Journal on Uncertainty Quantification* 4.1 (2016), pages 1398–1420.

[94] Jostins, L. and Barrett, J. C. "Genetic risk prediction in complex disease". In: *Human molecular genetics* 20.R2 (2011), R182–R188.

[95] Kang, J., Coates, J. T., Strawderman, R. L., Rosenstein, B. S., and Kerns, S. L. "Genomics models in radiotherapy: From mechanistic to machine learning". In: *Medical physics* 47.5 (2020), e203–e217.

[96] Karr, A. *Probability.* 3Island Press, 1993. ISBN: 9781461208921. URL: `https://books.google.it/books?id=H%5C_4hswEACAAJ`.

[97] Katznelson, Y. *An introduction to harmonic analysis.* Cambridge University Press, 2004.

[98] Kerns, S. L., Dorling, L., Fachal, L., Bentzen, S., Pharoah, P. D., Barnes, D. R., Gómez-Caamaño, A., et al. "Meta-analysis of genome wide association studies identifies genetic markers of late toxicity following radiotherapy for prostate cancer". In: *EBioMedicine* 10 (2016), pages 150–163.

[99] Kerns, S. L., Fachal, L., Dorling, L., Barnett, G. C., Baran, A., Peterson, D. R., Hollenberg, M., et al. "Radiogenomics consortium genome-wide association study meta-analysis of late toxicity after prostate cancer radiotherapy". In: *JNCI: Journal of the National Cancer Institute* 112.2 (2020), pages 179–190.

[100] Kerns, S. L., Stock, R. G., Stone, N. N., Rath, L., Vega, A., Fachal, L., Gómez-Caamaño, A., et al. "Genome-wide association study identifies a region on chromosome 11q14. 3 associated with late rectal bleeding following radiation therapy for prostate cancer". In: *Radiotherapy and Oncology* 107.3 (2013), pages 372–376.

[101] Kim, J. and Park, C. "End-to-end ego lane estimation based on sequential transfer learning for self-driving cars". In: *Proceedings of the IEEE conference on computer vision and pattern recognition workshops.* 2017, pages 30–38.

[102] Kingma, D. P. and Ba, J. "Adam: A method for stochastic optimization". In: *conference paper at the 3rd International Conference for Learning Representations* (2015), San Diego.

[103] Knezevic, D. J. and Patera, A. T. "A certified reduced basis method for the Fokker–Planck equation of dilute polymeric fluids: FENE dumbbells in extensional flow". In: *SIAM Journal on Scientific Computing* 32.2 (2010), pages 793–817.

[104] Knight, P. A. "Fast rectangular matrix multiplication and QR decomposition". In: *Linear algebra and its applications* 221 (1995), pages 69–81.

[105] Kolmogoroff, A. "Uber die beste Annaherung von Funktionen einer gegebenen Funktionenklasse". In: *Annals of Mathematics* (1936), pages 107–110.

[106] Koltovaya, N. A., Arman, I. P., and Devin, A. B. "Mutations of the CDC28 gene and the radiation sensitivity of Saccharomyces cerevisiae". In: *Yeast* 14.2 (1998), pages 133–146.

[107] Kosorok, M. R. and Laber, E. B. “Precision medicine”. In: *Annual review of statistics and its application* 6 (2019), page 263.

[108] Kovachki, N., Lanthaler, S., and Mishra, S. “On universal approximation and error bounds for Fourier Neural Operators”. In: *Journal of Machine Learning Research* 22 (2021), Art–No.

[109] Kovachki, N., Li, Z., Liu, B., Azizzadenesheli, K., Bhattacharya, K., Stuart, A., and Anandkumar, A. “Neural operator: Learning maps between function spaces”. In: *arXiv preprint arXiv:2108.08481* (2021).

[110] Kramer, M. A. “Nonlinear principal component analysis using autoassociative neural networks”. In: *AIChE journal* 37.2 (1991), pages 233–243.

[111] Krogh, A. “The number and distribution of capillaries in muscles with calculations of the oxygen pressure head necessary for supplying the tissue”. In: *The Journal of physiology* 52.6 (1919), page 409.

[112] Kutyniok, G., Petersen, P., Raslan, M., and Schneider, R. “A theoretical analysis of deep neural networks and parametric PDEs”. In: *Constructive Approximation* 55.1 (2022), pages 73–125.

[113] Laakmann, F. and Petersen, P. “Efficient approximation of solutions of parametric linear transport equations by ReLU DNNs”. In: *Advances in Computational Mathematics* 47.1 (2021), pages 1–32.

[114] Landoni, V., Fiorino, C., Cozzarini, C., Sanguineti, G., Valdagni, R., and Rancati, T. “Predicting toxicity in radiotherapy for prostate cancer”. In: *Physica Medica* 32.3 (2016), pages 521–532.

[115] Lanthaler, S., Mishra, S., and Karniadakis, G. E. “Error estimates for deeponets: A deep learning framework in infinite dimensions”. In: *Transactions of Mathematics and Its Applications* 6.1 (2022), tnac001.

[116] Lassila, T., Manzoni, A., Quarteroni, A., and Rozza, G. “Generalized reduced basis methods and n-width estimates for the approximation of the solution manifold of parametric PDEs”. In: *Analysis and numerics of partial differential equations*. Springer, 2013, pages 307–329.

[117] Laurino, F. and Zunino, P. “Derivation and analysis of coupled PDEs on manifolds with high dimensionality gap arising from topological model reduction”. In: *ESAIM: Mathematical Modelling and Numerical Analysis* 53.6 (2019), pages 2047–2080.

[118] Lee, K. and Carlberg, K. T. “Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders”. In: *Journal of Computational Physics* 404 (2020), page 108973.

[119] Lee, K.-Y., Leung, K.-S., Ma, S. L., So, H. C., Huang, D., Tang, N. L.-S., and Wong, M.-H. “Genome-Wide Search for SNP Interactions in GWAS Data: Algorithm, Feasibility, Replication Using Schizophrenia Datasets”. In: *Frontiers in genetics* 11 (2020).

[120] Lehner, B. “Modelling genotype–phenotype relationships and human disease with genetic interaction networks”. In: *Journal of Experimental Biology* 210.9 (2007), pages 1559–1566.

[121] Lehner, B. “Molecular mechanisms of epistasis within and between genes”. In: *Trends in Genetics* 27.8 (2011), pages 323–331.

[122] Li, X., Liao, B., Cai, L., Cao, Z., and Zhu, W. “Informative SNPs selection based on two-locus and multilocus linkage disequilibrium: Criteria of max-correlation and min-redundancy”. In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 10.3 (2013), pages 688–695.

[123] Lim, M. and Hastie, T. “Learning interactions via hierarchical group-lasso regularization”. In: *Journal of Computational and Graphical Statistics* 24.3 (2015), pages 627–654.

[124] Liu, D. C. and Nocedal, J. “On the limited memory BFGS method for large scale optimization”. In: *Mathematical programming* 45.1 (1989), pages 503–528.

[125] Loshchilov, I. and Hutter, F. “Decoupled weight decay regularization”. In: *Conference paper at ICLR 2019* (2017), International Conference on Learning Representations.

[126] Lu, L., Jin, P., Pang, G., Zhang, Z., and Karniadakis, G. E. “Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators”. In: *Nature Machine Intelligence* 3.3 (2021), pages 218–229.

[127] Luo, Z. and Chen, G. *Proper orthogonal decomposition methods for partial differential equations.* Academic Press, 2018.

[128] Lye, K. O., Mishra, S., and Molinaro, R. “A multi-level procedure for enhancing accuracy of machine learning algorithms”. In: *European Journal of Applied Mathematics* 32.3 (2021), pages 436–469.

[129] Mackay, T. F. and Moore, J. H. “Why epistasis is important for tackling complex human disease genetics”. In: *Genome medicine* 6.6 (2014), pages 1–3.

[130] Manolio, T. A., Collins, F. S., Cox, N. J., Goldstein, D. B., Hindorff, L. A., Hunter, D. J., McCarthy, M. I., et al. “Finding the missing heritability of complex diseases”. In: *Nature* 461.7265 (2009), pages 747–753.

[131] Marcati, C. and Schwab, C. “Exponential convergence of deep operator networks for elliptic partial differential equations”. In: *arXiv preprint* (2021), arXiv:2112.08125.

[132] Marshall, M. and Solomon, S. “Hereditary breast–ovarian cancer: clinical findings and medical management”. In: *Plastic Surgical Nursing* 27.3 (2007), pages 124–127.

[133] Massi, M. C., Franco, N. R., Manzoni, A., Paganoni, A. M., Park, H. A., Hoffmeister, M., Brenner, H., et al. “Learning High-Order Interactions for Polygenic Risk Prediction”. In: *PLOS ONE* (2023).

[134] Massi, M. C., Gasperoni, F., Ieva, F., and Paganoni, A. M. “Feature selection for imbalanced data with deep sparse autoencoders ensemble”. In: *Statistical Analysis and Data Mining: The ASA Data Science Journal* 15.3 (2022), pages 376–395.

[135] Massi, M. C., Gasperoni, F., Ieva, F., Paganoni, A. M., Zunino, P., Manzoni, A., Franco, N. R., et al. “A deep learning approach validates genetic risk factors for late toxicity after prostate cancer radiotherapy in a REQUITE multi-national cohort”. In: *Frontiers in oncology* 10 (2020), page 541281.

[136] Mishra, S. and Molinaro, R. “Estimates on the generalization error of physics-informed neural networks for approximating a class of inverse problems for PDEs”. In: *IMA Journal of Numerical Analysis* 42.2 (2022), pages 981–1022.

[137] Mishra, S. and Rusch, T. K. “Enhancing accuracy of deep learning algorithms by training with low-discrepancy sequences”. In: *SIAM Journal on Numerical Analysis* 59.3 (2021), pages 1811–1834.

[138] Mohri, M., Rostamizadeh, A., and Talwalkar, A. *Foundations of machine learning.* MIT press, 2018.

[139] Monga, I., Qureshi, A., Thakur, N., Gupta, A. K., and Kumar, M. “ASPsiRNA: a resource of ASP-siRNAs having therapeutic potential for human genetic disorders and algorithm for prediction of their inhibitory efficacy”. In: *G3: Genes, Genomes, Genetics* 7.9 (2017), pages 2931–2943.

[140] Moore, J. H. “The ubiquitous nature of epistasis in determining susceptibility to common human diseases”. In: *Human heredity* 56.1-3 (2003), pages 73–82.

[141] Moore, J. H. and Williams, S. M. “Epistasis and its implications for personal genetics”. In: *The American Journal of Human Genetics* 85.3 (2009), pages 309–320.

[142] Mottram, J. “A factor of importance in the radio sensitivity of tumours”. In: *The British Journal of Radiology* 9.105 (1936), pages 606–614.

[143] Mücke, N. T., Bohté, S. M., and Oosterlee, C. W. “Reduced order modeling for parameterized time-dependent PDEs using spatially and memory aware deep learning”. In: *Journal of Computational Science* 53 (2021), page 101408.

[144] Multhaup, M. L., Kita, R., Krock, B., Eriksson, N., Fontanillas, P., Aslibekyan, S., Del Gobbo, L., et al. “The science behind 23andMe’s Type 2 Diabetes report”. In: *Sunnyvale (CA): 23andMe* (2019), pages 23–19.

[145] Nagumo, J., Arimoto, S., and Yoshizawa, S. “An active pulse transmission line simulating nerve axon”. In: *Proceedings of the IRE* 50.10 (1962), pages 2061–2070.

[146] Necas, J. “Les méthodes directes en théorie des équations elliptiques”. In: *Paris: Masson et Cie, Éditeurs, Prague: Academia, Éditeurs* (1967), pages 90–104.

[147] Negri, F., Manzoni, A., and Amsallem, D. “Efficient model reduction of parametrized systems by matrix discrete empirical interpolation”. In: *Journal of Computational Physics* 303 (2015), pages 431–454.

[148] Nguyen, T. V. and Eisman, J. A. “Post-GWAS Polygenic Risk Score: Utility and Challenges”. In: *JBMR plus* 4.11 (2020), e10411.

[149] Nociti Jr, F. H., Casati, M. Z., and Duarte, P. M. “Current perspective of the impact of smoking on the progression and treatment of periodontitis”. In: *Periodontology 2000* 67.1 (2015), pages 187–210.

[150] Ohlberger, M. and Rave, S. “Reduced basis methods: Success, limitations and future challenges”. In: *Proceedings of ALGORITMY 2016* (2016), pages 1–12.

[151] Okser, S., Pahikkala, T., Airola, A., Salakoski, T., Ripatti, S., and Aittokallio, T. “Regularized machine learning in the genetic prediction of complex traits”. In: *PLoS genetics* 10.11 (2014), e1004754.

[152] Pagani, S., Manzoni, A., and Quarteroni, A. “Numerical approximation of parametrized problems in cardiac electrophysiology by a local reduced basis method”. In: *Computer Methods in Applied Mechanics and Engineering* 340 (2018), pages 530–558.

[153] Park, S., Yun, C., Lee, J., and Shin, J. “Minimum width for universal approximation”. In: *arXiv preprint arXiv:2006.08859* (2020).

[154] Park, Y. J. “On the Continuity of the Hardy-Littlewood Maximal Function”. In: *Journal of the Chungcheong Mathematical Society* 31.1 (2018), pages 43–43.

[155] Perera, S., Barnes, N., He, X., Izadi, S., Kohli, P., and Glocker, B. “Motion segmentation of truncated signed distance function based volumetric surfaces”. In: *2015 IEEE Winter Conference on Applications of Computer Vision*. IEEE. 2015, pages 1046–1053.

[156] Petersen, P. and Voigtlaender, F. “Equivalence of approximation by convolutional neural networks and fully-connected networks”. In: *Proceedings of the American Mathematical Society* 148.4 (2020), pages 1567–1581.

[157] Petersen, P. and Voigtlaender, F. “Optimal approximation of piecewise smooth functions using deep ReLU neural networks”. In: *Neural Networks* 108 (2018), pages 296–330.

[158] Pfaff, T., Fortunato, M., Sanchez-Gonzalez, A., and Battaglia, P. W. “Learning mesh-based simulation with graph networks”. In: *arXiv preprint* (2020), arXiv:2010.03409.

[159] Pinkus, A. “Approximation theory of the MLP model in neural networks”. In: *Acta numerica* 8 (1999), pages 143–195.

[160] Possenti, L., Cicchetti, A., Rosati, R., Cerroni, D., Costantino, M. L., Rancati, T., and Zunino, P. “A mesoscale computational model for microvascular oxygen transfer”. In: *Annals of Biomedical Engineering* 49.12 (2021), pages 3356–3373.

[161] Possenti, L., Di Gregorio, S., Gerosa, F. M., Raimondi, G., Casagrande, G., Costantino, M. L., and Zunino, P. "A computational model for microcirculation including Fahraeus-Lindqvist effect, plasma skimming and fluid exchange with the tissue interstitium". In: *International Journal for Numerical Methods in Biomedical Engineering* 35.3 (2019), e3165.

[162] Quarteroni, A., Manzoni, A., and Negri, F. *Reduced basis methods for partial differential equations: an introduction.* Volume 92. Springer, 2015.

[163] Quarteroni, A. and Valli, A. *Numerical approximation of partial differential equations.* Volume 23. Springer Science & Business Media, 2008.

[164] Raissi, M., Perdikaris, P., and Karniadakis, G. E. "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations". In: *Journal of Computational physics* 378 (2019), pages 686–707.

[165] Rancati, T., Massi, M. C., Franco, N. R., Avuzzi, B., Azria, D., Choudhury, A., Cicchetti, A., et al. "Prediction of toxicity after prostate cancer RT: the value of a SNP-interaction polygenic risk score". In: *Radiotherapy and Oncology* 161.S1 (2021), pages 526–528.

[166] Regazzoni, F., Pagani, S., Cosenza, A., Lombardi, A., and Quarteroni, A. "A physics-informed multi-fidelity approach for the estimation of differential equations parameters in low-data or large-noise regimes". In: *Rendiconti Lincei* 32.3 (2021), pages 437–470.

[167] Ritchie, M. D., Hahn, L. W., Roodi, N., Bailey, L. R., Dupont, W. D., Parl, F. F., and Moore, J. H. "Multifactor-dimensionality reduction reveals high-order interactions among estrogen-metabolism genes in sporadic breast cancer". In: *The American Journal of Human Genetics* 69.1 (2001), pages 138–147.

[168] Rohan, E., Lukeš, V., and Jonášová, A. "Modeling of the contrast-enhanced perfusion test in liver based on the multi-compartment flow in porous media". In: *Journal of mathematical biology* 77.2 (2018), pages 421–454.

[169] Rosenblatt, F. *Principles of neurodynamics. perceptrons and the theory of brain mechanisms.* Technical report. Cornell Aeronautical Lab Inc Buffalo NY, 1961.

[170] Rosenblatt, F. "The perceptron: a probabilistic model for information storage and organization in the brain." In: *Psychological review* 65.6 (1958), page 386.

[171] Sassano, M., Mariani, M., Quaranta, G., Pastorino, R., and Boccia, S. "Polygenic risk prediction models for colorectal cancer: a systematic review". In: *BMC cancer* 22.1 (2022), pages 1–21.

[172] Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. "The graph neural network model". In: *IEEE transactions on neural networks* 20.1 (2008), pages 61–80.

[173] Schierding, W., Antony, J., Karhunen, V., Vääräsmäki, M., Franks, S., Elliott, P., Kajantie, E., et al. "GWAS on prolonged gestation (post-term birth): analysis of successive Finnish birth cohorts". In: *Journal of Medical Genetics* 55.1 (2018), pages 55–63.

[174] Schmidhuber, J. "Deep learning in neural networks: An overview". In: *Neural networks* 61 (2015), pages 85–117.

[175] Schulze, M. B., Weikert, C., Pischon, T., Bergmann, M. M., Al-Hasani, H., Schleicher, E., Fritsche, A., et al. "Use of multiple metabolic and genetic markers to improve the prediction of type 2 diabetes: the EPIC-Potsdam Study". In: *Diabetes care* 32.11 (2009), pages 2116–2119.

[176] Schwab, C. and Zech, J. "Deep learning in high dimension: Neural network expression rates for generalized polynomial chaos expansions in UQ". In: *Analysis and Applications* 17.01 (2019), pages 19–55.

[177] Secomb, T. W. and Pries, A. R. “The microcirculation: physiology at the mesoscale”. In: *The Journal of physiology* 589.5 (2011), pages 1047–1052.

[178] Seibold, P., Behrens, S., Schmezer, P., Helmbold, I., Barnett, G., Coles, C., Yarnold, J., et al. “XRCC1 polymorphism associated with late toxicity after radiation therapy in breast cancer patients”. In: *International Journal of Radiation Oncology* Biology* Physics* 92.5 (2015), pages 1084–1092.

[179] Seibold, P., Webb, A., Aguado-Barrera, M. E., Azria, D., Bourgier, C., Brengues, M., Briers, E., et al. “REQUITE: a prospective multicentre cohort study of patients undergoing radiotherapy for breast, lung or prostate cancer”. In: *Radiotherapy and Oncology* 138 (2019), pages 59–67.

[180] Shah, A., Xing, W., and Triantafyllidis, V. “Reduced-order modelling of parameter-dependent, linear and nonlinear dynamic partial differential equation models”. In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 473.2200 (2017), page 20160809.

[181] Shi, J., Park, J.-H., Duan, J., Berndt, S. T., Moy, W., Yu, K., Song, L., et al. “Winner’s curse correction and variable thresholding improve performance of polygenic risk modeling based on genome-wide association study summary-level data”. In: *PLoS genetics* 12.12 (2016), e1006493.

[182] Siegel, J. W. and Xu, J. “High-order approximation rates for shallow neural networks with cosine and ReLUk activation functions”. In: *Applied and Computational Harmonic Analysis* 58 (2022), pages 1–26.

[183] Silver, M., Chen, P., Li, R., Cheng, C.-Y., Wong, T.-Y., Tai, E.-S., Teo, Y.-Y., et al. “Pathways-driven sparse regression identifies pathways and genes associated with high-density lipoprotein cholesterol in two Asian cohorts”. In: *PLoS genetics* 9.11 (2013), e1003939.

[184] Skopenkov, A. B. “Embedding and knotting of manifolds in Euclidean spaces”. In: *London Mathematical Society Lecture Note Series* 347 (2008), page 248.

[185] Sohail, M., Maier, R. M., Ganna, A., Bloemendal, A., Martin, A. R., Turchin, M. C., Chiang, C. W., et al. “Polygenic adaptation on height is overestimated due to uncorrected stratification in genome-wide association studies”. In: *Elife* 8 (2019), e39702.

[186] Spitzbart, A. “A generalization of Hermite’s interpolation formula”. In: *The American Mathematical Monthly* 67.1 (1960), pages 42–46.

[187] Strigari, L., Torriani, F., Manganaro, L., Inaniwa, T., Dalmasso, F., Cirio, R., and Attili, A. “Tumour control in ion beam radiotherapy with different ions in the presence of hypoxia: an oxygen enhancement ratio model based on the microdosimetric kinetic model”. In: *Physics in Medicine & Biology* 63.6 (2018), page 065012.

[188] Sultana, F., Sufian, A., and Dutta, P. “Evolution of image segmentation using deep convolutional neural network: a survey”. In: *Knowledge-Based Systems* 201 (2020), page 106062.

[189] Sur, P. and Candès, E. J. “A modern maximum-likelihood theory for high-dimensional logistic regression”. In: *Proceedings of the National Academy of Sciences* 116.29 (2019), pages 14516–14525.

[190] Taylor, M. B. and Ehrenreich, I. M. “Higher-order genetic interactions and their contribution to complex traits”. In: *Trends in genetics* 31.1 (2015), pages 34–40.

[191] Tibshirani, R. “Regression shrinkage and selection via the lasso”. In: *Journal of the Royal Statistical Society: Series B (Methodological)* 58.1 (1996), pages 267–288.

[192] Torkamani, A., Wineinger, N. E., and Topol, E. J. “The personal and clinical utility of polygenic risk scores”. In: *Nature Reviews Genetics* 19.9 (2018), pages 581–590.

[193] Vilhjálmsson, B. J., Yang, J., Finucane, H. K., Gusev, A., and Lindström, S. e. a. “Modeling Linkage Disequilibrium Increases Accuracy of Polygenic Risk Scores”. In: *Am J Hum Genet* 97.4 (2015), pages 576–592. DOI: `10.1016/j.ajhg.2015.09.001`.

[194] Visscher, P. M., Wray, N. R., Zhang, Q., Sklar, P., McCarthy, M. I., Brown, M. A., and Yang, J. "10 years of GWAS discovery: biology, function, and translation". In: *The American Journal of Human Genetics* 101.1 (2017), pages 5–22.

[195] Vivian-Griffiths, T., Baker, E., Schmidt, K. M., Bracher-Smith, M., Walters, J., Artemiou, A., Holmans, P., et al. "Predictive modeling of schizophrenia from genomic data: Comparison of polygenic risk score with kernel support vector machines approach". In: *American Journal of Medical Genetics Part B: Neuropsychiatric Genetics* 180.1 (2019), pages 80–85.

[196] Wang, S., Wang, H., and Perdikaris, P. "Learning the solution operator of parametric partial differential equations with physics-informed DeepONets". In: *Science advances* 7.40 (2021), eabi8605.

[197] Wei, Z., Wang, K., Qu, H.-Q., Zhang, H., Bradfield, J., Kim, C., Frackleton, E., et al. "From disease association to risk assessment: an optimistic view from genome-wide association studies on type 1 diabetes". In: *PLoS genetics* 5.10 (2009), e1000678.

[198] Weldy, C. S. and Ashley, E. A. "Towards precision medicine in heart failure". In: *Nature Reviews Cardiology* 18.11 (2021), pages 745–762.

[199] West, C. M. and Barnett, G. C. "Genetics and genomics of radiotherapy toxicity: towards prediction". In: *Genome medicine* 3.8 (2011), pages 1–15.

[200] Wijk, Y. van, Vanneste, B. G., Jochems, A., Walsh, S., Oberije, C. J., Pinkawa, M., Ramaekers, B. L., et al. "Development of an isotoxic decision support system integrating genetic markers of toxicity for the implantation of a rectum spacer". In: *Acta Oncologica* 57.11 (2018), pages 1499–1505.

[201] Xu, J., Pradhan, A., and Duraisamy, K. "Conditionally parameterized, discretization-aware neural networks for mesh-based modeling of physical systems". In: *Advances in Neural Information Processing Systems* 34 (2021), pages 1634–1645.

[202] Yarotsky, D. "Error bounds for approximations with deep ReLU networks". In: *Neural Networks* 94 (2017), pages 103–114.

[203] Yosida, K. *Functional analysis*. Springer Science & Business Media, 2012.

[204] Yuan, M. and Lin, Y. "Model selection and estimation in regression with grouped variables". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 68.1 (2006), pages 49–67.

[205] Zhang, H., Gao, X., Unterman, J., and Arodz, T. "Approximation capabilities of neural ODEs and invertible residual networks". In: *International Conference on Machine Learning*. PMLR. 2020, pages 11086–11095.

[206] Zhou, D.-X. "Universality of deep convolutional neural networks". In: *Applied and computational harmonic analysis* 48.2 (2020), pages 787–794.

[207] Zhu, Y. and Zabaras, N. "Bayesian deep convolutional encoder–decoder networks for surrogate modeling and uncertainty quantification". In: *Journal of Computational Physics* 366 (2018), pages 415–447.

[208] Zou, H. and Hastie, T. "Regression shrinkage and selection via the elastic net, with applications to microarrays". In: *JR Stat Soc Ser B* 67 (2003), pages 301–20.